



ibW Höhere Fachschule Südostschweiz

# Diplomarbeit Technik und Wirtschaftsinformatik 2023-2024

**Titel der Arbeit:** PostgreSQL HA Cluster - Konzeption und Implementation

**Name:** Gruber

**Vorname:** Michael

**Klasse:** DIPL. INFORMATIKER/-IN HF - 10.0002A-2021

**Firma:** Kantonsspital Graubünden

## **Zusammenfassung**

Disposition für die Diplomarbeit von Michael Graber. Ziel der Arbeit ist die Evaluation, Konzeption und Implementation eines PostgreSQL HA Clusters für das Kantonsspital Graubünden.

## **Management Summary**

Diplomarbeit Michael Graber

## Inhaltsverzeichnis

<b>1 Einleitung</b>	<b>1</b>
1.1 Ausgangslage und Problemstellung . . . . .	1
1.1.1 Das Kantonsspital Graubünden . . . . .	1
1.1.2 Die ICT des Kantonsspital Graubünden . . . . .	3
1.1.3 Rolle in der ICT vom Kantonsspital Graubünden . . . . .	5
1.1.4 Ausgangslage . . . . .	6
1.1.5 Problemstellung . . . . .	10
1.2 Zieldefinition . . . . .	14
1.3 Abgrenzungen . . . . .	17
1.4 Abhängigkeiten . . . . .	19
<b>2 Projektmanagement</b>	<b>21</b>
2.1 Risikomanagement . . . . .	22
2.1.1 Riskcontrolling . . . . .	24
2.2 Vorgehensweise und Methoden . . . . .	27
2.3 Projektplanung . . . . .	27
2.3.1 Projektcontrolling . . . . .	28
2.3.2 GANTT-Diagramm . . . . .	30
2.4 Expertengespräche . . . . .	32
<b>3 Umsetzung</b>	<b>33</b>
3.1 Evaluation . . . . .	33
3.1.1 Exkurs Architektur . . . . .	33
3.1.2 Erheben und Gewichten der Anforderungen . . . . .	39
3.1.3 Testziele erarbeiten . . . . .	44
3.1.4 PostgreSQL Benchmarking . . . . .	47
3.1.5 Analyse gängiger PostgreSQL HA Cluster Lösungen . . . . .	52
3.1.6 Vorauswahl . . . . .	75
3.1.7 Installation verschiedener Lösungen . . . . .	75
3.1.8 Testing Evaluationssysteme . . . . .	83
3.1.9 Gegenüberstellung der Lösungen . . . . .	84
3.1.10 Entscheid . . . . .	92
3.2 Aufbau und Implementation Testsystem . . . . .	92
3.2.1 Bereitstellen der Grundinfrastruktur . . . . .	92
3.2.2 Installation und Konfiguration PostgreSQL HA Cluster . . . . .	92

## Diplomarbeit

3.2.3	Technical Review der Umgebung . . . . .	92
3.3	Testing . . . . .	92
3.3.1	Testing . . . . .	92
3.3.2	Protokollierung . . . . .	92
3.3.3	Review und Auswertung . . . . .	93
3.4	Troubleshooting und Lösungsfindung . . . . .	93
<b>4</b>	<b>Resultate</b>	<b>94</b>
4.1	Zielüberprüfung . . . . .	94
4.2	Schlussfolgerung . . . . .	94
4.3	Weiteres Vorgehen / offene Arbeiten . . . . .	94
4.4	Persönliches Fazit . . . . .	94
<b>Abbildungsverzeichnis</b>		<b>95</b>
<b>Tabellenverzeichnis</b>		<b>98</b>
<b>Listings</b>		<b>99</b>
<b>Literatur</b>		<b>101</b>
<b>Abkürzungen</b>		<b>107</b>
<b>Glossar</b>		<b>109</b>
<b>Anhang</b>		<b>i</b>
I	Arbeitsrapport . . . . .	i
II	Protokoll - Fachgespräche . . . . .	ii
III	Statusbericht . . . . .	iii
III.I	Status Report 1 . . . . .	iii
III.II	Status Report 2 . . . . .	iii
IV	Kommentare / Anmerkungen . . . . .	iv
V	Evaluation . . . . .	vi
V.I	Maintenance - CloudNativePG . . . . .	vi
V.II	Maintenance - Patroni . . . . .	ix
V.III	Maintenance - StackGres - Citus . . . . .	xii
V.IV	Maintenance - YugabyteDB . . . . .	xix
VI	Evaluationssysteme - Installation . . . . .	xxii
VI.I	rke2 . . . . .	xxii
VI.II	yugabyteDB . . . . .	xxviii
VI.III	Stackgres mit Citus . . . . .	xli
VI.IV	sk9016 - YugabyteDB . . . . .	xlii

## Diplomarbeit

VI.V	Patroni . . . . .	xliii
VII	Evaluationssysteme - Testing . . . . .	xlv
VII.I	Patroni . . . . .	xlv
VII.II	StackGres - Citus . . . . .	xlv
VII.III	YugabyteDB . . . . .	xlviii
VIII	Exkurs Architekturen - Umsysteme und Prinzipien . . . . .	xlviii
VIII.I	Raft-Konsensus . . . . .	xlviii
VIII.II	local-path-provisioner . . . . .	xlviii
IX	Python Utils . . . . .	xlviii
IX.I	zotero.py . . . . .	xlviii
IX.II	zotero_bibtex_configuration.yaml . . . . .	liv
IX.III	zotero_biblatex_keystore.yaml . . . . .	lv
IX.IV	riskmatrix.py . . . . .	lxii
IX.V	riskmatrix_plotter_conf.yaml . . . . .	lxvi
IX.VI	riskmatrix_xy_axis_tuple_matrix.yaml . . . . .	lxix
IX.VII	cost_benefit_diagram.py . . . . .	lxx
IX.VIII	cost_benefit_diagram_plotter_conf.yaml . . . . .	lxxii
IX.IX	pandas_dataframe_to_latex_table.py . . . . .	lxxiii
IX.X	csv_to_latex_diplomarbeit.yaml . . . . .	lxxxii
IX.XI	pandas_data_chart_plotter.py . . . . .	cii
IX.XII	pandas_data_chart_plotter_conf.yaml . . . . .	cix

## 1 Einleitung

### 1.1 Ausgangslage und Problemstellung

#### 1.1.1 Das Kantonsspital Graubünden

Das Kantonsspital Graubünden ist das Zentrumsspital der Südostschweiz, welches Teil der sogenannten Penta Plus Spitäler ist. Die Penta plus Spitäler sind das Kantonsspital Baden, das Kantonsspital Winterthur, das Spitalzentrum Biel AG, das Kantonsspital Baselland, die Spital STS (Simmental-Thun-Saanenland) AG und eben das Kantonsspital Graubünden.

Das KSGR deckt dabei die Spitalregion Churer Rheintal ab

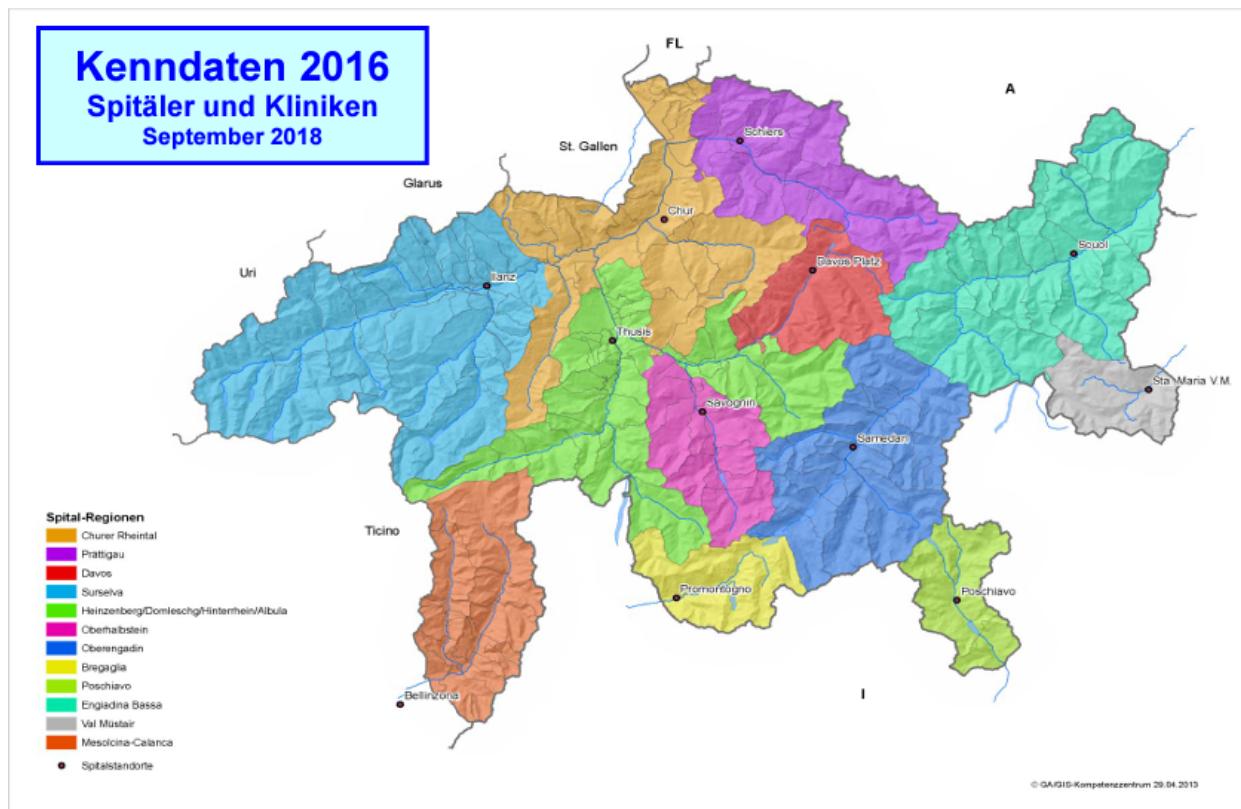


Abbildung 1.1: Spitalregionen Kanton Graubünden[53]

Seit dem 1. Januar 2023 betreibt das KSGR den Standort Walenstadt im Kanton St. Gallen und deckt primär den Wahlkreis Sarganserland ab.



Abbildung 1.2: Wahlkreise Kanton St. Gallen[78]

Da dieser Wahlkreis der Spitalregion Rheintal Werdenberg Sarganserland zugeordnet ist, wird das KSGR auch im restlichen südlichen Teil der Spitalregion aktiv sein.

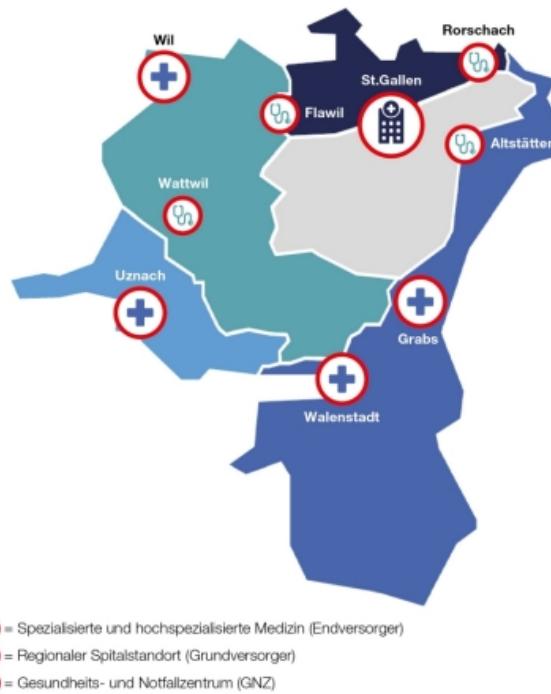


Abbildung 1.3: Spitalregionen / Spitalstrategie Kanton St. Gallen[47]

### 1.1.2 Die ICT des Kantonsspital Graubünden

Das Kantonsspital Graubünden hat eine Matrixorganisation. Die ICT ist ein eigenständiges Departement und gilt als sogenanntes Querschnittsdepartement, dh. die ICT bedient alle anderen Departemente.

## Diplomarbeit



### Organigramm des Kantonsspitals Graubünden

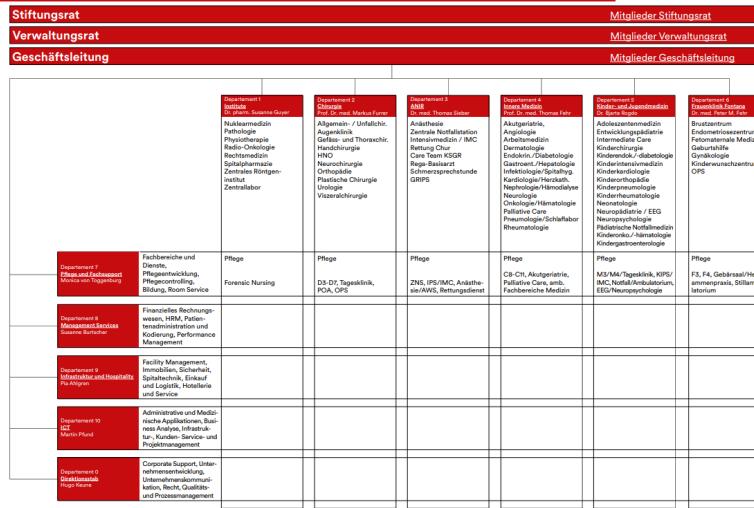
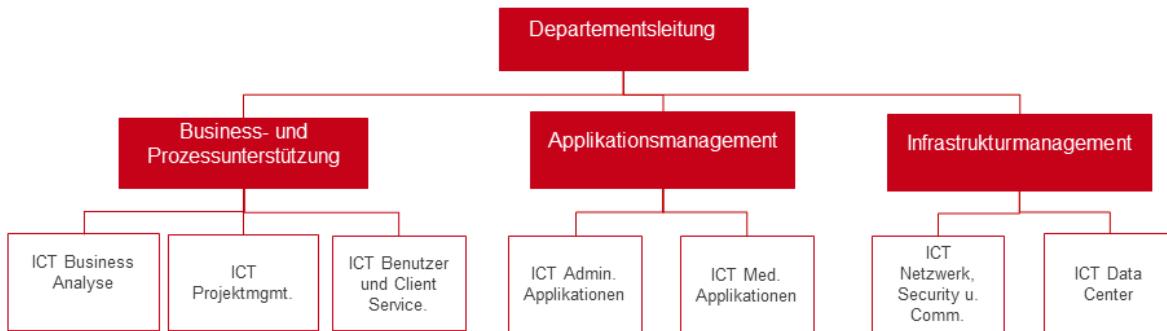


Abbildung 1.4: Organigramm Kantonsspital Graubünden

Die ICT betreibt über 400 Applikationen die auf mehr als 1055 physische und virtuelle Server und Appliances. Das Rückgrat der Infrastruktur ist dabei die Virtualisierungsplattformen VMware ESXi für Server und Citrix für die Thinclients der Enduser. Es werden aber auch Dienstleistungen für andere Spitäler und Kliniken oder andere Einrichtungen des Gesundheitswesens erbracht.

Entsprechend wurde die ICT in ein Applikationsmanagement, ein Infrastrukturmanagement sowie einem unterstützenden Bereich aufgegliedert. Das Applikationsmanagement wurde in je einen Bereich für die Administrativen und Medizinischen Applikationen aufgeteilt. Das Infrastrukturmanagement wiederum wurde in den Bereich Netzwerk und Data Center, welcher für Server zuständig ist, aufgeteilt. Der Bereich Business- und Prozessunterstützung beinhaltet je eine Abteilung für die Businessanalyse, das Projektmanagement und Benutzer- und Clientservices in der auch der Service-Desk untergebracht ist.

## (Führungs-)Organisation Departement 10 ab 2023



29.09.2023

3

Abbildung 1.5: Organigramm Departement 10 - ICT

Die Organisation der ICT wird sich aber bis spätestens zum Abschluss der Diplomarbeit noch verändern.

### 1.1.3 Rolle in der ICT vom Kantonsspital Graubünden

Meine Rolle im Kantonsspital Graubünden resp. in der ICT ist die eines DBA. Diese Rolle ist in der Abteilung ICT Data Center.

Da die Kernsysteme auf Oracle Datenbanken und HP-UX laufen, bin ich primär Oracle Database DBA und manage das HP-UX in Zusammenarbeit mit HPE. Die administrative Tätigkeit bei HP-UX besteht primär im Betrieb der HP-UX Cluster Packages (einer sehr rudimentären Art von Container), überwachen und erweitern des Filesystems, erweitern von SAN Storage Lunes für die Filesystem Erweiterung, Erstellen von PRTG-Sensoren für das Monitoring, SAP Printerqueue Management und andere Tasks die es noch auszuführen gibt. Daneben bin ich auch für andere Datenbanken, teilweise aber nur begrenzt Microsoft SQL Server, MySQL / MariaDB und vermehrt PostgreSQL zuständig. Darüber hinaus bin ich Teilweise in die Linux-Administration involviert und betreue auch noch einige Windows Server für das Zentrale klinische Informationssystem.

## Diplomarbeit

### 1.1.4 Ausgangslage

Die meisten der über 400 Applikationen, die das KSGR betreibt, haben in den allermeisten Fällen ihre Daten in Datenbanksysteme speichern. Entsprechend der Vielfalt der Applikationen existieren auch eine vielzahl an Datenbanksystemen und Versionen.

Basierend auf der Liste *DB-Engines Ranking*[44] der Top-Datenbanksysteme . Allerdings werden nicht alle Datenbanksysteme berücksichtigt, entweder weil das Datenbanksystem keine Client/Server Architektur hat oder nicht im Scope der IT oder des Projekts ist.

Folgende Datenbanken sind inventarisiert:

<b>DBMS</b>	<b>Datenbankmodell</b>	<b>Inventarisiert</b>	<b>Kommentar</b>
Oracle Database	Relational, NoSQL, OLAP	Ja	
MySQL	Relational	Ja	
Microsoft SQL Server	Relational, NoSQL, OLAP	Nein	Werden separat administriert und sind daher nicht in diesem Inventar gelistet
PostgreSQL	Relational, NoSQL	Ja	
MongoDB	NoSQL	Ja	
Redis	Key-value	Ja	
Elasticsearch	Search engine	Ja	
IBM DB2	Relational	Ja	
SQLite	Relational	Nein	Lokale Datenbank. Zudem wird die DB nicht via Netzwerk angesprochen
Microsoft Access	Relational	Nein	Nicht im Scope der ICT
Snowflake	Relational	Ja	
Cassandra	Relational	Ja	
MariaDB	Relational	Ja	
Splunk	Search engine	Ja	
Microsoft Azure SQL Database	Relational, NoSQL, OLAP	Nein	Datenbanken sind nicht On-Premise und somit nicht im Scope

Tabelle 1.1: Inventarisierte Datenbanksysteme

## Diplomarbeit

Folgende Datenbanksysteme sind demnach im KSGR im Einsatz:

	RDBMS	Instanz	Datenbanken	Appliance
0	MariaDB	2	2	0
1	MongoDB	2	2	0
2	MySQL	28	50	3
3	Oracle Database	27	30	0
4	PostgreSQL	20	20	4
5	Redis	1	1	0
Gesamtergebnis		80	105	7

Tabelle 1.2: Datenbankinventar

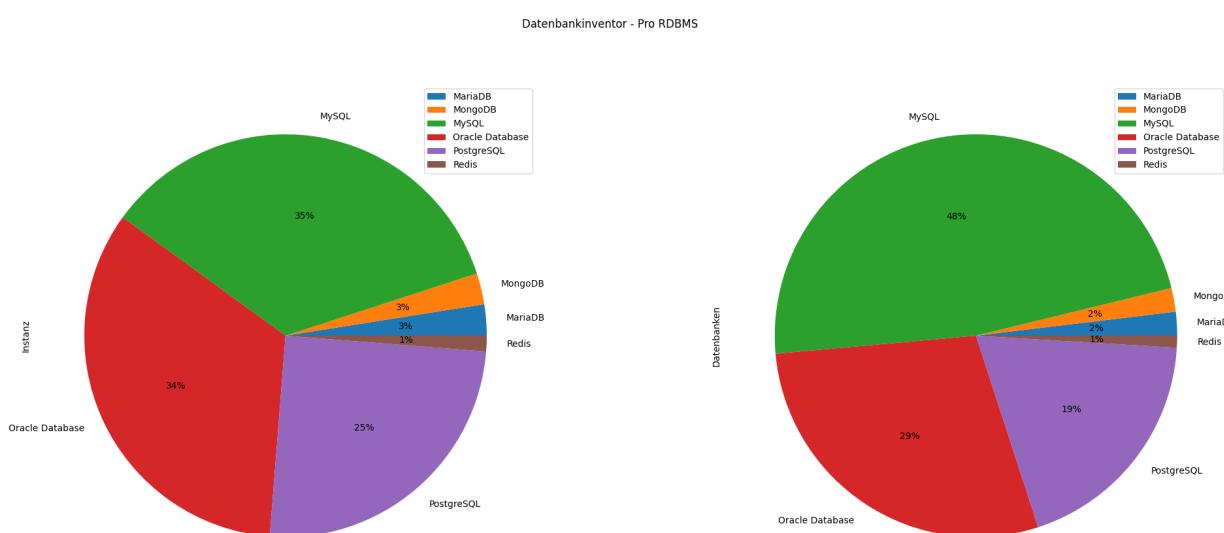


Abbildung 1.6: Datenbanken - Aufgeschlüsselt nach RDBMS

Aufgeschlüsselt auf die Betriebssysteme auf denen die Datenbanken laufen, ergibt sich folgendes Bild:

OS	RDBMS	Appliance	Datenbanken	Instanz
HP-UX	Oracle Database	0	24	21
Linux	MariaDB	0	2	2
Continued on next page				

Tabelle 1.3: Datenbankinventor - Nach Betriebssystemen aufgeschlüsselt

OS	RDBMS	Appliance	Datenbanken	Instanz
Windows Server	MySQL	3	36	14
	Oracle Database	0	1	1
	PostgreSQL	4	8	8
	Redis	0	1	1
Windows Server	MongoDB	0	2	2
	MySQL	0	14	14
	Oracle Database	0	5	5
	PostgreSQL	0	12	12
Gesamtergebnis		7	105	80

Tabelle 1.3: Datenbankinventor - Nach Betriebssystemen aufgeschlüsselt

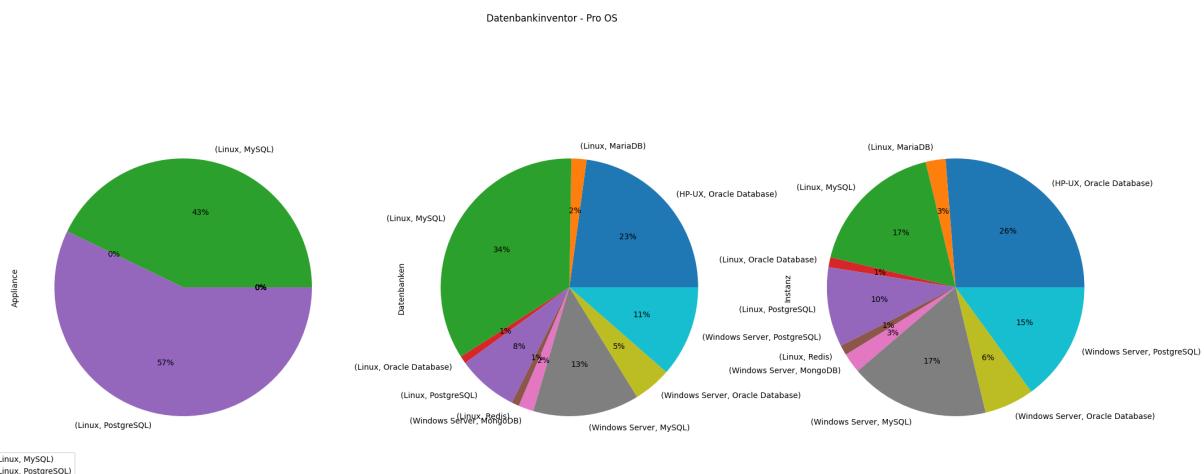


Abbildung 1.7: Datenbanken - Aufgeschlüsselt nach Betriebssystem

Die Kernsysteme des Spitals werden auf Oracle Datenbanken (Oracle Database) betrieben, die aktuell auf einer HP-UX betrieben werden. Stand heute gibt es kein Clustersystem für die Open-Source Datenbanken wie MariaDB/MySQL oder PostgreSQL.

Durch die Einführung von Kubernetes als Containerplattform wird der Bedarf an PostgreSQL Datenbanken immer grösser. Es werden in naher Zukunft auch verschiedene Oracle Datenbanken sowie MySQL Datenbanken auf PostgreSQL migriert werden.

Aktuell werden die Daten des Zabbix der Netzwerktechniker auf eine MariaDB Datenbank gespeichert, dies soll sich aber ändern. Da das Zabbix alle Netzwerkgeräte Überwacht, pro

Sekunde werden im Moment 1'200 Datenpunkte abgefragt und xxx in die Datenbank und wird im Laufe der Zeit mehrere Terrabyte gross werden.

### 1.1.5 Problemstellung

Zusammen mit den bestehenden PostgreSQL-Datenbankinstanzen werden die PostgreSQL Datenbanken in der Art, wie sie bisher Betrieben werden, nicht mehr Betreibbar sein. Die bisherige Strategie erzeugt sehr viele Aufwände und provoziert Risiken, namentlich:

- dezentrale Backups und fragmentierte Backup-Strategien
  - Fehlende Kontrolle
  - Wiederherstellbarkeit nicht garantiert
- Verschiedene Betriebssysteme mit verschiedenen Versionen
  - Fehlernder Überblick
  - Veraltete Betriebssystem- und Datenbankversionen
  - Grosser Administrationsaufwand
- Uneinheitliche Absicherung und Härtung
  - Hohe Angreifbarkeit
  - Veraltete Betriebssystem- und Datenbankversionen
  - Grosser Administrationsaufwand
- Uneinheitliche HA-Fähigkeit
  - Hohe Angreifbarkeit
  - Veraltete Betriebssystem- und Datenbankversionen
  - Grosser Administrationsaufwand

Dadurch ergeben sich nach BSI folgende Risiken:

# Diplomarbeit

Identifikation			Beschreibung / Ursache	Auswirkung	Abschätzung	Behandlung		
ID	Schlüssel	Fehlerz.	Risiko		WS	SM	Massnahmen eingehen?	
1	I	Gu.22	Manipulation von Informationen	Zurück veraltete Systeme die zudem unterschiedlich gut gehandelt und bearbeitet das Risiko das Daten manipulieren werden. Manche Datenbanken und deren Benutzersysteme sind sehr alt und sehr lange im Einsatz. Einige dieser Systeme sind so alt, dass keine Hotfixes, Patches und Upgrades mehr erhältlich sind. Hierdurch entsteht das Risiko, das System Fehlfunktionen erleidet.	Die Auswirkungen reichen von einer Fehlfunktion des Systems bis hin zum vollständigen Verlust der integrität der Daten.	2	4	Ja
2	A	Gu.25	Ausfall von Geräten oder Systemen	Allerdings verursachen Datenbanksysteme die Auswirkungen so gering wie möglich zu halten.	Sofern keine HA-Architektur aufgebaut wurde, ist die Verfügbarkeit ernsthaft gefährdet resp. die Applikation steht nicht mehr zur Verfügung.	4	4	Ja
3	C, I, A	Gu.26	Fehlfunktion von Geräten oder Systemen	Manche Datenbanken nur durch Benutzersysteme sind sehr alt und sehr lange im Einsatz. Einige dieser Systeme sind schon so alt, dass keine Hotfixes, Patches und Upgrades mehr erhältlich sind. Hierdurch entsteht das Risiko, das System Ausfallen.	Fehlfunktionen können innerhalb von Datenbanksystemen die Datenbasis schädigen, Daten können verloren gehen oder ungewollt von Daten und unveränderten Personen eingespielt werden.	2	4	Ja
4	C, I, A	Gu.27-1	Ressourcenmangel (personelle Ressourcen)	Administrationsaufgaben die ledigen Systeme sehr gross. Einige der sehr heterogenen Landeskraft ist der Aufwand der regelmässigen Datenbank und deren Betriebssystem sehr groß. Es kann sein, dass für eine bedarfsgerechte Administration eindruckt werden kann.	Die Auswirkungen können variablen sein, abhängig davon welcher Aspekt des Ressourcenmangels ledet.	3	3	Ja
5	A	Gu.27-2	Ressourcenmangel (technische Ressourcen)	Kann auftreten wenn Ressourcenverbrauch zu spät bemerkt wird. So kann die CPU Usage oder das Memory Usage schwer anzuwaschen. Bei einem akuten Problem ist es nicht garantiert, dass die Leute erreichbar sind die notwendig sind.	Grundsätzlich wird aber sowohl die Verfügbarkeit, Integrität und Verfügbarkeit gefährdet.	2	2	Ja
6	C, I, A	Gu.31	Fehlerhafte Nutzung oder Administration von Geräten und Systemen	Durch die Verlall an Datenbankversionen und Betriebssystemen und Plattformen wird auf diese unzureichende Administration und Konfiguration bestellt allen voran das Risiko einer fehlerhaften Administration und Konfiguration.	Wenn die CPU- und Memory-Usage über einen gewissen Schwelwert geht, fängt die Betriebsystem an zu Priorisieren. Dies wird primär der Endnutzer in Form von Performance Einbußen betreffen.	2	2	Ja
7	C, I, A	Gu.32	Missbrauch von Berechtigungen	Doch die folgen bleiben nicht seltsam übersehbar. Abhängig davon, welche Fehler gemacht wurden können die Auswirkungen auch stark variieren. Sie reichen von fehlender Verschlüsselung bis hin zu nicht vorhandenem Backup mit nicht mehr gesicherter Wiederherstellbarkeit von Systemen.	Im schlimmsten Fall steht eine Anwendung nicht mehr zur Verfügung.	4	3	Ja
8	A, I	Gu.45	Datenverlust	Gefährlicher sind Storage Overflows, besonders wenn die Datenbank nicht mehr alle Informationen schreiben konnte, die sie für einen korrekten Nutzaufl benötigte.	Daraus erschließt sich das auch bei diesem Risiko die Vermutlichkeit, dass die folgen bleiben nicht seltsam übersehbar.	2	4	Ja
				Als dem Risiko ergeben sich zwei Auswirkungen, die aber beide ein hohes Mass an Schaden verursachen können. Unter anderem können Daten missbräuchlich abgezogen werden, Daten manipuliert oder das ganze System komplett zerstört werden.	Der Wissensstand oder Universitätsstand der Missbrauch von Berechtigungen kann weiterführende Auswirkungen haben.	4	5	Ja
				Wiederherstellung der Datenbanken ist in manchen Fällen nicht möglich.	Erstens können Backups gar nicht mehr Wiederhergestellt werden, dies hätte dann einen Totalen Datenverlust zur Folge. Die zweit Ursache resultiert aus der fehlenden WAL-Archivierung, dadurch können zwar die Daten bis zu einem Zeitpunkt X wiederhergestellt werden allerdings sind diese dann nicht zwangsläufig konsistent.	4	5	Ja

Tabelle 1.4: Risiko-Matrix aktuelle Situation PostgreSQL Datenbanken

Daraus ergeben sich folgende Strategien und Handlungsfelder um die Massnahmen zur Risikominimierung umzusetzen:

- Systemabsicherung erarbeiten und einsetzen
- HA-Clustering einführen um die Redundanz zu gewährleisten und Systeme zentral verwalten und betreiben zu können
- Lifecycle-management für Datenbanken und Betriebssysteme erarbeiten und einsetzen
- Backupkonzept erarbeiten
- Berechtigungskonzept erarbeiten und einführen

Mit diesen Massnahmen lassen sich die Risiken senken.

Die Risiken werden wie folgt gesenkt:

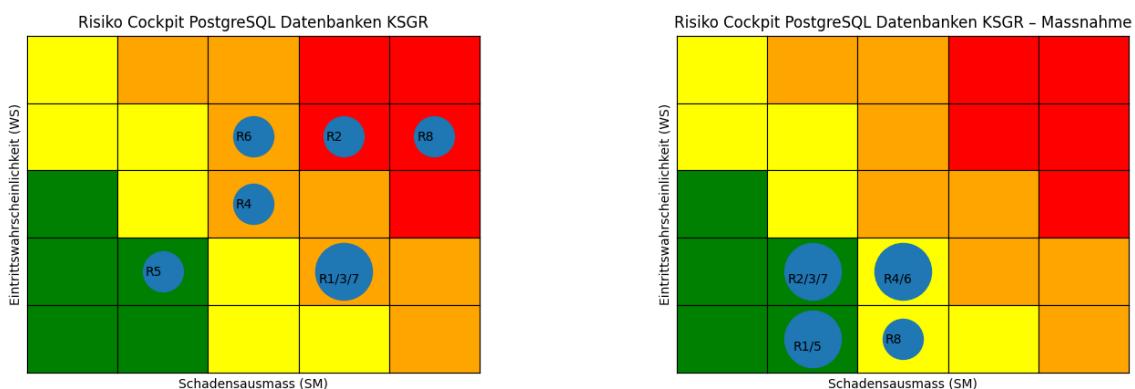


Abbildung 1.8: Risikomanagement PostgreSQL

## 1.2 Zieldefinition

Das administrieren einer PostgreSQL Datenbank umfasst i.d.R. [63, 69] folgende zehn Tasks die zum täglichen Alltag gehören:

Nr.	Aufgabe	Beschreibung	Wichtigkeit
1	Failover	In einem Fehlerfall soll die DB-Node auf einen Standby-Node übergeben werden. Nach einem Failover muss der DB-Node wieder vom Standby-Node auf den Primären Node zurückgesetzt werden.	Hoch
2	Failover Restore	Dabei darf es zu keinem Datenverlust kommen, also alle Daten die auf dem Standby-Node erfasst wurden, müssen auf den Primären DB-Node zurückgeschrieben werden beim Failover Restore Die Datenmenge von Datenbanken wachsen in der Regel beständig.	Hoch
3	Filesystem Management	Die Belegung von Tablespace und Filesystem muss deshalb Überwacht und ggf. erweitert werden. Läuft eine Disk voll kommt es im besten Fall zu einem Stillstand der DB, im schlimmsten Fall zu Inkonsistenzen und Datenverlust	Hoch
4	Monitoring	Nebst den allgemeinen Metriken wie CPU / Memory Usage und der Port Verfügbarkeit gibt es noch eine Reihe weiterer Aspekte die Überwacht werden müssen. Zum Beispiel ob es zu Verzögerungen bei der Replikation kommt oder die Tablespace genügend Platz haben. Dazu gehört auch das Überwachen des Logs und entsprechende Schritte im Fehlerfall. PostgreSQL sammelt Statistiken um SQL Queries optimaler ausführen zu können.	Mittel
5	Statistiken / Cleanup Jobs justieren	Zudem wird im Rahmen des gleichen Scheduled Tasks ein Cleanup Vorgenommen, so dass z.B. gelöschte Datensätze den Disk Space nicht sinnlos belegen. Die Konfiguration dieser Jobs muss an der Metrik der Datenbank angepasst werden, weil gewisse Tasks dann entweder viel zu oft oder viel zu wenig bis gar nicht mehr ausgeführt werden.	Mittel
6	SQL optimierungen	In PostgreSQL können unperfekte SQL Statements ausgelesen werden und zum Teil werden auch Informationen zum Tuning geliefert[42]. Diese müssen regelmäßig ausgelesen werden	Tief
7	Health Checks und Aktionen (Maintenance)	Regelmäßig muss die Gesundheit der DBs überprüft werden, etwa ob Tabellen und/oder Indizes sich aufgeblättert haben oder ob Locks vorhanden sind[2]. Während der Hauptarbeitszeit muss dies mindestens alle 90 Minuten geprüft und ggf. reagiert werden.	Hoch
8	Housekeeping	Mit Housekeeping Jobs werden regelmäßig Trace- und Alertlogfiles aufgeräumt, um Platz auf den Disken zu sparen aber auch um die Übersichtlichkeit zu wahren.	Mittel
9	Verwalten von DB Objekten	Regelmäßig müssen DB Objekte wie Datenbanken, Tabellen, Trigger, Views etc. angepasst oder erstellt werden. Dies richtet sich nach den Bedürfnissen der Kunden resp. deren Applikationen.	Tief
10	User Management	Die Zugriffe der User müssen überwacht, angepasst, erfasst oder gesperrt werden. Auch diese Aufgabe richtet sich nach den Bedürfnissen der Kunden.	Tief

Tabelle 1.5: Administrative Aufgaben

Von diesen Tasks müssen Teile davon zu 50% automatisiert werden wobei alle Muss-Aufgaben automatisiert werden müssen. Diese wären nachfolgende Tasks die automatisiert werden können.

Nr.	Aufgabe	Wichtigkeit	Zu automatisierender Task	Priorität	Muss / Kann	Spätester Termin
1	Failover	Hoch	Automatisierter Failover auf mindestens einen Sekundären DB-Node	1	Muss	Abgabe
2	Failover Restore	Hoch	Sobald der Primäre DB-Node wieder vorhanden ist, muss automatisch auf den Primären DB-Node zurückgesetzt werden. Das Filesystem muss beim erreichen von 95% Usage automatisiert vergrössert werden.	1	Muss	
3	Filesystem Management	Hoch	Die Vergrösserung muss anhand der Wachstumsrate (die mittels Linux Commands zu ermitteln ist), vergrössert werden	4	Kann	
4	Monitoring	Mittel	Der Status der Clusterumgebung und der Replikation muss im PRTG überwacht werden	2	Muss	
5	Statistiken / Cleanup Jobs justieren	Mittel	Regelmässig müssen die Parameter für den AUTOVACUUM Job berechnet werden und das Configfile postgresql.conf automatisch angepasst werden Es gibt SQL Abfragen, mit dem fehlende Indizes ermittelt werden können. Diese Indizes sollen automatisiert erstellt werden.	2	Muss	
6	SQL optimierungen	Tief	Im gleichen Zug sollen aber auch Indizes, welche nicht verwendet werden, entfernt werden. Sie tragen nicht nur nichts zu performanteren Abfragen bei sondern beziehen unnötige Ressourcen bei Datenmanipulationen[42]. Tabellen und Indizes können sich aufblähen (bloated table / bloated index)	2	Kann	
7	Health Checks und Aktionen (Maintenance)	Hoch	Ist ein Index aufgebläht, kann dies mittels eines REINDEX mit geringem Impact auf die Datenbank gelöst werden[2].	2	Muss	
8	Housekeeping	Mittel	Log Rotation muss aktiviert werden und alte Logs regelmässig gelöscht werden.	3	Kann	
9	Verwalten von DB Objekten	Tief	Keine automatisierung möglich	5		
10	User Management	Tief	Regelmässige Reports sollen User aufzeigen, die seit mehr als einer Woche nicht mehr aktiv waren.	4	Kann	

Tabelle 1.6: Automatisierung Administrativer Aufgaben

Mit der Arbeit sollen folgende Ergebnisse und Resultate erzielt werden:

- Ergebnisse  
Mindestens drei Methoden einen PostgreSQL Cluster aufzubauen müssen analysiert und evaluiert werden
- Resultate  
Aus den mindestens drei Methoden muss die optimale Methode ermittelt werden.  
Am Ende muss zudem ein Funktionierendes Testsystem bestehen.

Daraus ergeben sich folgende Ziele:

Nr.	Ziel	Beschreibung	Priorität
1	Evaluation	Am Ende der Evaluationsphase müssen mindestens drei Methoden für einen PostgreSQL HA Cluster müssen evaluiert werden. Innerhalb der evaluation muss analysiert werden, welche Methode oder welches Tool sich hierfür eignen würde.	Hoch
2	Testsystem	Am Ende der Diplomarbeit muss ein funktionierendes Testsystem installiert sein.	Hoch
3	Automatisierter Failover	Ein PostgreSQL Cluster muss im Fehlerfall auf mindestens einen Standby-Node umschwenken. Dabei muss das Timeout so niedrig sein, dass Applikationen nicht auf ein Timeout laufen.	Hoch
4	Automatisierter Failover Restore	Nach einem Failover muss es zu einem Fallback oder Failover Restore kommen, sobald der Primary-Node wieder verfügbar ist.	Hoch
5	Monitoring - Cluster Healthcheck	Die wichtigsten Parameter für das Monitoring des PostgreSQL Clusters (isready, Locks, bloated Tables), der Replikation (Replay Lag, Standby alive) und des PostgreSQL HA Clusters müssen überwacht werden.	Mittel
6	AUTOVACUUM - Parameter verwalten	Täglich müssen die Parameter für den AUTOVACUUM Job berechnet werden und das Configfile postgresql.conf automatisch angepasst werden	Mittel
7	SQL optimierungen - Indizes tracken und verwalten	Täglich fehlende Indizes automatisiert erstellen und nicht mehr verwendete Indizes automatisiert entfernen	Mittel
8	Maintenance - Indizes säubern	Täglich bloated Indices, also aufgeblähte Indizes, automatisiert erkennen und mittels REINDEX bereinigen	Hoch
9	Housekeeping - Log Rotation	Die Log Rotation muss aktiviert werden. Die Logs müssen aber auch in das KSGR-Log Repository geschrieben werden	Hoch
10	User Management - Monitoring	Nicht verwendete User sollen einmal pro Woche automatisiert erkannt und in einem Report gemeldet werden.	Tief
11	Evaluationsziel	Am Ende der Evaluationsphase muss ein Entscheid getroffen worden sein, welche Methode verwendet wird.	Hoch
12	Installationsziel	Die Testinstallation muss lauffähig sein und zudem alle Anforderungen und Ziele (3 und 4) erfüllen  Folgende Testziele müssen erreicht werden: 1. Der PostgreSQL Cluster muss immer lauffähig sein solange noch ein Node up ist, unabhängig davon welche Nodes des PostgreSQL HA Clusters down ist 2. Ein Switchover auf alle Secondary Nodes muss möglich sein 3. Der Fallback auf den Primary Node muss erfolgreich sein, unabhängig davon ob ein Failover oder Switchover stattgefunden hat 4. Das Timeout bei einem Failover / Switchover muss unterhalb der Default Timeouts der Applikationen GitLab und Harbor liegen. 5. Das Replay Lag zwischen Primary und Secondary darf beim Initialen Start nicht über eine Minute dauern oder 1KiB nicht überschreiten	Hoch
13	Testziele		

Tabelle 1.7: Ziele

### 1.3 Abgrenzungen

Im Kantonsspital Graubünden sind bereits einige Systeme im Einsatz, die gegeben sind.

	<b>Produkt</b>	<b>Beschreibung</b>
Storage	HPE 3PAR 8450 SAN Storage System	
Virtualisierungsplattform	VMware® vSphere®	
Primäres Backupsystem	VEEAM Backup System	
Provisioning / lifecycle management system	Foreman	Ist zurzeit nur für Linux angedacht
Primäre Linux Distribution	Debian	
	Rocky Linux	
Sekundäre Linux Distributionen	Oracle Linux	RedHat Enterprise Linux (RedHat Enterprise Linux (RHEL)), Rocky Linux oder Oracle Linux wird nur eingesetzt, wenn es nicht anders möglich ist
	RedHat Enterprise Linux (RedHat Enterprise Linux (RHEL))	
Primäres Monitoring System	Paessler Router Traffic Grapher (PRTG)	Monitoring System für alle ausser dem Netzwerkbereich
Sekundäres Monitoring System	Zabbix	Wird nur vom Netzwerkbereich verwendet
Container-Plattform	Kubernetes	
Infrastructure as code (IaC) System	Ansible und Terraform	Ansible wird von Foreman verwendet, Terraform wird für die Steuerung der Kubernetes-Plattform verwendet
Logplattform / SIEM System		Wird neu Ausgeschrieben.
Usermanagement	Microsoft Active Directory	Produkt zurzeit nicht definiert

Tabelle 1.8: Gegebene Systeme

Daraus ergeben sich nach nach Züst, Troxler 2002[93] folgende Abgrenzungen:

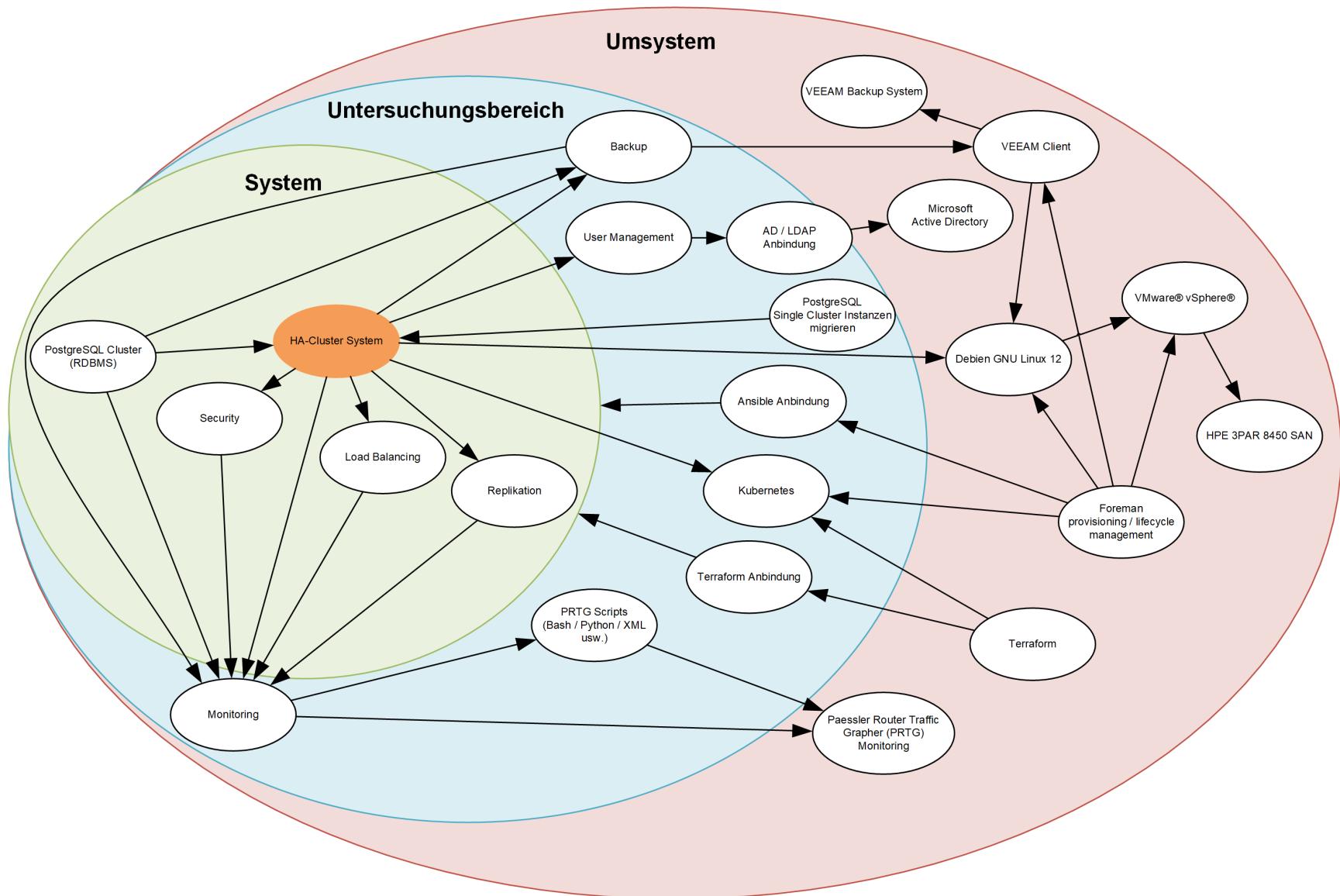


Abbildung 1.9: Systemabgrenzung

## 1.4 Abhängigkeiten

Es existieren Technische und Organisatorische Abhängigkeiten. Diese haben sowohl ein Risiko als auch einen Impact wenn das Risiko eintrifft. Dies wären folgende:

Nr.	Objekt	Abhängigkeit	Beschreibung	Status	Risiko	Impact
1	Foreman	VMs	Das Lifecycle Management und Provisioning System muss zur Verfügung stehen um in der Evaluationsphase Develop-VMs und in der Installationsphase Test-VMs erstellen zu können.	Im Moment ist Foreman in einer Proof of Concept Phase.	Das Risiko besteht, dass Foreman nicht betriebsbereit ist	VMs müssen von Hand aufgesetzt werden. Entsprechend wird sehr viel mehr Zeit in der Evaluations- und Installationsphase benötigt.
2	Storage	Speicher für VMs / Daten	Es müssen genügend Kapazitäten auf dem Storage vorhanden sein, um die VMs und Datenbanken in Betrieb zu nehmen.	Storage wurde bereits erweitert, neue Disks für den SAN Storage wurden bestellt.	Auf dem SAN ist keine Kapazität mehr vorhanden	Es können keine VMs oder Datenbanken erstellt werden. Log Retention muss stark erhöht werden. Dies wird mehr Storage in Anspruch nehmen.
3	Log Management / SIEM System	Sichern der Logfiles für Log Rotation	Ein Log Management System / SIEM muss vorhanden sein, um Logs langfristig sichern zu können.	Die neue Log Management Plattform ist noch nicht betriebsbereit	Die neue Log Management Plattform ist noch nicht betriebsbereit	Projekt kann nicht Zeitgemäß abgeschlossen werden
4	HP-UX Ablöseprojekt	Ressourcen	Das Projekt zur Ablösung der HP-UX Plattform für die Oracle Datenbanken geht in die Konzeptions- und Umsetzungsphase.	Umsetzungsphase.	Als Oracle DBA bin ich stark in das Projekt eingebunden. Es besteht das Risiko eines Ressourcenengpasses	Keine Versionierung und Teile Sicherungen mehr von Konfigurationsfiles, Scripts usw.
5	GitLab	Sicherung	Sicherung von Konfigurationen, Scripts usw.	GitLab ist implementiert und betriebsbereit.	GitLab steht nicht mehr zur Verfügung	Zertifikate können aus Zeitgründen nicht in der Evaluationsphase eingesetzt werden. Für die Testphase müssen Zertifikate manuell ausgestellt werden.
6	PKI	Key Management	Es braucht einen PKI um Keys und Zertifikate handeln zu können	Bestehender PKI wird abgelöst. Ablösungsprojekt in der Initialisierungsphase. Bestehender PKI nicht für Zertifikate im Einsatz	Es steht kein moderner PKI im Einsatz.	
7	Veeam Kasten K10[3]	Backup	Kubernetes Nodes und Pods können nicht mit Klassivem Veeam gesichert werden. Hierfür hat Veeam mit der Version V12 die spezialisierte Veeam Kasten K10 Lösung heraus.	Kann erst beim offiziellen Release von Kubernetes beim KSGR eingeführt werden.	Stellt nicht zur Verfügung, bis das Projekt abgeschlossen wird.	Backup muss z.B. einen nfs-Share gesichert werden.

Tabelle 1.9: Abhängigkeiten



**2**

## Projektmanagement

## 2.1 Risikomanagement

Aus den Abhängigkeiten heraus wurden folgende Risiken identifiziert:

Identifikation			Abschätzung		Behandlung			
ID	Risiko	Beschreibung / Ursache	WS	SM	Massnahmen ergreifen?	Zielwert WS	Zielwert SM	Massnahme
1	Fehlende Ressourcen	Viele parallele Projekte, Aufträge und der Tagesbetrieb	3	4	Ja	2	2	Organisation und Selbstmanagement
2	HP-UX Ablöseprojekt	Das Projekt ist sehr umfangreich und ist in die Konzeptions- und Umsetzungsphase gestartet	4	4	Ja	3	3	Ressourcen reservieren
3	Alte Infrastruktur kann ungeplant sämtliche Ressourcen binden	HP-UX Plattform, DELL NetWorker / Data Domain Umgebung und HPE 3PAR SAN Storage Umgebung sind über dem Lifecycle und haben in den vergangenen Monaten immer wieder kritische Ausfälle erlebt	4	4	Ja	3	3	Monitoring vorgängig ausbauen und Massnahmen definieren
4	Selbstmanagement und in der Selbstorganisation	Selbstmanagement und Organisation ist nicht meine Stärke	3	3	Ja	2	2	Werkzeuge im Vorfeld definieren und bereitstellen
5	Scope Verlust während des Projekts	Der Scope kann während des Projekts verloren gehen	3	4	Ja	2	3	Ziele klar definieren
6	Scope Creep	Der Umfang kann stark steigen wenn Ziele nicht genau genug definiert wurden	3	4	Ja	3	3	Ziele SMART definieren
7	SIEM / Log Plattform nicht betriebsbereit	Die öffentliche Ausschreibung für die neue / Log Plattform wurde erst am 23.10.2023 veröffentlicht. Bis zur Implementation kann noch Zeit vergehen.	4	1	Nein			
8	Foreman nicht betriebsbereit	Die Foreman Provisioning- und Lifecycle Plattform befindet sich aktuell erst in der Proof of Concept Phase. Dadurch besteht das Risiko, dass sie nicht betriebsbereit zum Start der Diplomarbeit ist	3	5	Ja	3	4	Massnahmen ergreifen um die manuelle Installation so effizient wie möglich zu gestalten.

Tabelle 2.1: Risiko-Matrix der Diplomarbeit

Daraus ergeben sich, mit den Massnahmen, folgende Risikomatrizen:

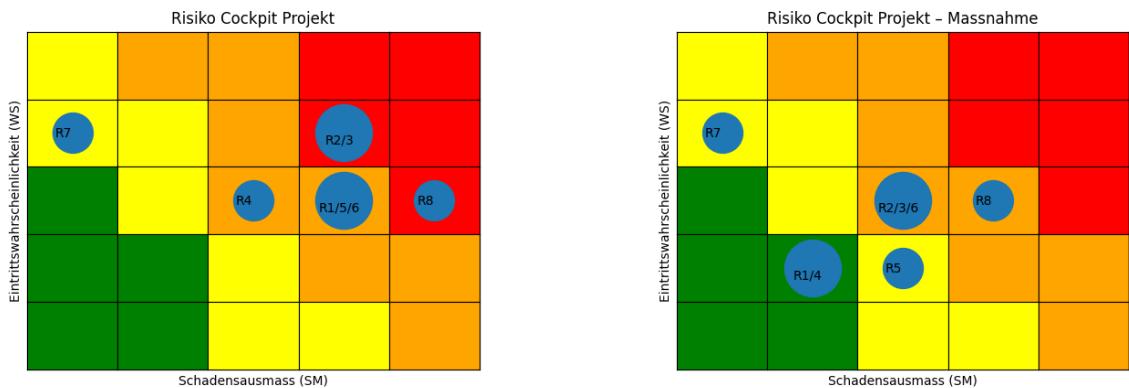


Abbildung 2.1: Risikomanagement Projekt

## 2.1.1 Riskcontrolling

### 2.1.1.1 Neu erfasste Risiken

ID	Definiert / Erkannt	Risiko	Beschreibung / Ursache	Auswirkung	WS	SM	Massnahmen notwendig	WS.1	SM.1	Massnahmen
9	19.03.2024 Keine Kubernetes-Gerechten Sicherungen von Pods usw. Ohne Kubernetes kein Veeam Kasten. Sicherungen inkonsistent o.ä. Ohne Backup kann das Ziel des Projekts nicht erreicht werden	Veeam Kasten K10[3] nicht betriebsbereit	Abhängigkeit zum KSGR k8s Projekt.							

Tabelle 2.2: Neu Erkannte / Erfasste Risiken



### 2.1.1.2 Assessment 21.03.2024

ID	Risiko	Assessment-Datum	WS	SM	Status	Egriffene Massnahmen	Wirksamkeit	Begründung
1	Fehlende Ressourcen	21.03.2024	3	4	hoch	Dokumentation ausserhalb Arbeitszeit	begrenzt	Mentale Ressourcen setzen Limits. ExaCC Server werden mitten während der Diplomarbeit geliefert.
2	HP-UX Ablöseprojekt	21.03.2024	5	4	sehr hoch	Ressourcen reserviert	begrenzt	Von KSGR Seite fehlt eine Stellvertretung. Mithilfe notwendig.
3	Alte Infrastruktur kann ungeplant sämtliche Ressourcen binden Schwächen beim	21.03.2024	4	4	hoch	Externe Partner sensibilisiert	wirksam	Externe Partner können meinen Teil der Aufgaben bei Problemen abfedern. Allerdings nicht vollständig
4	Selbstmanagement und in der Selbstorganisation	21.03.2024	3	3	hoch	- Projektplanung erstellt. - Arbeitspakete geplant	begrenzt	Nicht an alle Tasks gedacht, wie z.B. Risikocontrolling.
5	Scope verlust während des Projekts	21.03.2024	2	2	mittelmässig	Ziele SMART definiert	wirksam	Ziele sind klar definiert. Allerdings gibt es zwangsläufig gewisse Unschärfen.
6	Scope Creep	21.03.2024	3	3	hoch	Ziele SMART definiert	begrenzt	Sehr viele mögliche Lösungen am Markt. SIEM wird nicht rechtzeitig stehen.
7	SIEM / Log Plattform nicht betriebsbereit	21.03.2024	5	1	sehr hoch	keine		Das Schadensmass ist aber zu gering, damit Massnahmen ergriffen werden müssten.
8	Foreman nicht betriebsbereit	21.03.2024	1	1	erledigt	keine		Foreman ist in Betrieb
9	Veeam Kasten K10[3] nicht betriebsbereit	21.03.2024	5	5	sehr hoch	noch keine		

Tabelle 2.3: Risiko-Assessment 21.03.2024

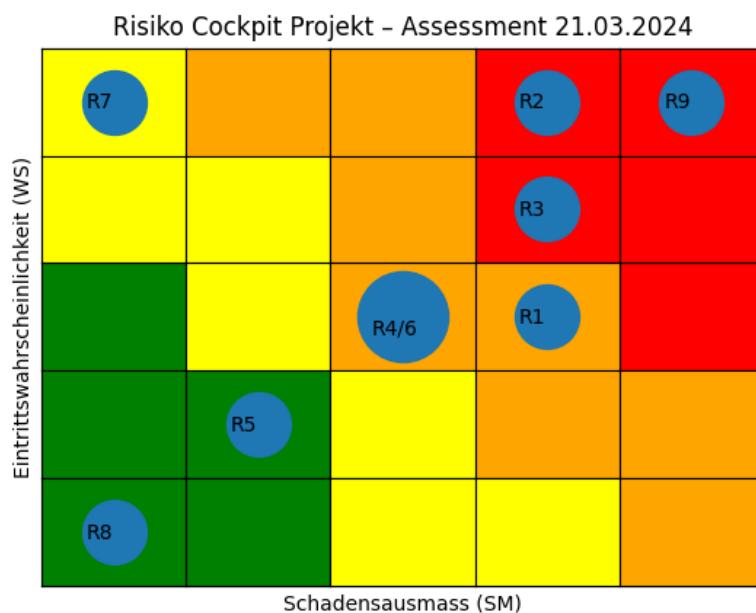


Abbildung 2.2: Riskikomatrix - Assessment 21.03.2024

## 2.2 Vorgehensweise und Methoden

## 2.3 Projektplanung

### 2.3.1 Projektcontrolling

	Phase	Subphase	Dauer [h]	Geplante Dauer [h]	Verbleibende Zeit [h]
0	1. Expertengespräch	1. Expertengespräch	1.0	1.0	0.0
1	2. Expertengespräch	2. Expertengespräch	0.2	1.0	0.8
2	Aufbau und Implementation Testsystem	Basisinfrastruktur	0.0	4.0	4.0
3	Aufbau und Implementation Testsystem	Installation und Konfiguration PostgreSQL HA Cluster	0.0	20.0	20.0
4	Aufbau und Implementation Testsystem	Technical Review	0.0	3.0	3.0
5	Dokumentation	Dokumentation	31.0	80.0	49.0
6	Evaluation	Analyse PostgreSQL HA Cluster Lösungen	17.8	16.0	-1.8
7	Evaluation	Anorderungskatalog	4.5	16.0	11.5
8	Evaluation	Gegenüberstellung	0.0	8.0	8.0
9	Evaluation	Variantenentscheid	0.0	4.0	4.0
10	Evaluation	Vorbereitung Benchmarking	5.0	4.0	-1.0
11	Letztes Expertengespräch	Letztes Expertengespräch	0.0	1.0	1.0
12	Puffer	Puffer	0.0	16.0	16.0
13	Resultate	Persönliches Fazit	0.0	2.0	2.0
14	Resultate	Schlussfolgerung	0.0	2.0	2.0
15	Resultate	Weiteres Vorgehen / offene Arbeiten	0.0	1.0	1.0
16	Resultate	Zielüberprüfung	0.0	2.0	2.0
17	Testing	Protokollierung	0.0	4.0	4.0
18	Testing	Review und Auswertung	0.0	2.0	2.0
19	Testing	Testing Testsystem	0.0	8.0	8.0
20	Troubleshooting und Lösungsfindung	Troubleshooting und Lösungsfindung	0.0	8.0	8.0
	Total		59.5	203.0	143.5

Tabelle 2.4: Projektcontrolling

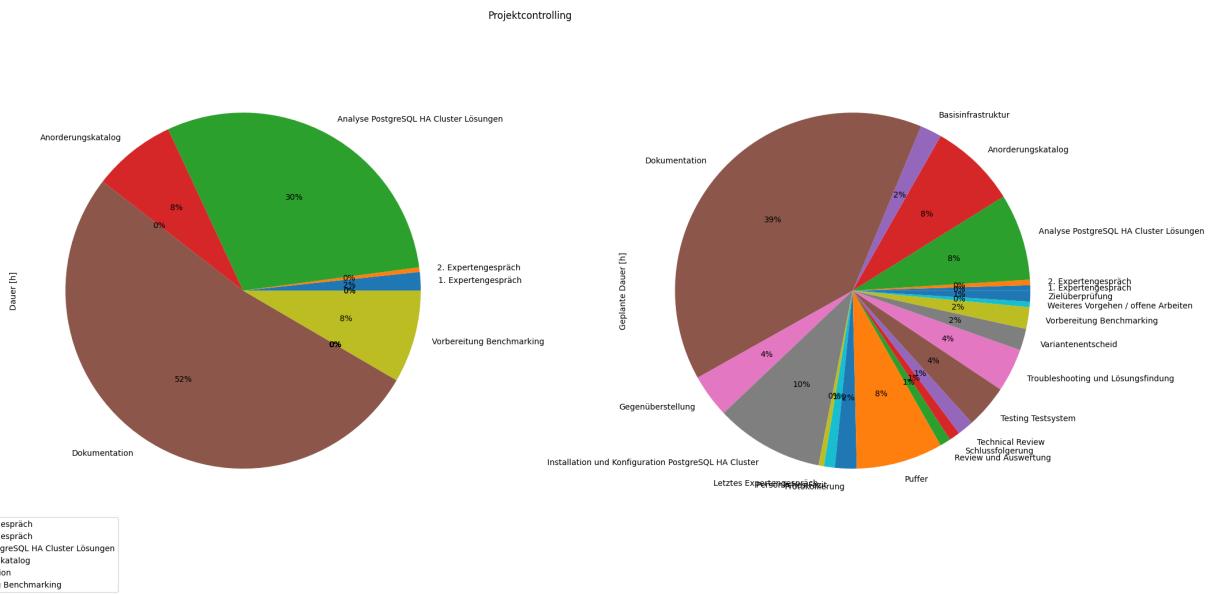
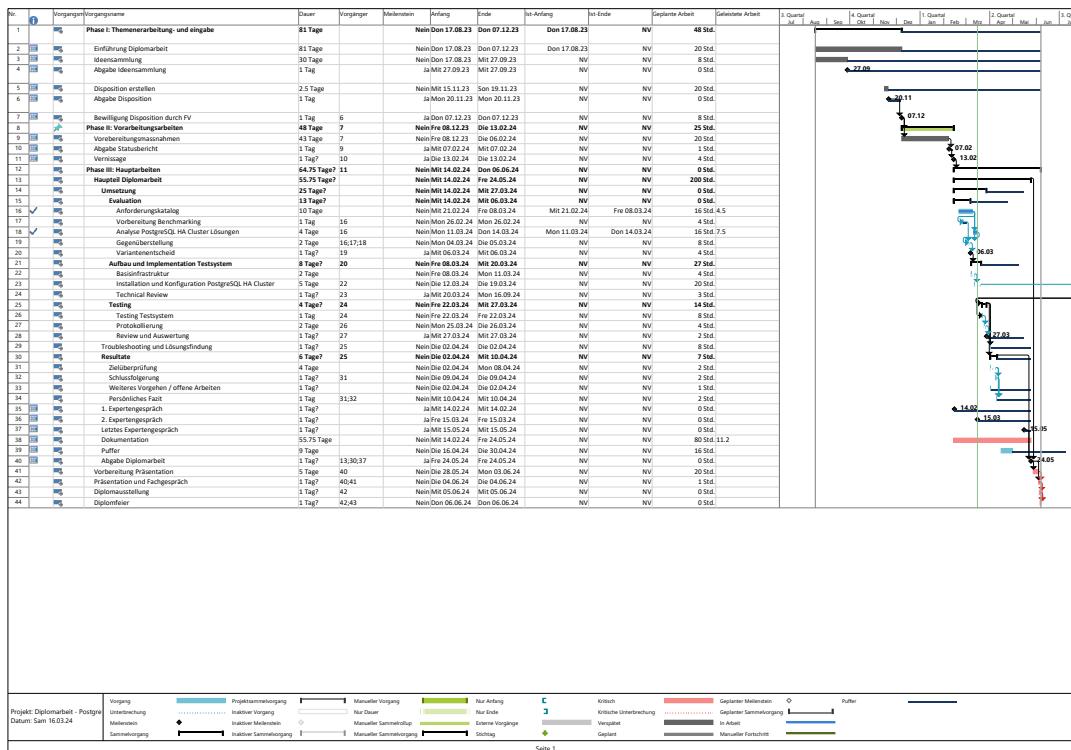


Abbildung 2.3: Projektcontrolling

### 2.3.2

#### GANTT-Diagramm



## 2.4 Expertengespräche

Folgende Expertengespräche fanden statt:

Fachgespräch	Datum	Fachexperte	Nebenexperte	Studenten	Bemerkungen
1	14.02.2024	Norman Süsstrunk	-	Michael Graber Curdin Roffler	- Es wurden zwar für alle Studenten von Norman Süsstrunk Zoom-Räume bereitgestellt, aus effizienzgründen nahmen Curdin Roffler und ich beide am selben Meeting teil
2	26.03.2024	Norman Süsstrunk	-	Michael Graber	

Tabelle 2.5: Fachgespräche

Das Protokoll ist im Anhang zu finden.

### 3 Umsetzung

#### 3.1 Evaluation

##### 3.1.1 Exkurs Architektur

###### 3.1.1.1 Sharding

###### 3.1.1.1.1 Vertikales / Horizontales Sharding

Tabellen können Horizontal oder Vertikal partitioniert werden.

Primary Key	Column 1	Column 2	Column 3
1	A	B	C
2	D	E	F
3	G	H	I
4	J	K	L
5	M	N	O
6	P	Q	R

Komplette Tabelle

Primary Key	Column 3
1	C
2	F
3	I
4	L
5	O
6	R

Primary Key	Column 1	Column 2
1	A	B
2	D	E
3	G	H
4	J	K
5	M	N
6	P	Q

Vertikale Partitionen

Abbildung 3.1: Sharding - Vertikale Partitionierung

## Diplomarbeit

Primary Key	Column 1	Column 2	Column 3
1	A	B	C
2	D	E	F
3	G	H	I
4	J	K	L
5	M	N	O
6	P	Q	R

Primary Key	Column 1	Column 2	Column 3
1	A	B	C
6	P	Q	R

Primary Key	Column 1	Column 2	Column 3
2	D	E	F
3	G	H	I

Primary Key	Column 1	Column 2	Column 3
4	J	K	L
5	M	N	O

Primary Key	Column 1	Column 2	Column 3
6	P	Q	R
7	S	T	U

Abbildung 3.2: Sharding - Horizontales Partitionierung

Horizontales Partitionieren wird meistens für das Sharding von Tabellen benutzt. Die Partitionen entsprechen dann den Shards.

### **3.1.1.1.2 Key Based Sharding**

Hierbei wird das sharding anhand eines oder mehreren Keys ausgeführt.

### **3.1.1.1.3 Range Based Sharding**

Das Sharding wird dabei anhand von Ranges ausgeführt. Zum Beispiel anhand von Preis-Ranges.

### **3.1.1.1.4 Directory Based Sharding**

Hierfür wird eine lookup-Tabelle geführt, welche die Schlüssel für das Sharding bereitstellen. Anhand dieser werden dann die entsprechenden Zieltabellen aufgeteilt.

### **3.1.1.1.5 Hash Based Sharding**

Das Hash Based Sharding ist eine Form des Range Based Shardings, bei dem Hashwerte der Datensätze benutzt werden. Je nach Bereich wird der Datensatz dann einem Shard zugewiesen.

## **3.1.1.2 Monolithische vs. verteilte SQL Systeme**

Klassische SQL-Datenbanken sind Monolithische Systeme, selbst wenn sie mittels Replikation eine Primary/Standby-Architektur aufweisen. Man kann mittels eines SQL Proxys ein gewisses Mass an Load Balancing betreiben, hat aber immer noch das Problem das es einen Primary Node gibt auf dem beschrieben wird. Monolithische Systeme sind daher nicht Cloud Native.

Nur verteilte Systeme, sogenannte Distributed SQL wiederum sind Cloud Native

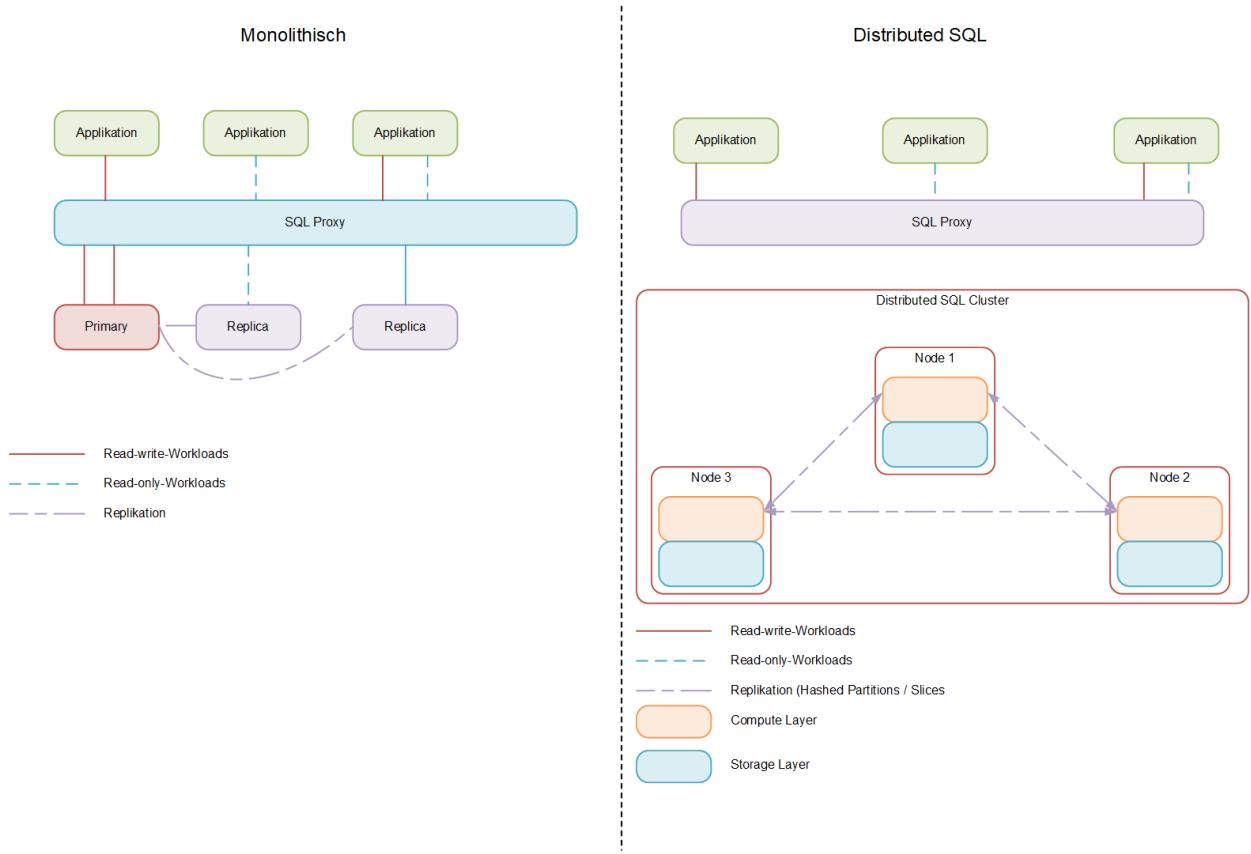


Abbildung 3.3: Monolithische vs. verteilte SQL Systeme

### 3.1.1.3 High Availability und Replikation

Wenn eine Datenbank HA (High Availability), also Hochverfügbar, sein soll, braucht es eine Primäre und mindestens eine Sekundäre- oder Failover-Datenbank. Um Datenverlust zu vermeiden, müssen die Daten permanent von der Primären auf die sekundäre Datenbank repliziert werden, dies nennt man Replikation[67]. Dabei wird zwischen den folgenden beiden Replikationen unterschieden:

#### Synchrone Replikation

Wenn bei einer Synchroen Replikation eine Transaktion abgesetzt wird, wird der Commit auf der primären Seite erst gesetzt, wenn die Änderung auf der sekundären Seite oder den sekundären Seiten ebenfalls eingetragen und Committed ist. Bis zu diesem Moment ist die Transaktion nicht als Committed.

Dies wird dann zum Problem, wenn keine Verbindung mehr zu mindesten einer sekundären Seite vorhanden ist. Zudem wird die Synchrone Replikation bei hohen Latenzen zum Bottleneck der Datenbank.

#### Asynchrone Replikation

Bei der Asynchronen Replikation wird eine Transaktion erst auf der eigenen primären Seite Committed und erst dann an die sekundären Nodes gesendet. Besonders bei hohen Latenzen bleibt die Datenbank immer perfomant, allerdings kann es je nach Latenz und genereller Auslastung zu Datenverlusten kommen, wenn es zum Failover kommt.

### 3.1.1.4 Quorum

Ein Quorum-System soll die Integrität und Konsistenz in einem Datenbank-Cluster sicherstellen. Dabei gilt zu beachten, dass nicht eine beliebige Anzahl an Nodes hinzugefügt werden können. Auch hat das Hinzufügen von Nodes immer eine einbusse an Performance zur Folge, besonders dann, wenn eine Synchrone Replikation gewählt wird und auf jedes Commitmend von den Replica-Nodes gewartet werden muss.

#### Quorum

Die Mehrheit der Server, die einen funktionierenden Betrieb gewährleisten können, ohne eine Split-brain-Situation zu erzeugen. Die Formel ist gemeinhin  $n/2 + 1$

#### Throughput

Beschreibt, wie sich die Anzahl Nodes auf die Schreibgeschwindigkeit der Commitments auf die restlichen Nodes auswirkt.

Die Verdopplung der Server halbiert i.d.R. den Throughput.

#### Fehlertoleranz

Beschreibt, wie viele Nodes ausfallen können, damit der Cluster noch arbeitsfähig ist.

Wobei eine Erhöhung der Nodes von 3 auf 4 die Fehlertoleranz nicht erhöht da nun eine Split-brain-Situation entstehen kann.

Hier ein Beispiel wie sie in den Artikeln [65, 76, 60] beschrieben werden. Es zeigt auf, ab wie vielen Nodes die Fehlertoleranz erhöht wird und wie sich der Representative Throughput verhält.

Anzahl Nodes	Quorum	Fehlertoleranz	Representative Throughput
1	1	0	100
2	2	0	85
3	2	1	82
4	3	1	57
5	3	2	48
6	4	2	41
7	4	3	36

Tabelle 3.1: Quorum Beispiele

### 3.1.1.5 CAP Theorem

Das CAP Theorem besagt, dass nur zwei der drei folgenden drei Merkmale von verteilten Systemen gewährleistet werden können[49].

#### Konsistenz - Consistency

Die Datenbank ist konsistent, alle Clients sehen gleichzeitig die gleichen Daten unabhängig von welchem Node zugegriffen wird. Hierzu muss eine Replikation der Daten an alle Nodes stattfinden und der Commit zurückgegeben werden, also eine synchrone Replikation stattfinden.

#### Verfügbarkeit - Availability

Jeder Client, der eine Anfrage sendet, muss auch eine Antwort erhalten. Unabhängig davon wie viele Nodes im Cluster noch aktiv sind.

#### Ausfalltoleranz / Partitionstoleranz - Partition tolerance

Der Cluster muss auch dann noch funktionsfähig bleiben, wenn es eine beliebige Anzahl von Verbindungsunterbrüchen oder anderen Netzwerkproblemen zwischen den Nodes gibt.

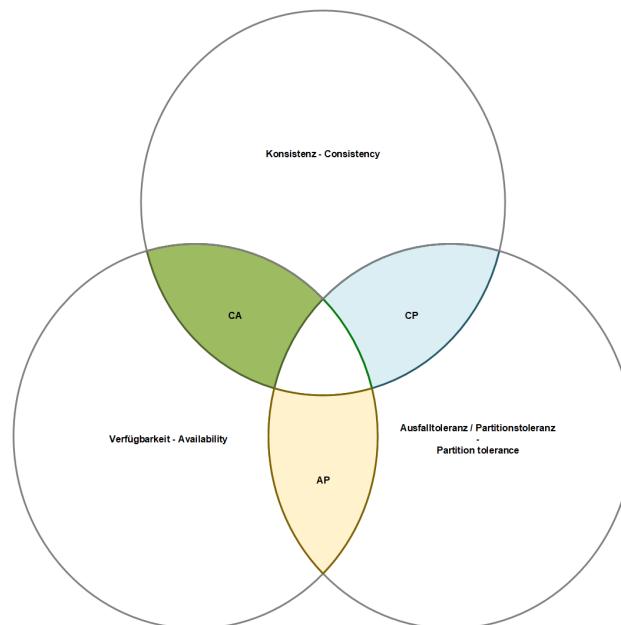


Abbildung 3.4: CAP-Theorem

PostgreSQL, Oracle Database oder IBM DB2 präferieren CA, also Konsistenz und Verfügbarkeit.

### 3.1.1.6 Skalierung

Datenbanken müssen skalierbar sein. Dabei wird unterschieden zwischen einer vertikalen Skalierung (scale-up) und horizontaler Skalierung (scale-out). Bei der vertikalen Skalierung

## Diplomarbeit

werden den DB-Servern mehr CPU-Cores und Memory sowie zum Teil Storage hinzugefügt, wobei der Storage in jedem Fall wachsen wird. Beim horizontalen Skalieren werden weitere DB-Nodes in den Cluster eingehängt[62]:

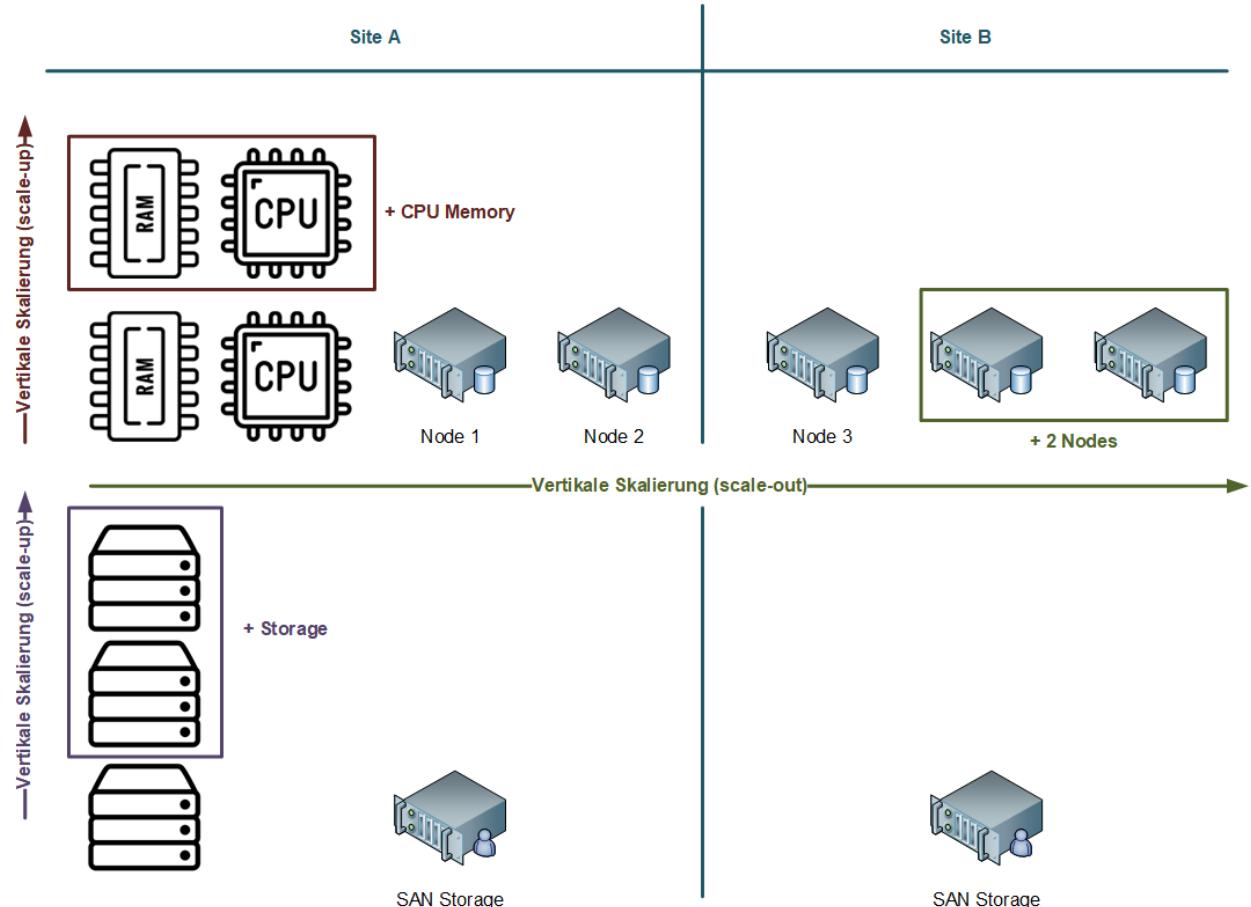


Abbildung 3.5: Datenbank skalierung

Bei monolithischen Datenbanken, werden irgendwann die Grenzen der horizontalen Skalierung erreicht und man muss wieder vertikal Skalieren, um dem Primary Node genügend Rechnerleistung vorzuhalten.

### 3.1.2 Erheben und Gewichten der Anforderungen

#### 3.1.2.1 Anforderungen

Das KSGR hat eine Cloud First Strategie.

Das heißt, alle neuen Applikationen und entsprechend deren Datenbanken müssen Cloud Ready bzw. Cloud Native sein. Um die Voraussetzung dafür zu schaffen, muss auch der PostgreSQL Cluster Cloud Ready sein.

Daher müssen zwei von drei genauer evaluierten Lösungen Cloud Native Lösungen sein. Wenn

## Diplomarbeit

der Zeitaufwand reicht, können auch eine Cloud Native und Monolithisches System aufgebaut werden.



Nr.	Anforderung	Bezeichnung	Beschreibung	System	Muss / Kann
1	Systemvielfalt		Es muss mindestens eine Monolithisches und mindestens 2 zwei Distributed SQL Cluster ermittelt werden	Beides	MUSS
2	Synergien		Skripte und APIs des Monolithisches Systems müssen auch in einem Distributed SQL System verwendet werden können	Beides	MUSS
3	Failover	Automatismus	Das Clustersystem muss bei einem Nodeausfall automatisch auf einen anderen Node umstellt	Beides	MUSS
4	Failover	Connection - Stabilität	Beim Failover dürfen bestehende Connections nicht getrennt werden oder sofort Wiederhergestellt werden	Beides	MUSS
5	Failover	Geschwindigkeit	Das umstellen auf den nächsten Node muss so schnell ausgeführt werden, das ein Disconnect mittels Client-Konfiguration (Timeout) verhindert wird.	Beides	MUSS
6	Switchover	Skript / API	Das System muss ein Skript oder eine API liefern, welche einen geordneten Switchover auf einen anderen Node erlaubt	Beides	MUSS
7	Switchover	Connection - Stabilität	Beim Switchover dürfen bestehende Connections nicht getrennt werden oder sofort Wiederhergestellt werden	Beides	MUSS
8	Switchover	Geschwindigkeit	Das umstellen auf den nächsten Node muss so schnell ausgeführt werden, das ein Disconnect mittels Client-Konfiguration (Timeout) verhindert wird.	Beides	MUSS
9	Restore	Skript / API	Das Clustersystem muss ein Skript oder eine API liefern, welche das einfache und ggf. automatisierte Restore eines oder mehreren Nodes ermöglichen	Beides	MUSS
10	Restore	Datensicherheit	Beim Wiederherstellen des Ursprungszustands darf es zu keinem Datenverlust kommen	Beides	MUSS
11	Restore	Connection - Stabilität	Bei der Wiederherstellung einzelner Nodes darf es zu keinen Unterbrechungen auf den Applikationen kommen	Beides	MUSS
12	Restore	Geschwindigkeit	Das Wiederherstellen des Ursprungszustands muss innert weniger Stunden für alle Datenbanken aus dem Backup Wiederhergestellt und im Clustersystem Synchronisiert werden	Beides	MUSS
13	Replikation	Synchrone Replikation	Es muss eine Synchrone Replikation sichergestellt werden	Monolithisch	MUSS
14	Replikation	Failover / Switchover Garantie	Die Replikation muss sicherstellen, das es bei einem Failover/Switchover zu keinem Fehler kommt	Monolithisch	MUSS
15	Replikation	Throughput	Beschreibt, wie viele Transaktionen pro Zeiteinheit vom Primary an die Replikas gesendet und Committed werden. Dieser Wert ist bei Synchrone Replikation entscheidend da Commits auf allen Replicas abgesetzt sein müssen.	Beides	MUSS
16	Sharding	Datenschutz- und integrität	Die Datenkonsistenz und Datenintegrität auf den Shards muss sichergestellt werden	Distributed SQL	MUSS
17	Sharding	Schutz vor Datenverlust	Die Synchronisation der Shards muss sicherstellen, dass es zu keinem Datenverlust kommt	Distributed SQL	MUSS
18	Quorum	Quorum-System vorhanden	Das Clustersystem muss über ein Quorum-System besitzen	Beides	MUSS
19	Quorum	Robustheit	Das Quorum des Clustersystems muss robust genug sein, um eine Split-Brain-Situation zu verhindern	Beides	MUSS
20	Connection		Das Clustersystem muss sicherstellen, dass eine Applikation ohne Entwicklungsaufwand mittels dem PostgreSQL Wired Connector zugreifen kann	Beides	MUSS
			Das Clustersystem muss Skripte oder eine API liefern, mit dem das System zu konfigurieren, verwalten oder überwachen zu können.	Beides	MUSS
21	Management-API	Management-API vorhanden	Zudem müssen mit geringen Arbeitsaufwand damit Nodes hinzugefügt oder entfernt werden können	Beides	MUSS
22	Management-API	Authentifizierung & Autorisierung	Es müssen gängige Standards für Authentifizierung und Autorisierung mitgebracht werden	Beides	MUSS
23	Management-API	Aufwand	Der Aufwand, der benötigt wird um die DB zu verwalten,	Beides	MUSS
			Nodes hinzuzufügen oder zu entfernen usw. muss gegeneinander verglichen werden.	Beides	MUSS
24	Backup	Backup mit PostgreSQL Standards	Backups müssen mittels PostgreSQL Standards angezogen werden	Beides	MUSS
25	Backup	Restore mit PostgreSQL Standards	Backups müssen mittels PostgreSQL Standards restored werden können	Beides	MUSS
26	Housekeeping - Log Rotation		Das Clustersystem muss die möglichkeit zur Log Rotation bieten	Beides	MUSS
27	Self Healing		Das Clustersystem muss im Fehlerfall Nodes selber wiederherstellen können	Beides	KANN
28	Monitoring - Node Failure		Läuft ein Node auf einen Fehler, muss das Clustersystem dies erkennen und Melden resp.	Beides	MUSS
			eine Schnittstelle liefern die abgefragt werden kann	Beides	MUSS
			Da die meisten PostgreSQL HA Lösungen Open-Source sind,	Beides	MUSS
			muss sichergestellt werden,	Beides	MUSS
			das die gewählte Lösung auch aktiv gepflegt wird.	Beides	MUSS
			Als Basis dienen hier Informationen wie z.B. GitHub Insights.	Beides	MUSS
30	Performance	tps - Read-Only	Die Transaktionsrate (transactions per second / tps) für DQL Transaktionen	Beides	MUSS
31	Performance	tps - Read-Writes	Die Transaktionsrate (transactions per second / tps) für DML Transaktionen	Beides	MUSS
32	Performance	Ø Latenz - Read-Only	Die Latenzzeit bei DQL Transaktionen	Beides	MUSS
33	Performance	Ø Latenz - Read-Write	Die Latenzzeit bei DML Transaktionen	Beides	MUSS

Tabelle 3.2: Anforderungskatalog

## Diplomarbeit

### 3.1.2.2 Stakeholder

Rolle	Funktion	Departement	Bereich	Abteilung
Zabbix Stakeholder	Abteilungsleiter	D10 ICT	Infrastrukturmanagement	ICT Netzwerk, Security und Comm.
Stakeholder Data Center Infrastruktur	Abteilungsleiter	D10 ICT	Infrastrukturmanagement	ICT Data Center
k8s Stakeholder	ICT System Ingenieur	D10 ICT	Infrastrukturmanagement	ICT Data Center

Tabelle 3.3: Stakeholder

### 3.1.2.3 Gewichtung

Die Gewichtung wurde mittels einer Präferenzmatrix ermittelt.

Dabei wurden folgende Anforderungen aus übersichtsgründe in Sub-Matrizen aufgeteilt:

- Failover
- Switchover
- Restore
- Replikation
- Sharding
- Quorum
- Management-IP
- Backup
- Performance

Die Grundlegende Gewichtung wurde folgendermassen vorgenommen:

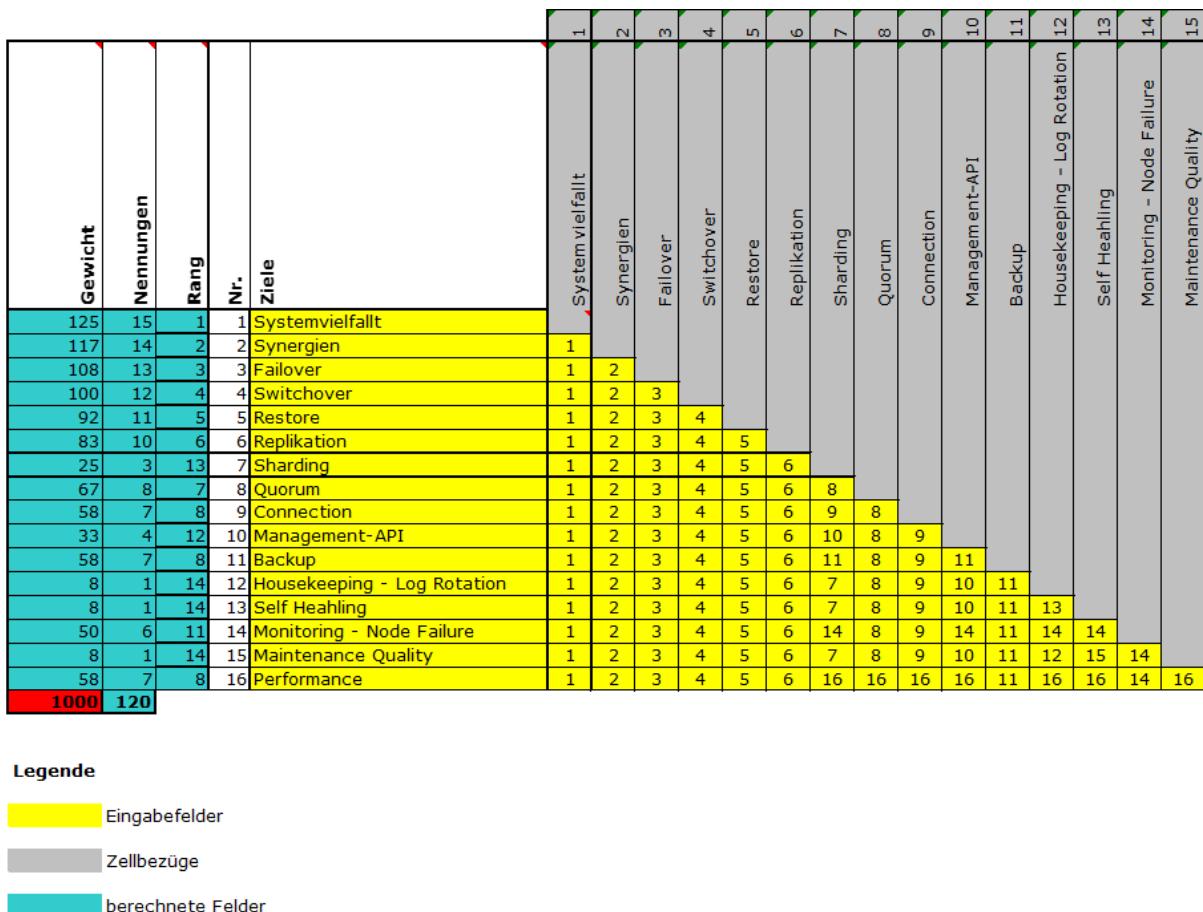


Abbildung 3.6: Präferenzmatrix

### 3.1.3 Testziele erarbeiten

#### 3.1.3.1 Evaluation - Testfälle

##### 3.1.3.1.1 Patroni

###### Failover

- Der Server des Primary-Node wird manuell heruntergefahren.
- Während dem Failover müssen Daten via SQL eingeführt und ausgelesen werden.
- Während dem Failover muss mindestens eine längere Abfrage gestartet werden.

###### Switchover

- Mit der REST-API wird der Switchover auf einen anderen Nod abgesetzt.
- Mit dem patronictl-Command wird der Switchover gesetzt

6. Während dem Switchover müssen Daten via SQL eingeführt und ausgelesen werden.
7. Während dem Switchover muss mindestens eine längere Abfrage gestartet werden.

### Restore

8. Mit der REST-API wird der Node erst mit dem `reinitialize` wiederhergestellt und dann mit einem Switchover wieder als Primary gesetzt.
9. Mit dem `patronictl`-Command und Parameter `reinit` der Node wiederhergestellt und abschliessend mittels Switchover wieder als Primary gesetzt.
10. Mit der REST-API wird der Node mit dem `reinitialize` wiederhergestellt
11. Mit dem `patronictl`-Command und Parameter `reinit` der Node wiederhergestellt
12. Vor, während und nach dem Restore müssen Tabellen mit Foreign-Key-Constraints und Daten geprüft werden.
13. Während dem Restore muss mindestens eine längere Abfrage gestartet werden und Daten via SQL eingeführt und ausgelesen werden.

#### 3.1.3.1.2 StackGres - Citus

### Failover

1. Der Server des Leader-Coordinator-Node wird manuell heruntergefahren.
2. Während dem Failover müssen Daten via SQL eingeführt und ausgelesen werden.
3. Während dem Failover muss mindestens eine längere Abfrage gestartet werden.

### Sharding

4. Vor, während und nach dem Failover müssen Tabellen mit Foreign-Key-Constraints geprüft werden.
5. Nach einem Failover-Test müssen alle Daten vorhanden sein.

### Self Healing

6. Der Node muss wieder hochgefahren werden.  
Der Node muss selbstständig Daten synchronisieren.
7. Der Leader muss automatisch neu gesetzt werden, wenn notwendig

### 3.1.3.1.3 YugabyteDB

#### Failover

1. Ein k8s Node wird manuell heruntergefahren, indem der entsprechende Server heruntergefahren wird.
2. Während dem Failover müssen Daten via SQL eingeführt und ausgelesen werden.
3. Während dem Failover muss mindestens eine längere Abfrage gestartet werden.

#### Sharding

4. Vor, während und nach dem Failover müssen Tabellen mit Foreign-Key-Constraints geprüft werden.
5. Nach einem Failover-Test müssen alle Daten vorhanden sein.

#### Self Healing

6. Der Node muss wieder hochgefahren werden.  
Der Node muss selbstständig Daten synchronisieren.

### 3.1.3.2 Evaluation - ERD self\_healing\_test

Die Tests müssen bei allen drei Varianten anhand der Datenbank `self_healing_test` durchgeführt werden.

Dabei werden die Tabellen, im Hinblick auf das Citus Schema Based Sharding, in Schemas organisiert.

Zwischen den einzelnen Schemas sollen einige Tabellen einen Foreign-Key auf andere Tabellen legen:

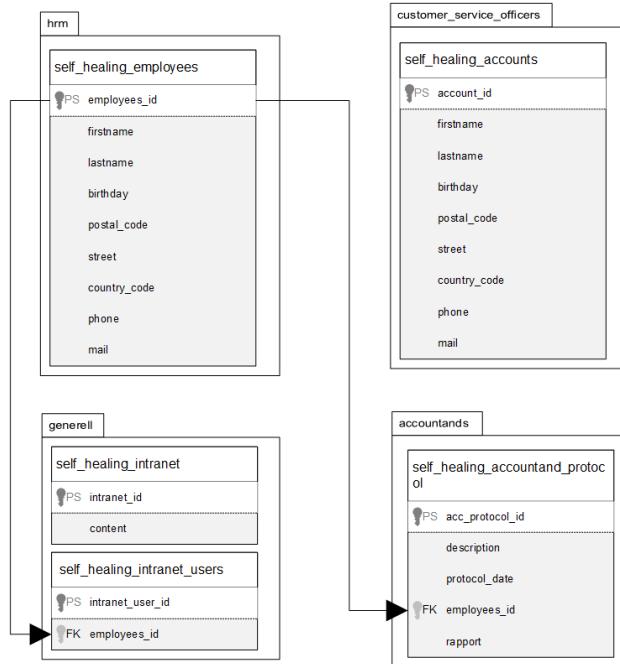


Abbildung 3.7: Testing - ERD DB self\_healing\_test

### 3.1.4 PostgreSQL Benchmarking

#### 3.1.4.1 pgBench - Basis-Benchmarking

PostgreSQL bietet ein Benchmarking-Tool,[56, 1] mit dem die DB vermessen werden kann.

Damit die Tests aussagekräftig sind, werden mit den Testläufen mehrere Läufe gestartet. Der erste Lauf muss dabei ignoriert werden, denn erst dann wird die DB in den Cache geladen. Wird dies nicht eingehalten, so wird die ganze Testreihe unbrauchbar.

Es gibt einiges zu beachten, wenn PostgreSQL einem Benchmarking unterzogen wird. Aus diversen Quellen [37, 34, 91, 1] sind dies folgende Anforderungen:

##### **pgGather**

Mit pgGather [64] müssen vorgängig alle Probleme analysiert und beseitigt werden.

##### **Maintenance**

Vor und nach dem Initialisieren des Benchmarks muss AUTOVACUUM gestartet werden.

##### **Statistiken bereinigen**

Um saubere Informationen mit pgGather sammeln zu können, müssen sie jeweils neu generiert werden.

##### **Non-Default Konfiguration**

Die PostgreSQL DB sollte nicht mit der Default Konfiguration betrieben werden.

Die Konfiguration sollte anhand der zu erwartenden Workloads und Sessions konfiguriert werden.

### Benchmark anpassen

Der Benchmark sollte sich an die zu erwartenden Anzahl Sessions und Anzahl Transaktionen richten.

### Benchmark Dauer

Die Zeit zwischen den Transaktionen und die Dauer an sich sollten über einen längeren Zeitraum stattfinden.

Nur so, kann ein echtes Verhalten gemessen werden.

### Störfaktoren

Störfaktoren, etwa Netzwerklatenzen [87] usw., müssen ebenfalls bemessen werden.

Nur so kann sichergestellt werden, dass die Umgebung sauber ist.

#### 3.1.4.2 Replication Throughput Benchmarking

Etwas Komplexer wird es beim Benchmark des Throughput. Den nebst den DB-Latenzen usw. kommt nun noch die Netzwerklatenz usw. hinzu [75].

#### 3.1.4.3 Benchmark Settings

Das Mass aller Dinge ist die Zabbix-DB.

Sie wird vorerst die grössten Zugriffszahlen, das höchste Datenwachstum und die meisten Transaktionen erzeugen.

Es werden alle Switches sowie der grösste Teil der Router erfasst, es sind im Moment etwas mehr als 32'000 Items erfasst.

Ein Item kann ein Gerät, ein Port oder mehrere States pro Port sein:

System information		
Parameter	Value	Details
Zabbix server is running	Yes	localhost:10051
Number of hosts (enabled/disabled)	250	240 / 10
Number of templates	345	
Number of items (enabled/disabled/not supported)	323479	318628 / 4796 / 55
Number of triggers (enabled/disabled [problem/ok])	136777	135529 / 1248 [148 / 135381]
Number of users (online)	24	3
Required server performance, new values per second	950.86	
High availability cluster	Disabled	

Abbildung 3.8: Benchmark Settings - Zabbix - Systeminformationen

## Diplomarbeit

Pro Sekunde werden ca. 950 Datenpunkte abgeholt.

Da der grossteil der Netzwerksysteme aber erfasst sind, wird die Anzahl Items nicht mehr stark anwachsen.

Auf die Datenbank wird sehr stark zugegriffen. Es werden bis zu 23 Connections pro Sekunde ausgeführt:

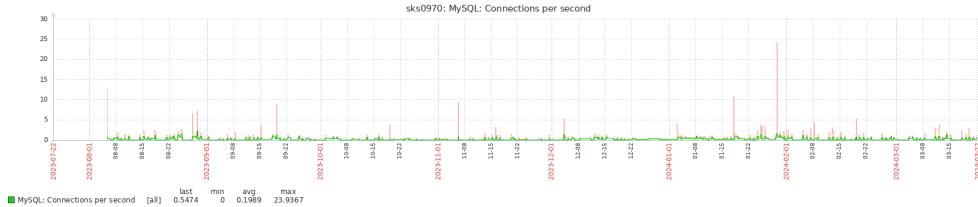


Abbildung 3.9: Benchmark Settings - Zabbix - Connections per Seconds

Pro Sekunde wurden bisher bis zu über 7'000 Queries ausgeführt. Dies schliesst Abfragen von Stored Programs ein:

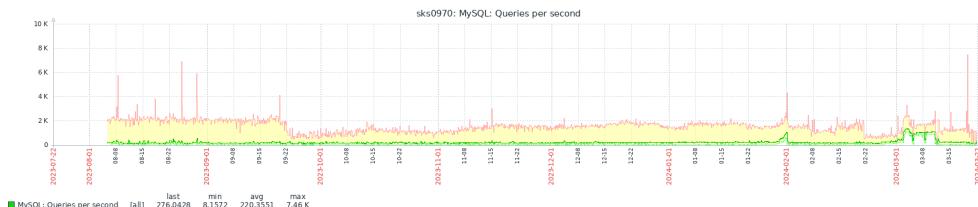


Abbildung 3.10: Benchmark Settings - Zabbix - Queries per Seconds

Reine Client anfragen waren nichtsdestotrotz über 4'000 Queries pro Sekunde:

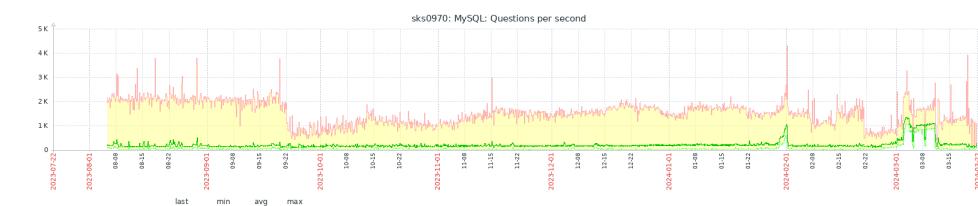


Abbildung 3.11: Benchmark Settings - Zabbix - Client Queries per Seconds

Auch das wachstum ist beachtlich. Die DB startete mit 180GiB und ist zurzeit bei rund 232GiB, war aber schon bei 238GiB:

## Diplomarbeit

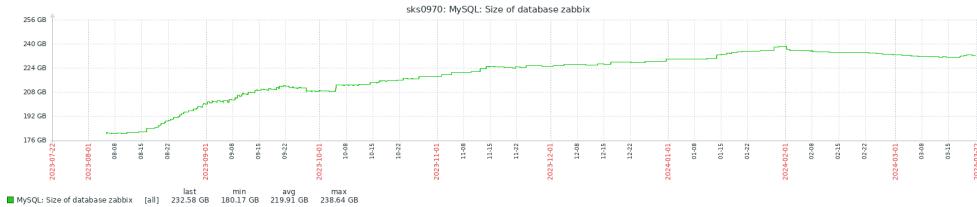


Abbildung 3.12: Benchmark Settings - Zabbix - DB Size

Nun kommen noch die restlichen, kleineren DBs hinzu. Heisst, für den Mixed Benchmark (DML und DQL Transaktionen) werden folgende Werte und Parameter gesetzt:

Typ	Parameter	pgbench-Parameter	1. Lauf	2. Lauf	3. Lauf	4. Lauf
mixed	Datenbank		pgbench_eval_bench	pgbench_eval_bench	pgbench_eval_bench	pgbench_eval_bench
mixed	DB-Grösse		5GiB	15GiB	50GiB	250GiB
mixed	1. Iteration Lauf ignorieren		ja	ja	ja	ja
mixed	Select only	-S	nein	nein	nein	nein
mixed	Iterationen pro Lauf		4	4	4	4
mixed	Vacuum	-v	ja	ja	ja	ja
mixed	Separate Connects	-C	ja	ja	ja	ja
mixed	Client count	-c	10	50	100	1000
mixed	Anzahl Transaktionen pro Client	-t	10	50	50	7
mixed	Anzahl Transaktionen Total		100	2500	5000	7000
mixed	Anzahl Worker Threads	-j	4	4	4	4

Tabelle 3.4: Benchmark Settings - Mixed Transaktionen

Für den DQL-Only Benchmark wird mit folgenden Konfigurationen gearbeitet:

Typ	Parameter	pgbench-Parameter	1. Lauf	2. Lauf	3. Lauf	4. Lauf
dql	Datenbank		pgbench_eval_bench	pgbench_eval_bench	pgbench_eval_bench	pgbench_eval_bench
dql	DB-Grösse		5GiB	15GiB	50GiB	250GiB
dql	1. Iteration Lauf ignorieren		ja	ja	ja	ja
dql	Select only	-S	ja	ja	ja	ja
dql	Iterationen pro Lauf		4	4	4	4
dql	Vacuum	-v	ja	ja	ja	ja
dql	Separate Connects	-C	ja	ja	ja	ja
dql	Client count	-c	10	50	100	1000
dql	Anzahl Transaktionen pro Client	-t	10	50	50	7
dql	Anzahl Worker Threads	-j	4	4	4	4

Tabelle 3.5: Benchmark Settings - DQL Transaktionen

Bei pgbench kann nicht einfach die größe angegeben werden.

Es muss der Skalierungsfaktor angepasst werden.

Dieser legt allerdings fest, wie viele Daten gespeichert werden.

Wird eine 1 eingeben, so werden 100'000 Records angelegt.

Um nun auf eine gewisse größe zu kommen, gibt es verschiedene Formeln.

Die als beste stellte sich folgende heraus[57]:

Zielobjekt	Skalierungsfaktor Formel
DB	$0.0669 * DB - Zielgrsse - 0.5$
Tabelle (pgbench_accounts / ysql_bench_accounts)	$0.0781 * Tabelle - Zielgrsse$
Index (pgbench_accounts_pkey / ysql_bench_accounts_pkey)	$0.4668 * Index - Zielgrsse$

Tabelle 3.6: Benchmark Settings - Skalierungsfaktoren

Daraus errechnen sich für die DB-Größen des Benchmark-Settings folgende Skalierungsfaktoren:

DB Grösse [GiB]	DB Grösse [MiB]	Scale Faktor
5	5120	342
15	15360	1027
50	51200	3425
250	256000	17126

Tabelle 3.7: Benchmark Settings - Datenbankgrößen / Skalierungsfaktor

yugabyteDB speichert die Daten anders als PostgreSQL, nämlich als Key-Value-Store. Das verhindert, dass die DB Grösse nicht ausgelesen werden kann, nur die Tabellen sind ersichtlich.

Um einen Vergleich zu haben, muss daher die Tabellengrösse berechnet werden.

Der Skalierungsfaktor für die Tabellen ist folgendermassen aufgebaut:

DB Grösse [GiB]	DB Grösse [MiB]	Scale Faktor
5	5120	400
15	15360	1200
50	51200	3999
250	256000	19994

Tabelle 3.8: Benchmark Settings - Tabellengrößen / Skalierungsfaktor

Der Skalierungsfaktor wird pro 100'000 gerechnet, gebe ich also den Faktor 1 ein, werden 100'000 Zeilen in der Tabelle pgbench\_accounts resp. ysql\_bench\_accounts erzeugt. Entsprechend wachsen die Anzahl Records wie folgt an:

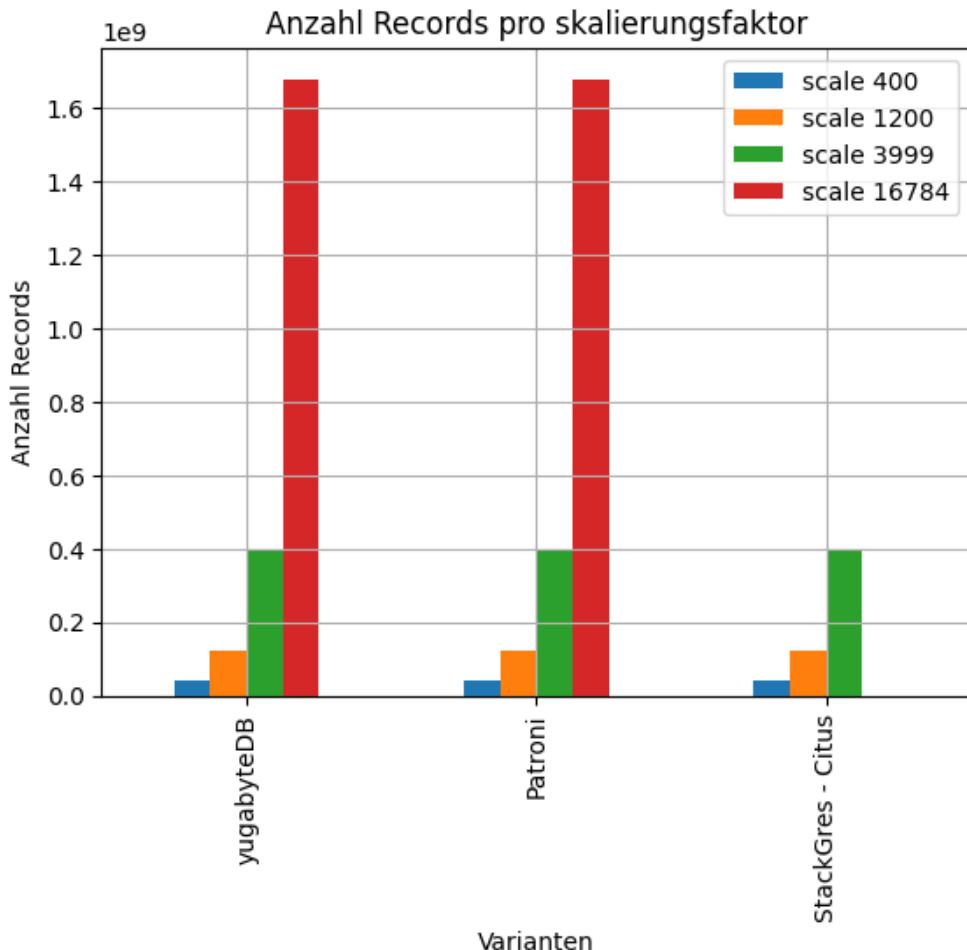


Abbildung 3.13: Benchmark Settings - Anzahl Records / Skalierungsfaktor

### 3.1.5 Analyse gängiger PostgreSQL HA Cluster Lösungen

#### 3.1.5.1 Percona

Percona bietet nebst dem bekannteren Galera-Cluster und MySQL- und MariaDB auch Dienstleistungen [33] aller Art für PostgreSQL an.

Percona arbeitet oft auf Basis von Patroni, bietet aber auch eigene Erweiterungen und eigene Software an[30].

Da Percona keine Open-Source-Lösungen bietet, sondern Service-orientiert ist, wird Percona nicht betrachtet.

## Diplomarbeit

Allerdings wird immer mal wieder auf frei zugängliche Dokumentationen von Percona verwiesen werden.

### 3.1.5.2 EnterpriseDB

EnterpriseDB, oder EDB, ist ein führendes Software- und Servicehaus für PostgreSQL[18]. Nebst Dienstleistungen für das Management von PostgreSQL-Umgebungen, Migrationen aus Oracle-Umgebungen und anderem, bietet EDB auch eine vielzahl an zusätzlicher Software und eigene Replikationslösungen.

EDB bietet PostgreSQL auf Kubernetes mittels CloudNativePG an, hat aber auch eine eigens entwickelte Distributed SQL / Active-Active Lösung an.

Da die EDB-Lösungen nicht Open-Source sind resp. von EDB aufgekauft Lösungen nicht mehr ohne Subscription betreibbar sind, werden sie nicht berücksichtigt. Allerdings wird immer mal wieder auf frei zugängliche Dokumentationen von EDB verwiesen werden.

### 3.1.5.3 KSGR Lösung

Das Kantonsspital Graubünden hat basierend auf keepalived wird geprüft ob die primäre Seite erreichbar und betriebsbereit ist. Trifft dies nicht mehr zu, wird ein Failover durchgeführt[77]. Ist die primäre Seite wieder verfügbar, wird ein Restore auf die primäre Seite gefahren.

Es wird beim Restore immer ein komplettes Backup der sekundären Seite auf die primäre Seite übertragen. Ursache ist, dass die normalerweise für den Datenrestore benötigten PostgreSQL Board mittel nur für eine relativ kurze Zeit eingesetzt werden können ehe die differenzen zwischen den beiden Seiten zu gross werden.

Bei kleinen Datenbanken wie jene für Harbor und GitLab ist die Zeit die hierfür benötigt wird, nicht relevant. Sind die Datenbanken auf dem PostgreSQL Cluster jedoch grösser, kann der Restore mehrere Minuten dauern.

### 3.1.5.4 pgpool-II

pgpool-II ist eine Middleware die zwischen einem PostgreSQL Cluster und einem PostgreSQL Client gesetzt wird.

#### 3.1.5.4.1 Core-Features

pgpool-II bietet folgende Core-Feature[54]:

- Watchdog für Automatischer Failover

## Diplomarbeit

- Eigener Quorum-Algorithmus
- Integrierter Pooler
- Eigenes Replikationssystem
- Integriertes Load Balancing
- Limiting Exceeding Connections, also queuen von Connections
- In Memory Query Caching
- Online Node Recovery / Resynchronisation

### 3.1.5.4.2 Replikation

pgpool-II bietet ein eigenes Replikationssystem an.

Es besteht allerdings die Möglichkeit, die PostgreSQL Standardreplikationen zu verwenden.

### 3.1.5.4.3 Proxy

pgpool-II hat bereits einen integrierten Load Balancer.

Einen zusätzlichen Proxy wird daher nicht benötigt.

### 3.1.5.4.4 Pooling

pgpool-II bietet ebenfalls von Haus aus einen Pooler.

### 3.1.5.4.5 API / Skripte

pgpool-II bietet mit pgpool ein eigenes Command- und Toolset an.

Mit dem CLI-Tool pcp\_command bietet pgpool-II zudem über eine abgesicherte API, die auch via Netzwerk erreichbar ist.

### 3.1.5.4.6 Maintenance

pgpool-II hat kein GitLab- oder GitHub-Repository. Metriken wie die GitHub Insights, welche Aufschluss über den Zustand des Projekts geben, finden sich nicht.

Die Dokumentation wird zwar aktuell gehalten, ist aber sehr minimalistisch gehalten. Sie bietet nur wenig Informationen zum Aufbau und Architektur von pgpool-II.

### 3.1.5.5 pg\_auto\_failover

pg\_auto\_failover ist eine PostgreSQL Erweiterung, die von der Microsoft Subunternehmen Citus Data entwickelt wird.

#### 3.1.5.5.1 Core-Features

Die wichtigsten Features von pg\_auto\_failover sind:

- API
- PostgreSQL Extension, also reines PostgreSQL
- State Machine Driven
- Replikations-Quorum
- Citus kompatibel
- Azure VM Support

#### 3.1.5.5.2 Replikation

pg\_auto\_failover bietet die Standard PostgreSQL Replikationen.

#### 3.1.5.5.3 Proxy

pg\_auto\_failover benötigt einen HAProxy, um Load Balancing usw. [22]

#### 3.1.5.5.4 API / Skripte

pg\_auto\_failover bietet ein eigenes CLI-Tool, pg\_autoctl. Dieses bietet Commands für das einbinden neuer Nodes, das Managen von Nodes (Maintenance resp. Switchover), konfigurieren oder monitoren des Systems bietet[26].

#### 3.1.5.5.5 Architektur

Die Dokumentation von pg\_auto\_failover [6] zeigt auf, wie der Failover funktioniert:

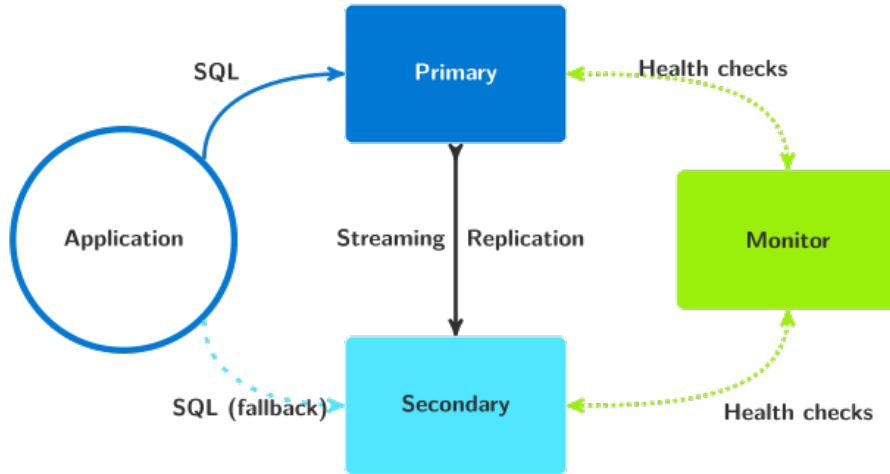


Abbildung 3.14: pg\_auto\_failover-Architektur - Single Standby

Aber auch Multi-Nodes können eingebunden werden[29]:

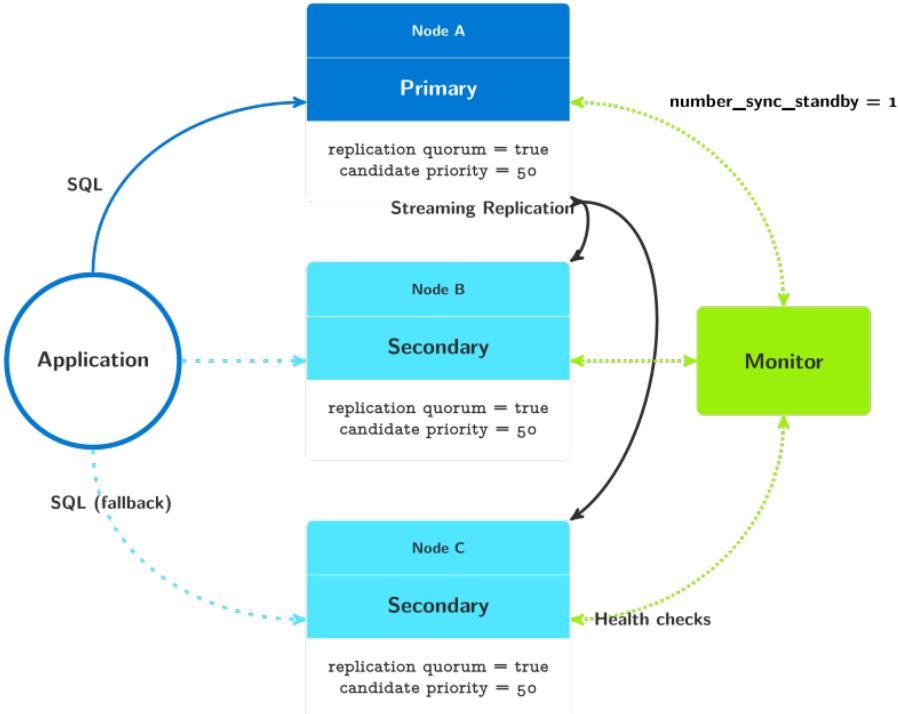


Abbildung 3.15: pg\_auto\_failover-Architektur - Multi-Node Standby

pg\_auto\_failover kann Citus einbinden[11]. Allerdings bleibt die Architektur im Kern immer Monolithisch.

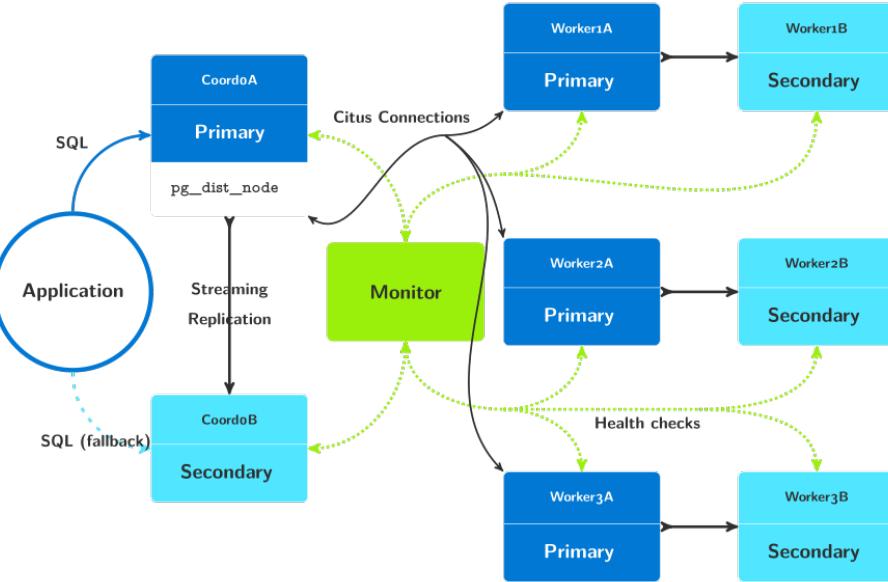


Abbildung 3.16: pg\_auto\_failover-Architektur - Citus

### 3.1.5.5.6 Synergien und Mehrwert

pg\_auto\_failover bietet eine Docker-Compose-Integration.  
Allerdings ist keine Kubernetes-Integration dokumentiert.

Damit bietet pg\_auto\_failover keine Möglichkeit,  
Synergien zwischen monolithischer Architektur und einer Cloud-Native-Umsetzung auf  
Kubernetes.  
Entsprechend ist kein Mehrwert vorhanden.

### 3.1.5.6 CloudNativePG

CloudNativePG ist eine Containerlösung für PostgreSQL auf Kubernetes.  
CloudNativePG wurde ursprünglich von EDB entwickelt.

#### 3.1.5.6.1 Core-Features

Die wichtigsten Features von CloudNativePG sind[13]:

- k8s API integration
- Autoamtischer Failover
- Self-Healing von Nodes resp. Replikas
- Skalierbarkeit (Vertikal, Horizontal bedingt)

- Volumne Backup
- Object Backup
- Rolling PostgreSQL Upgrade / Updates
- Pooling mit PgBouncer
- Offline und Online Import von bestehenden PostgreSQL DBs

### 3.1.5.6.2 Replikation

CloudNativePG bietet die üblichen PostgreSQL Replikaionen an.

### 3.1.5.6.3 Proxy

CloudNativePG benötigt keinen zusätzlichen Proxy.

### 3.1.5.6.4 Pooling

CloudNativePG unterstützt pgBouncer als Pooler.

### 3.1.5.6.5 API / Skripte

CloudNativePG bietet eine API zum Monitoren und Verwalten von Backups, Clustern und dem System selbst[4].

### 3.1.5.6.6 Architektur

Kubernetes regelt die Zugriffe mittels eines entsprechenden Services in die Nodes auf den Pods:

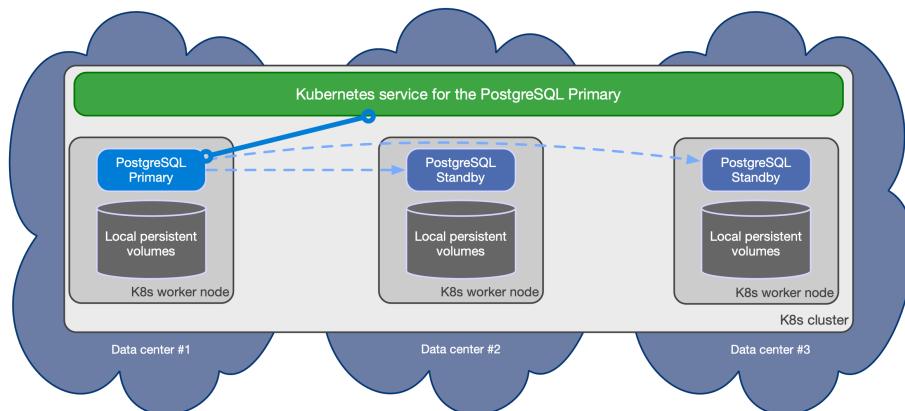


Abbildung 3.17: CloudNativePG - Kubernetes - PostgreSQL

Dabei werden die Read-write workloads an den Primary Node gesendet:

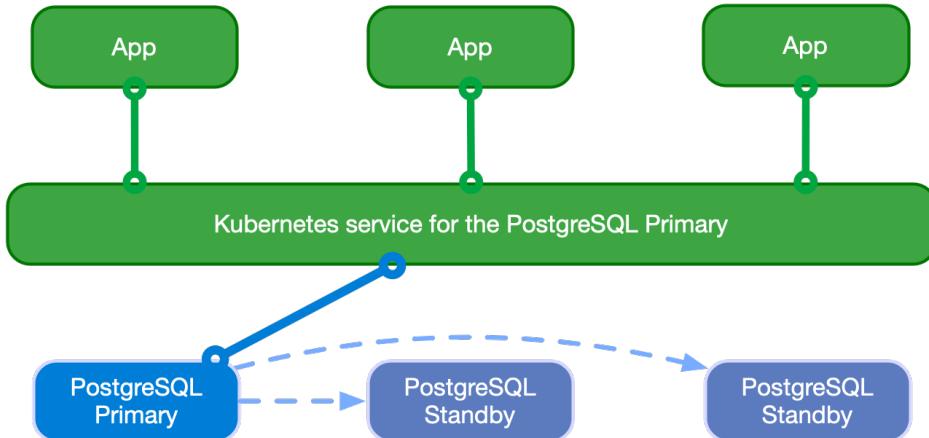


Abbildung 3.18: CloudNativePG - Kubernetes - Read-write workloads

Read-only workloads werden mit Round robin an die Replicas zugewiesen:

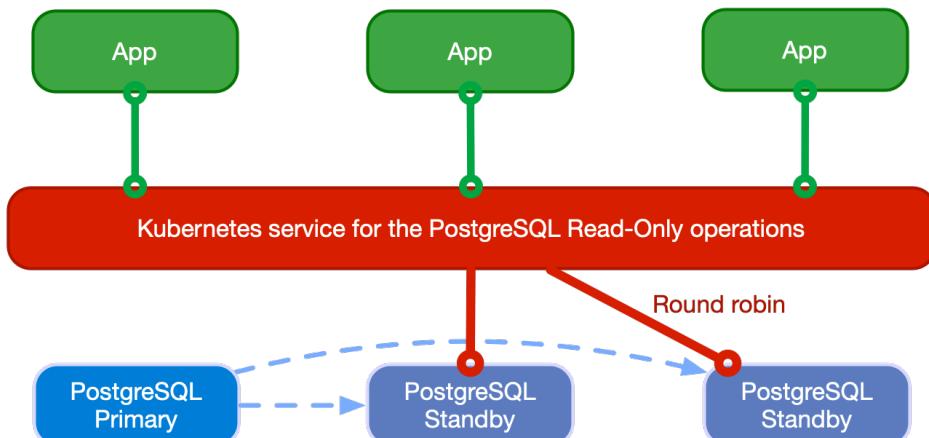


Abbildung 3.19: CloudNativePG - Kubernetes - Read-only workloads

Es könnten auch Lösungen mit Designated Kubernetes-Clustern in einem anderen RZ oder einer anderen Geo-Region realisiert werden.

### 3.1.5.6.7 Maintenance

Anhang - Maintenance

### 3.1.5.6.8 Synergien und Mehrwert

CloudNativePG bleibt ein Monolithisches System, welches aber keine Möglichkeit bietet, auch auf einem Normalen Serversetting betrieben zu werden.

Daher bietet CloudNativePG weder einen Benefit durch seine Architektur noch mit der Möglichkeit, Synergien nutzen zu können.

### 3.1.5.7 Patroni

Patroni ist eine von Zalando auf Python Basis entwickelte HA-Lösung für PostgreSQL. Patroni wird aktiv von Zalando gepflegt.

#### 3.1.5.7.1 Core-Features

- Rest-API und eigenes Skript- und Toolset
- Aktionen und Konfigurationen im Konsensprinzip abgestimmt
- Manueller oder Scheduled Switchover
- Reines PostgreSQL als Basis, Patroni setzt mittels Python darauf auf
- Automatische reintegration von Nodes nach einem Fehler
- Citus kompatibel
- Docker und Docker-compose Dokumentation

#### 3.1.5.7.2 Replikation

Patroni bietet per Default eine eigene Replikation an.  
Diese ist allerdings eine Asynchrone Replikation.

Es besteht allerdings die Möglichkeit, die Synchrone Replikation von PostgreSQL selbst einzuschalten.

#### 3.1.5.7.3 Proxy

Patroni benötigt einen HAProxy, um Load Balancing betreiben zu können[22].

#### 3.1.5.7.4 Pooling

Patroni benötigt einen externen Pooler.  
Hier wird oft PgBouncer [31] verwendet.

### 3.1.5.7.5 API / Skripte

Patroni hat ein eigenes Tool- und Commandset, `patronictl`, welches die Verwaltung vereinfacht. Es umfasst das ändern und erfassen von Konfigurationen, das forcieren eines Failovers als Switchover, Maintenance Handling und Informationsbeschaffung. Zusätzlich bietet Patroni eine API, welche Daten für das Monitoring bereitstellt aber auch Betriebsfunktionen bereitstellt.

### 3.1.5.7.6 etcd

Patroni benötigt etcd als key-value-store

### 3.1.5.7.7 Architektur

Das Architektur-Schaubild sieht folgendermassen aus:

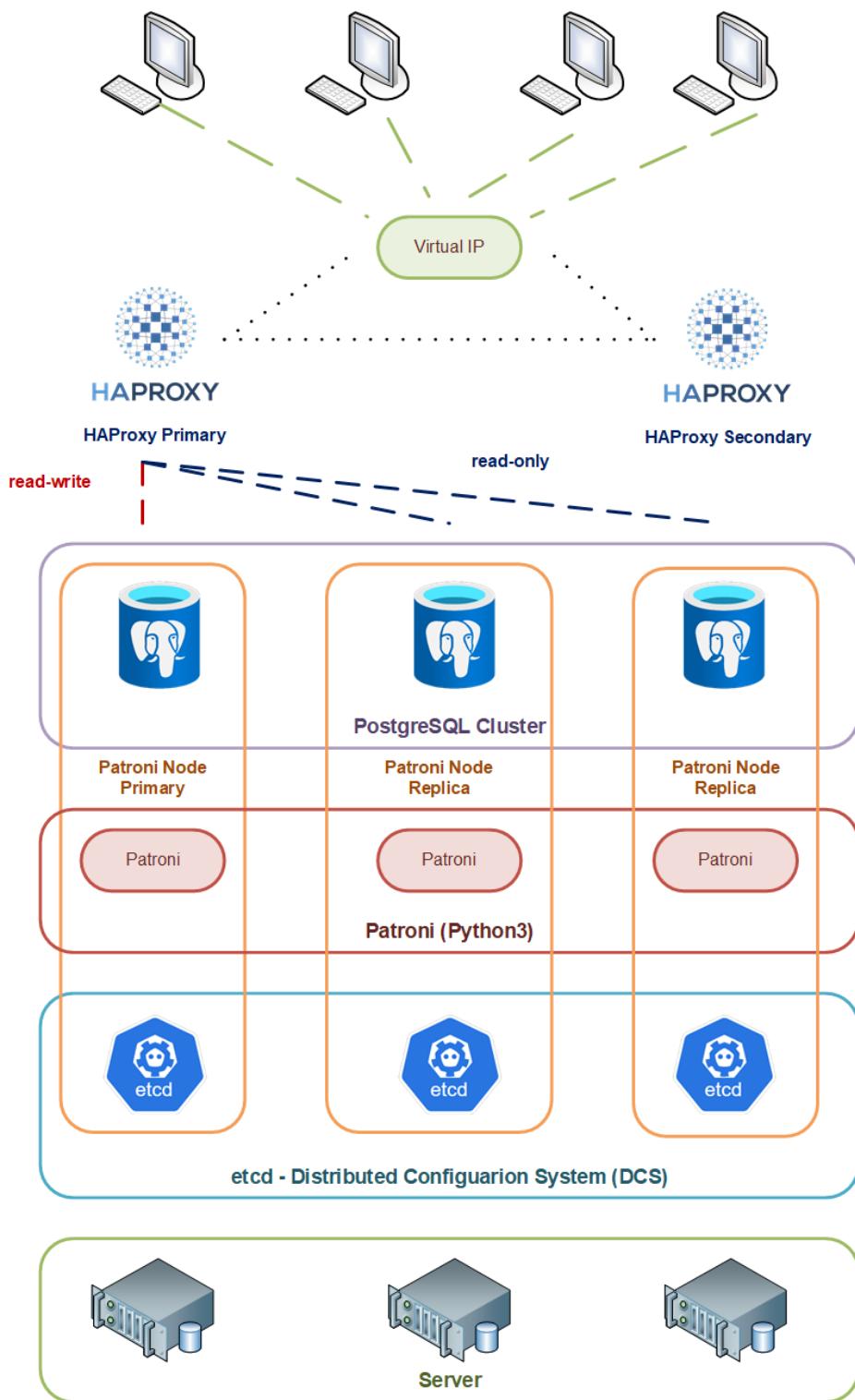


Abbildung 3.20: Patroni-Architektur

### 3.1.5.7.8 Maintenance

Anhang - Maintenance

### 3.1.5.7.9 Synergien und Mehrwert

Patroni kann nicht nur mit Citus zu einem Distributed / Sharded SQL System umgebaut werden, es ist auch Kern von Stackgres.

Damit könnten die API und Skripte in beiden Welten verwendet werden.

Der Aufwand für die Verwaltung und optimierung würde stark gesenkt.

Projekte wie vitabaks / postgresql\_cluster[92] bieten zudem die vorlage für eine noch stärkere automatisierung.

### 3.1.5.8 Stackgres mit Citus

Stackgres ist eine PostgreSQL Implementation die dafür vorgesehenen ist, in einem Kubernetes Cluster betrieben zu werden.

An sich wäre Stackgres nur eine Implementation von Patroni in Kubernetes inkl. Load Balancer.

Nun kommt das Citus-Plugin ins Spiel, welches aus einer jeden Monolithischen, klassischen PostgreSQL Installation eine Distributed SQL Umgebung macht.

Citus Data, der Entwickler von Citus, wiederum ist in den Microsoft Konzern eingebettet

#### 3.1.5.8.1 Core-Features

Die wichtigsten Features von Stackgres sind[21]:

- k8s Integration
- Deklaratives k8s CRD
- Automatische Backups
- Grafana und Prometheus Integration
- Management Web Konsole
- Erweiterte Replikationsmöglichkeiten
- Integriertes Pooling
- Integrierter Proxy

### 3.1.5.8.2 Replikation

Stackgres bietet Asynchrone und Synchrone Replikation, Gruppenreplikation sowie kaskadierende Replikation an.

Citus bietet sein eigenes Modell mit dem Sharding an.

### 3.1.5.8.3 Proxy

Stackgres hat den Proxy bereits mit envoy[19] implementiert.

### 3.1.5.8.4 Pooling

PgBounder[31] ist bereits integriert, es braucht also keinen weiteren Pooler.

### 3.1.5.8.5 API / Skripte

Stackgres wird Primär über YAML-Files und Kubernetes gesteuert, bietet aber eine eigene API an.

Citus bietet ebenfalls eine eigene API, mit der Citus vollständig verwaltet werden kann.

### 3.1.5.8.6 Architektur

#### 3.1.5.8.6.1 StackGres

Stackgres packt PostgreSQL, Patroni, PgBouncer und envoy in einen Kubernetes Pod:

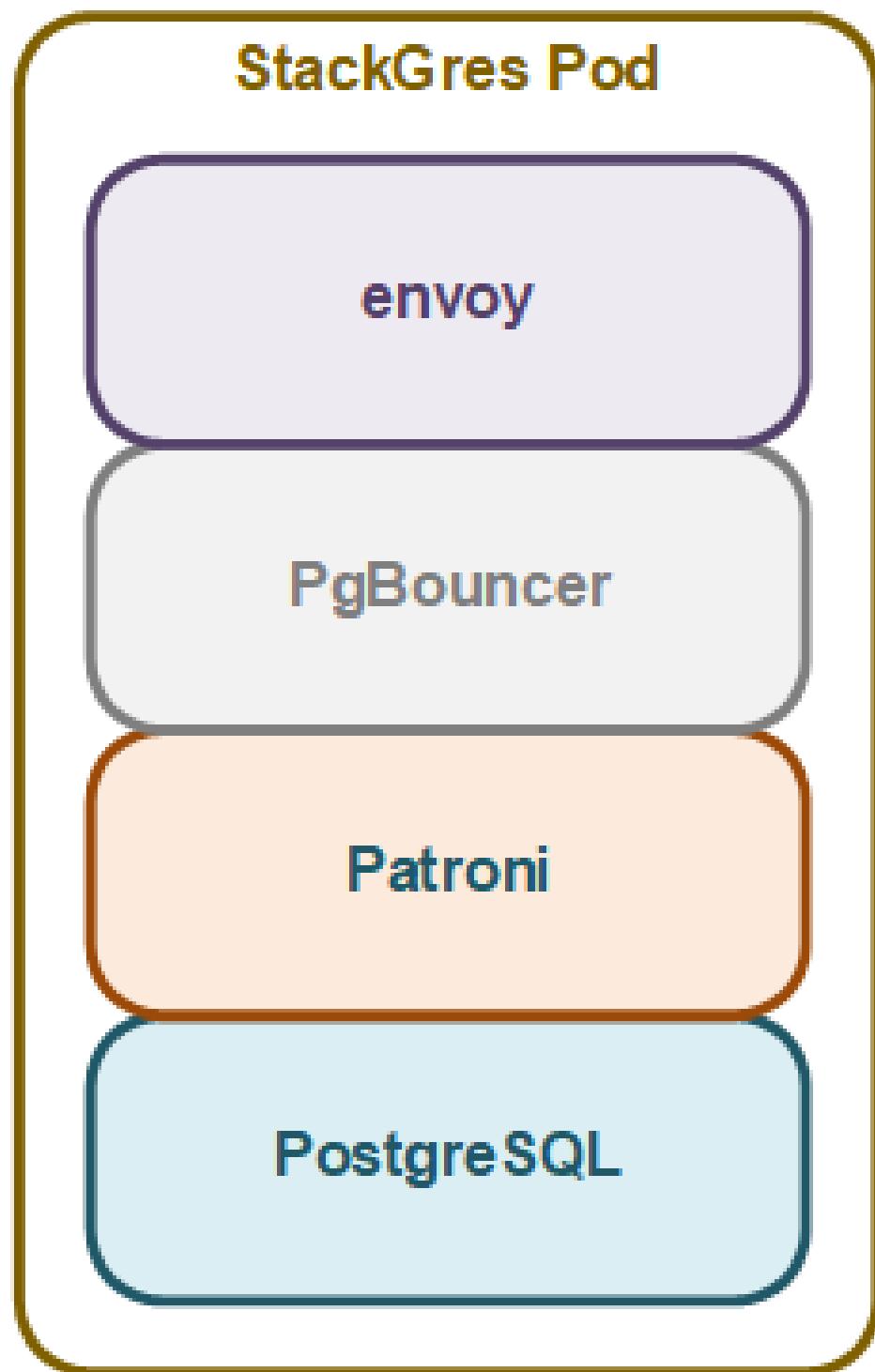


Abbildung 3.21: Stackgres - Grundarchitektur

### 3.1.5.8.6.2 Citus Coordinator und Workers

Citus arbeitet mit einem Coordinator-Node, der jedes Query analysiert und an einen Worker-Node weitergibt.

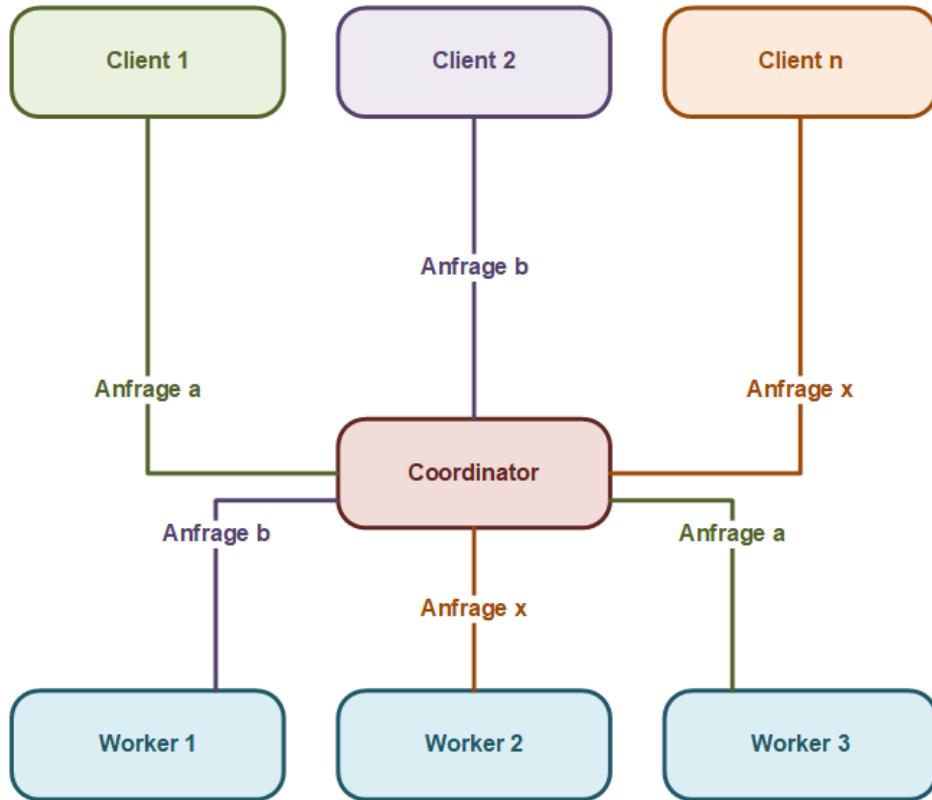


Abbildung 3.22: Citus - Coordinator und Workers

### 3.1.5.8.6.3 Citus Sharding

Citus bietet zwei Sharding-Modelle an.

**Row-based sharding** Beim diesen sharding werden Tabellen anhand einer Distribution Column aufgeteilt. [16, 8]

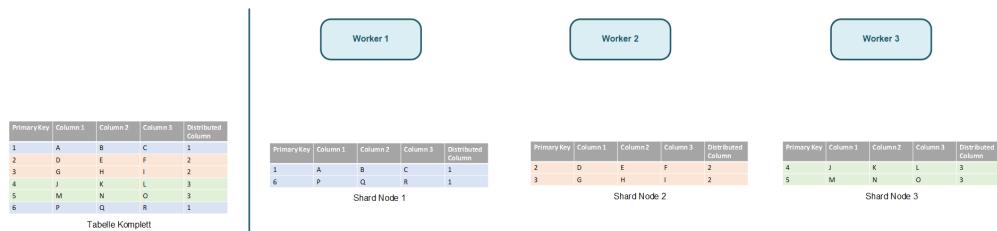


Abbildung 3.23: Citus - Row-Based-Sharding

### Schema-based sharding

## Diplomarbeit

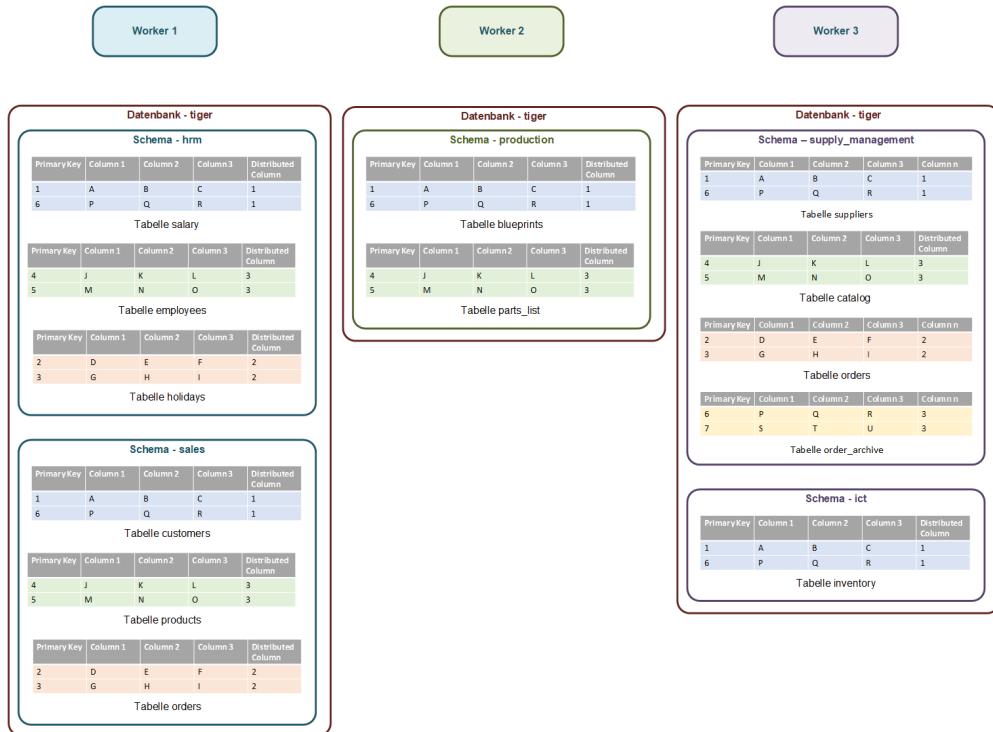


Abbildung 3.24: Citus - Schema-Based-Sharding

**Schlussfolgerung** Beide Sharding-Methoden haben eine grosse Schwäche. In Version 7.2 konnte noch ein Replikationsfaktor angegeben werden[15], ab Version 11 wurde auch diese Variante gestrichen und man konnte noch eine 1:1 Repliacation auf einen Worker fahren[10]. Spätestens mit Version 12 steht auch dies nicht mehr zur verfügung, man muss eine Replication auf e Sie sind nicht vollständig ACID-Konform da Datenverlust entstehen kann, wenn ein Node wegfällt. Dies muss aber bei der evaluation mittels Tests noch bestätigt werden.

Die Shards müssen aber, stand heute, mit entsprechenden mit Replikation gesichert werden[14]. Daraus ergibt sich aber ein nicht zu vernachlässigbarer Mehraufwand, wenn man self-healing Nodes implementieren möchte.

Jeder Node ist für sich genommen, eine eigene Zone, um sicherzustellen, dass es zu keinem Datenverlust kommt,

müsste jeder Shared-Node in eine der jeweiligen Zonen repliziert werden.

Das heisst, es müssten  $Shard - Nodes^2$  Replika-Nodes erstellt werden, hier ein Schematisches-Beispiel mit drei Shard-Nodes:

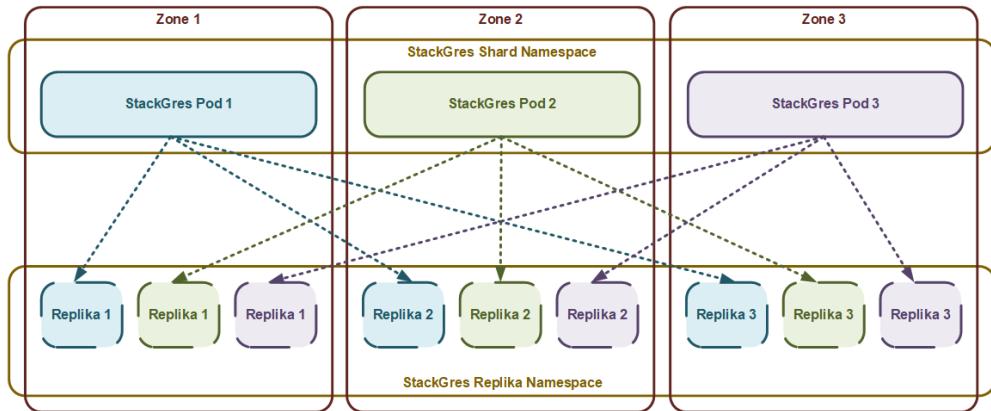


Abbildung 3.25: StackGres-Citus - Shard-Replikation

Die Automation und nur schon die Konfiguration für das Mitskalieren, dürfte einiges an Zeit in Anspruch nehmen.

Eine nicht unwesentliche Folge, wäre ein starker Rückgang des Throughput's und Performance-Einbussen.

Alternativ kann natürlich ein Klassischer Replika-Server verwendet werden, wo die ganze Datenbank gesichert wird.

Bis alle Daten wieder in den StackGres-Citus-Cluster zurückgeholt wurden, das Re-Balancing abgeschlossen ist usw.,

muss der ganze Cluster für die User unerreichbar sein, da dieser in dieser Zeit nicht mehr konsistent ist.

Dieser zweite Ansatz bietet zwar Vorteile beim Throughput, doch im Fehlerfall ist ein HA-Betrieb nur noch begrenzt garantiebar.

### 3.1.5.8.7 Maintenance

Anhang - Maintenance

### 3.1.5.9 YugabyteDB - Distributed SQL 101

yugabyteDB - Distributed SQL 101 ist eine nahezu komplett PostgreSQL Kompatible Datenbank. Sie ist eine Distributed SQL Datenbank, also eine Verteilte Datenbank[88].

#### 3.1.5.9.1 Core-Features

Die wichtigsten Features von YugabyteDB sind[7]:

- PostgreSQL Kombatibel

## Diplomarbeit

- Cassandra-Kompatibilität
- Horizontal skalierbarkeit
- Global verteilbar
- Cloud Native

### 3.1.5.9.2 Replikation

### 3.1.5.9.3 Proxy

YugabyteSQL nutzt Kubernetes und seine Core-Functions als Load Balancer. Ein zusätzlicher Proxy wird nicht benötigt.

### 3.1.5.9.4 Pooling

YugabyteDB hat ein Connection Pooling mit dem YSQL Connection Manager integriert[58].

### 3.1.5.9.5 API / Skripte

YugabyteDB bietet eigene APIs[5] und CLIs[12] für das Verwalten an.

Diese bieten auch die möglichkeit, abgesichert zu werden.

### 3.1.5.9.6 Architektur

yugabyteDB ist kein reines RDBMS, resp. gar keines. Die Basis besteht aus einem Key-Value-Store. Darüber wurde eine Cassandra-like Query API und eine PostgreSQL like SQL API aufgebaut:

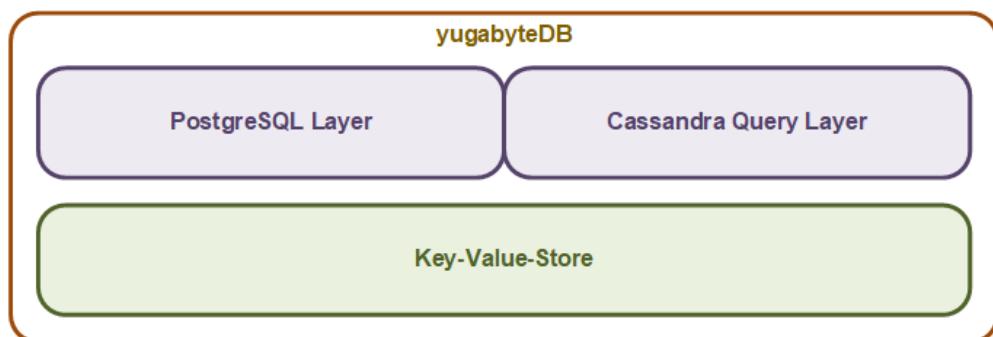


Abbildung 3.26: yugabyteDB - Grundkonzept

Der Basisaufbau wiederum beinhaltet diverse Dienste für das Sharding, die Replikation und Transaktionen:

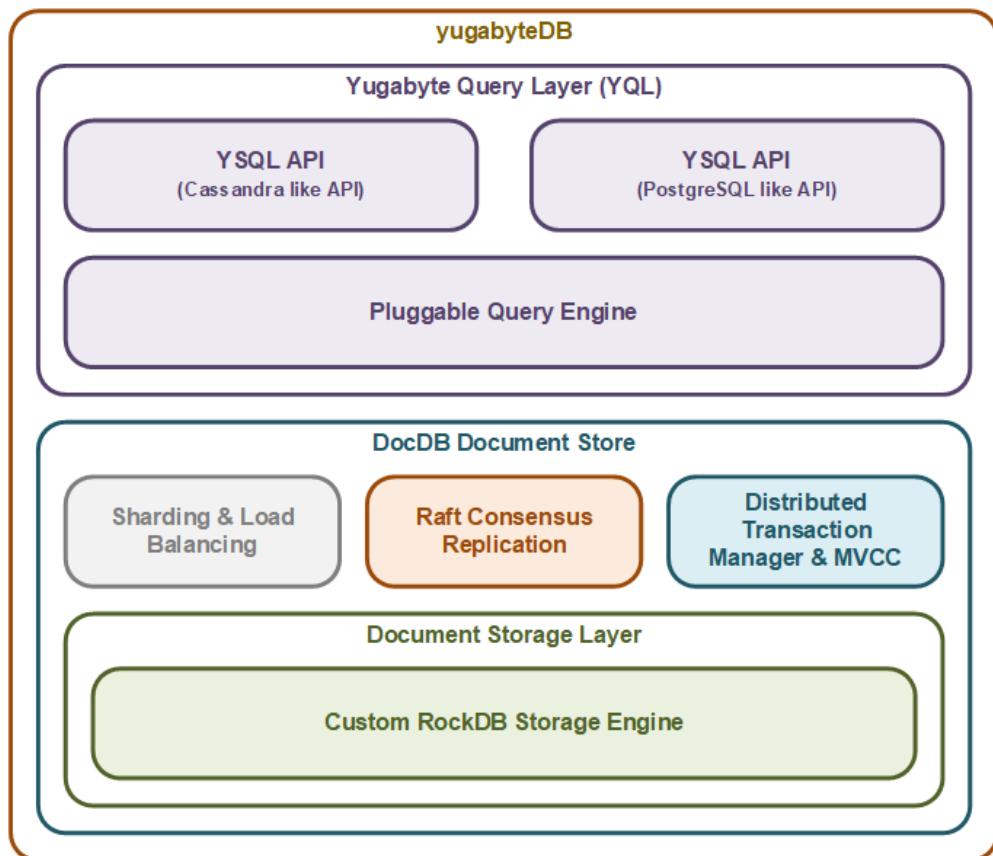


Abbildung 3.27: YugabyteDB - Architektur

### 3.1.5.9.6.1 YugabyteDB - Sharding

yugabyteDB teilt seine Tabellen in Tablets auf. Die Aufteilung kann gemäss Sharding-Standards gemacht werden:

Primary Key	Column 1	Column 2	Column 3	Distributed Column
1	A	B	C	1
2	D	E	F	2
3	G	H	I	2
4	J	K	L	3
5	M	N	O	3
6	P	Q	R	1

Tabelle Komplett

Primary Key	Column 1	Column 2	Column 3	Distributed Column
1	A	B	C	1
6	P	Q	R	1

Tablet 1

Primary Key	Column 1	Column 2	Column 3	Distributed Column
2	D	E	F	2
3	G	H	I	2

Tablet 2

Primary Key	Column 1	Column 2	Column 3	Distributed Column
4	J	K	L	3
5	M	N	O	3

Tablet 3

Abbildung 3.28: YugabyteDB - Sharding

Dabei hat jedes Tablet auf einem Node einen Leader, der an die Follower auf den anderen Nodes repliziert:

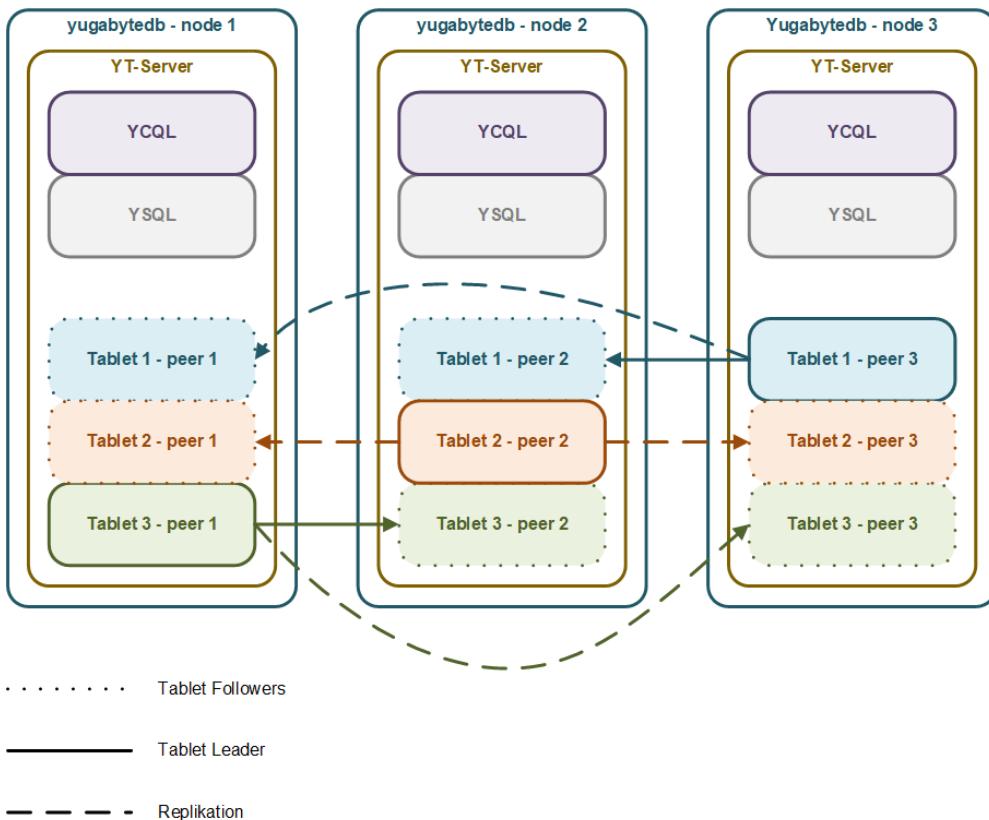


Abbildung 3.29: YugabyteDB - Tablet - Leader und Follower

Mit dem Replikationsfaktor kann angegeben werden, auf wie vielen Nodes ein Tablet repliziert werden soll. Bei einem 4-Node System können z.B. einige Tablets einen Faktor 3 haben, dass

## Diplomarbeit

heisst, dass die Daten nur auf 3 Nodes repliziert werden. Bei einem Replikationsfaktor 4 werden die Daten auf alle Nodes repliziert. Dies wird mit einem eigenen Service, dem YB-TServer service [35] geregelt:

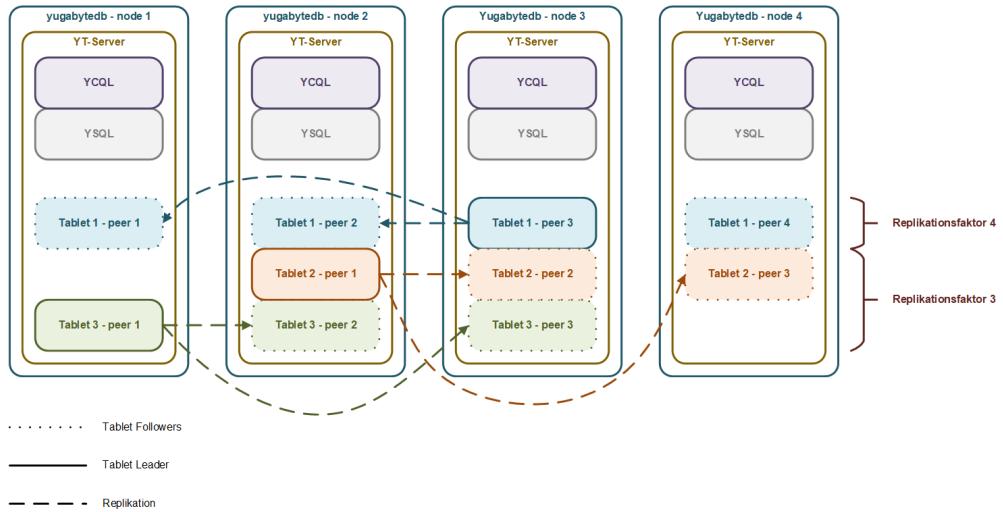


Abbildung 3.30: yugabyteDB - Tablet - Replikationsfaktor

Durch das Raft-Protokoll werden die Tablet-Leader regelmäßig gewechselt.

Mehrere Nodes können zu Zonen zusammengebunden werden, die dann z.B. auf verschiedene Rechenzentren verteilt werden:

# Diplomarbeit

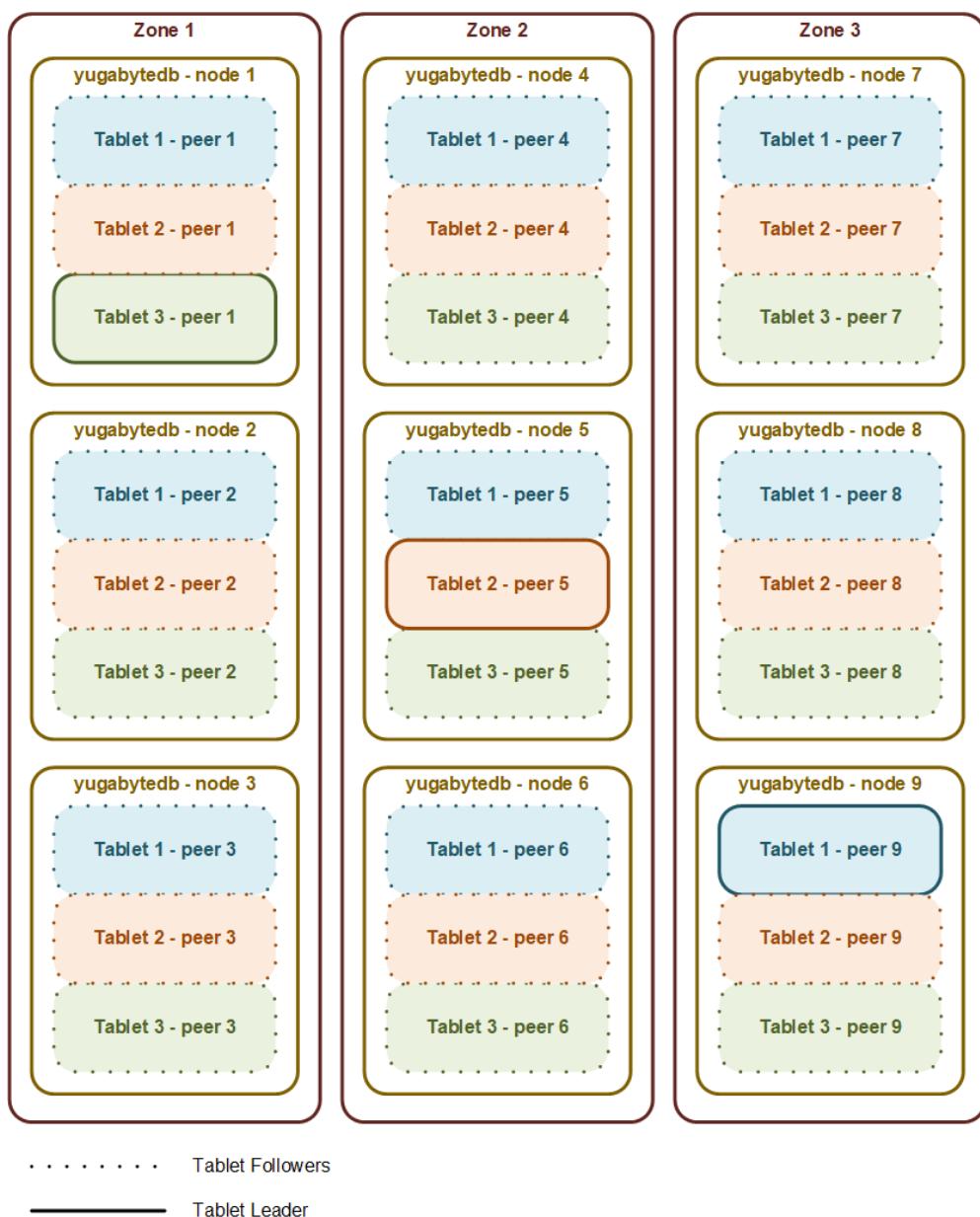


Abbildung 3.31: YugabyteDB - Zonen

Dies wird dann sinnvoll, wenn eine gewisse Ausfalltoleranz erreicht werden soll. Fällt nämlich ein Tablet Peer oder ein Node in einer Zone aus, so wird die ganze Zone sofort als nicht mehr Arbeitsfähig angesehen. Entsprechend werden in allen Nodes die Tablet-Leader stillgelegt und auf die übrigen Zonen verteilt. YuganyteDB nennt dies Zone outage Tolerance[32].

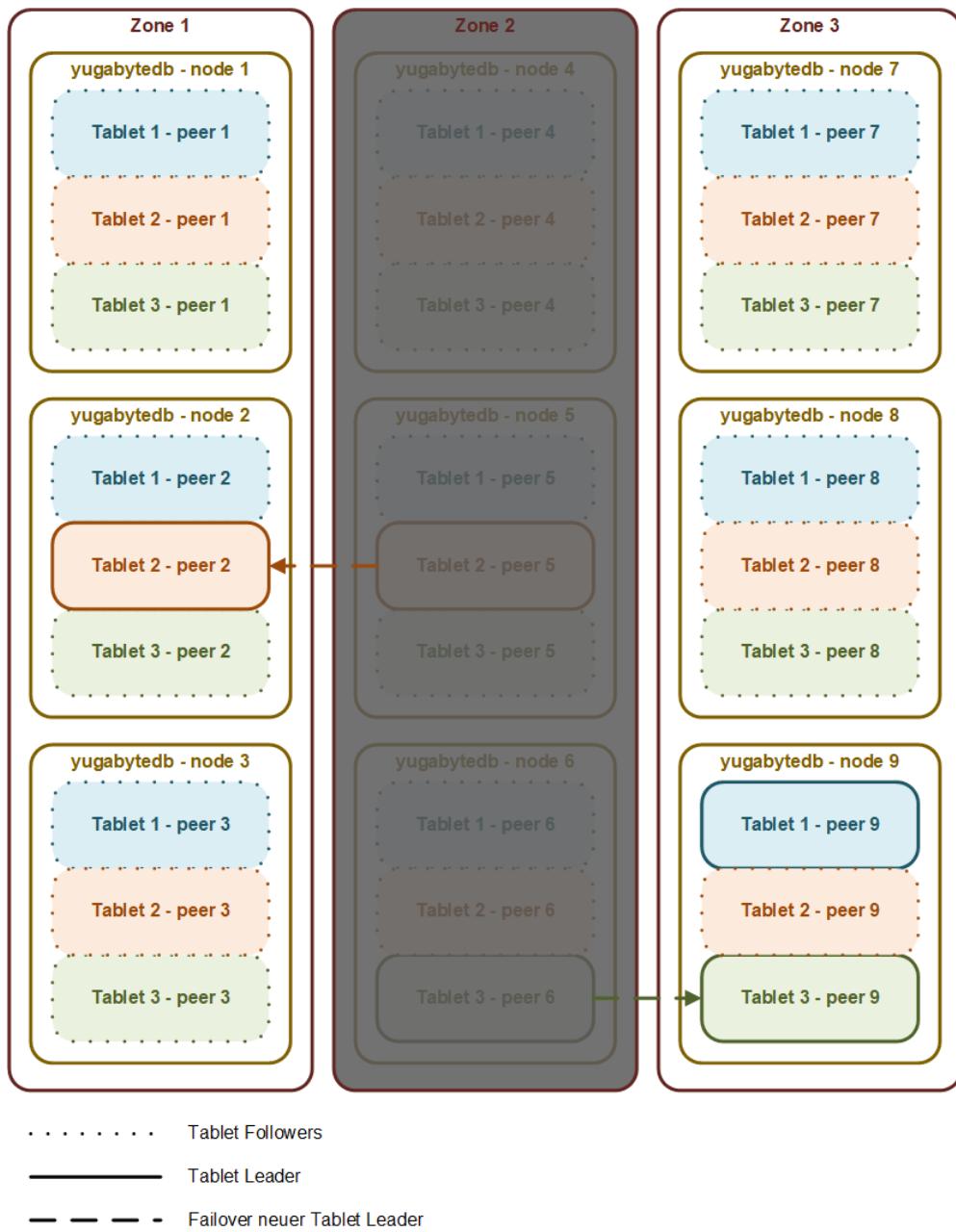


Abbildung 3.32: YugabyteDB - Zone outage Tolerance

### 3.1.5.9.7 Maintenance

Anhang - Maintenance

### 3.1.5.9.8 Synergien und Mehrwert

Der grosse Benefit von YugabyteDB ist sein Distributed SQL Ansatz.

## Diplomarbeit

Zudem bietet YugabyteDB eine vollständige Cassandra Integration.

Der Benefit ist auf jeden Fall gegeben.

### 3.1.6 Vorauswahl

Folgende Lösungen werden nicht evaluiert, sondern bereits zu Beginn ausgeschieden:

Nr.	Lösung	Status	Begründung
1	KSGR-Lösung	Vorausgeschieden	Hat nur einen Standy / Replika-Node. Failover Funktioniert nur bei kleineren Datenmengen wirklich in einer vernünftigen Zeit.
2	pgpool-II	Vorausgeschieden	pgpool-II hat kein GitHub-Repository und bietet daher keine vergleichswerte mittels Github Insights.
3	pg_auto_failover	Vorausgeschieden	pg_auto_failover würde zwar Citus-Support bieten, allerdings gibt es keine gut dokumentierte Implementation für Kubernetes. Erfüllt daher das Kriterium für die Synergien nicht
4	CloudNativePG	Vorausgeschieden	CloudNativePG ist keine vollständige Cloud Native Lösung. Mittels Citus könnte sogar eine Distributed SQL Lösung implementiert werden. Die Grundarchitektur bleibt aber Monolithisch mit einem Primary und Replicas. Und da kein Benefit in Form von Synergien vorhanden sind, fällt CloudNativePG raus.
8	Citus row-based-sharding	Vorausgeschieden	Citus row-based-sharding wäre Hocheffizient wenn es um Ressourcenverteilung geht und zudem echtes Sharding. Allerdings setzt es Anpassungen an den Tabellen der Applikationen voraus.
			Das KSGR ist allerdings kein Softwarehaus und kann keine Forks durchführen, auch weil viele Applikationen zertifiziert sein müssen. Scheitert daher an der Machbarkeit

Tabelle 3.9: Vorauswahl - Ausgeschieden

Entsprechend werden nur noch nachfolgende Lösungen genauer betrachtet:

Nr.	Lösung	Status	Begründung
5	Patroni	Evaluation	Patroni kann als Monolithisches System genutzt werden, ist aber auch Kern von Stackgres. Die API und Skripte können also in beiden Welten verwendet werden
6	Stackgres mit Citus	Evaluation	Bietet eine einfache und kompakte Möglichkeit für ein Distributed SQL System.
7	Yugabyte-DB	Evaluation	Da Patroni unter der Haube ist, kann die API und sonstige Skripte auch auf einem Monolithischen System eingesetzt werden.

Tabelle 3.10: Vorauswahl - Evaluation

### 3.1.7 Installation verschiedener Lösungen

Entsprechend wurden folgende Server bereitgestellt:

## Diplomarbeit

Server	Typ	Funktion	Full Qualified Device Name	IP
sk1183	Distributed SQL	Server	sk1183.ksgr.ch	10.0.20.97
sk1184	Distributed SQL	Agent	sk1184.ksgr.ch	10.0.20.104
sk1185	Distributed SQL	Agent	sk1185.ksgr.ch	10.0.20.105
sk1232	Monolith	Server	sk1232.ksgr.ch	10.0.20.110
sk1233	Monolith	Server	sk1233.ksgr.ch	10.0.20.111
sk1234	Monolith	Server	sk1234.ksgr.ch	10.0.20.112
sk9016	Benchmark Server	Client	sk9016.ksgr.ch	10.0.21.216
vks0032	Distributed SQL	Virteulle IP	vks0032.ksgr.ch	10.0.20.106
vks0040	Monolith	Virteulle IP	vks0040.ksgr.ch	10.0.20.113

Tabelle 3.11: Evaluationssyssteme

### 3.1.7.1 rke2 - Evaluationsplattform

Die Grundsätzliche Evaluationsplattform für Distributed SQL / Shards sieht folgendermassen aus:

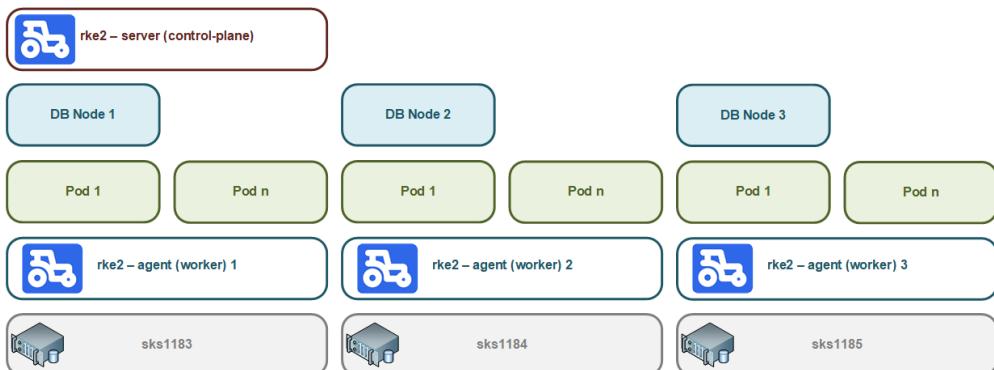


Abbildung 3.33: Evaluationssystem - Distributed SQL / Shards

Die Konfiguration der rke2-Nodes sieht folgendermassen aus:

<b>Kubernetes Runtime</b>	rke2
<b>Container-Enviroment</b>	containerd
<b>Container Network Interface (CNI)</b>	cilium
<b>Cloud Native Storage (CNS)</b>	local-path-provisioner
<b>cluster-cidr</b>	198.18.0.0/16
<b>service-cidr</b>	198.18.0.0/16
<b>External IP Range</b>	10.0.20.106,10.0.20.150-10.0.20.155

Tabelle 3.12: Evaluationssystem - Distributed SQL / Sharding

## Diplomarbeit

### 3.1.7.2 Patroni

#### 3.1.7.2.1 Architektur

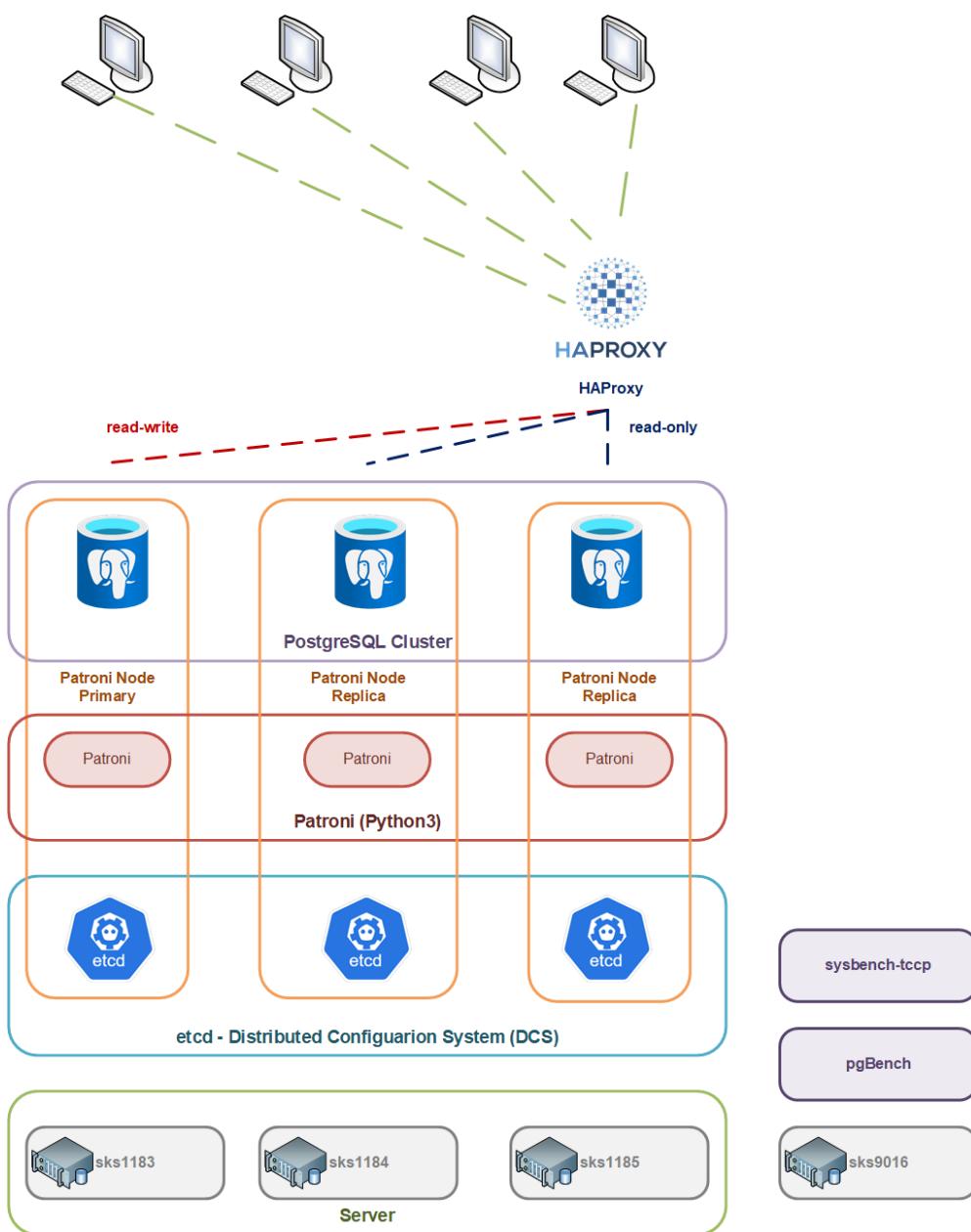


Abbildung 3.34: Patroni - Evaluationsarchitektur

### 3.1.7.3 StackGres - Citus

### 3.1.7.3.1 Architektur

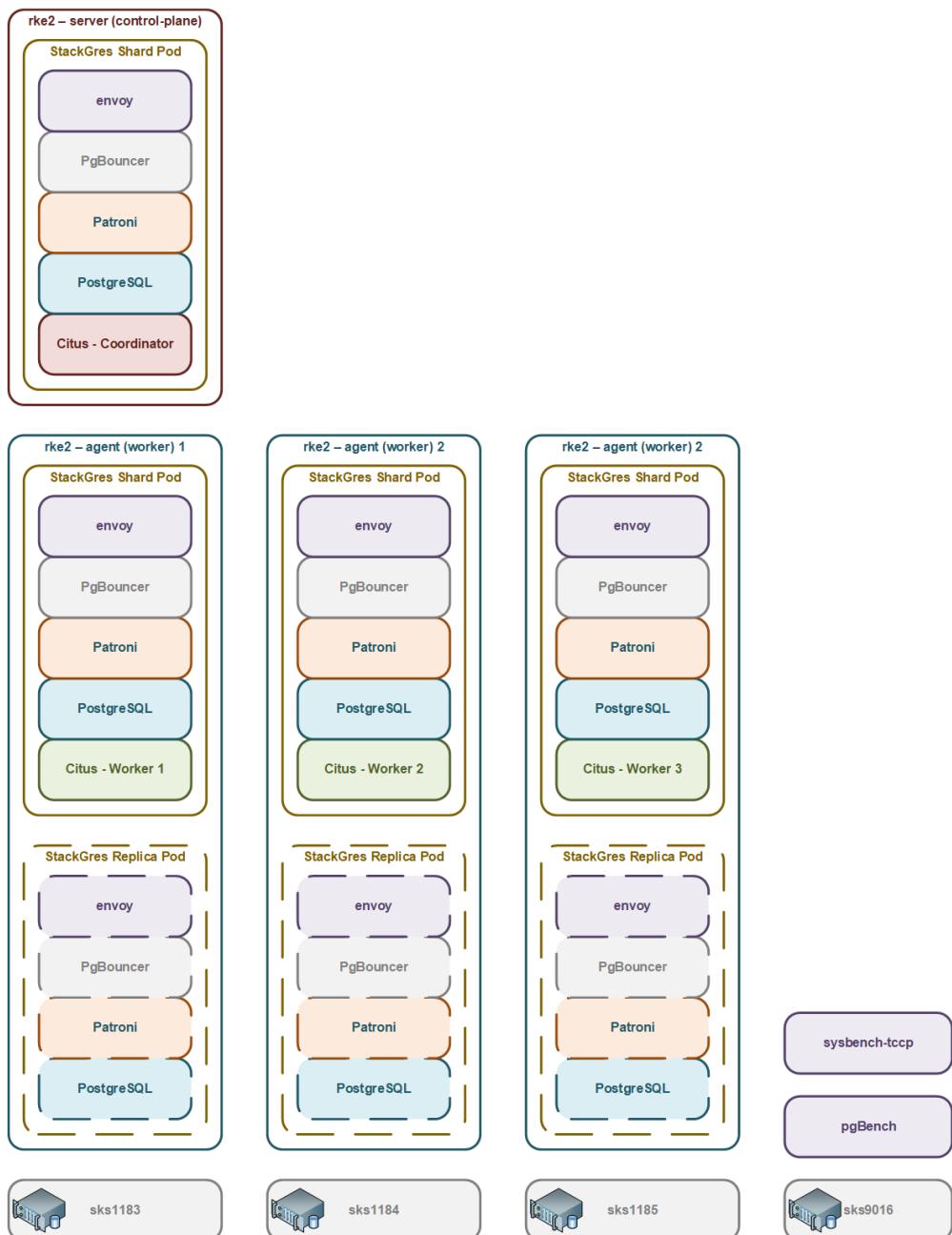


Abbildung 3.35: Stackgres - Citus - Evaluationsarchitektur

### 3.1.7.4 YugabyteDB

#### 3.1.7.4.1 Installation

Während der Installation des YugabyteDB Evaluations-Enviroment wurde festgestellt, das man zwei Varianten installieren kann. YugabyteDB (Repository Yugabyte) und YugabyteDB Anywhere

## Diplomarbeit

(Repository `yugawre`):

```

Context: default          <C>   Copy
Cluster: default          <E>   Edit
User: default              <N>   Next Match
K9s Rev: v0.31.8 > v0.32.4 <Shift-Ins> Prev Match
K8s Rev: v1.29.0+rke2r1    <R>   Toggle Auto-Refresh
CPU: 1%                   <F>   Toggle Fullscreen
MEM: 38%
Name: yw-test-yugaware-pg-upgrade
optional: false
pg-init:
  Type: ConfigMap (a volume populated by a ConfigMap)
  Name: yw-test-yugaware-pg-prerun
  optional: false
pg-sample-config:
  Type: ConfigMap (a volume populated by a ConfigMap)
  Name: yw-test-pg-sample-config
  optional: false
kube-api-access-rgtwb:
  Type: Projected (a volume that contains injected data from multiple sources)
  TokenExpirationSeconds: 3600
  ConfigMapName: kube-root-ca.crt
  ConfigMapOptional: enabled
  DownwardAPI: true
QoS Class: Burstable
Node-Selectors: <none>
Tolerations: node.kubernetes.io/not-ready:NoExecute op=Exists for 300s
node.kubernetes.io/unreachable:NoExecute op=Exists for 300s
Events:
  Type  Reason           Age            From           Message
  ----  ----            --            --            --
  Normal Scheduled        3m22s          default-scheduler  Successfully assigned yb-platform/yw-test-yugaware-0 to sks1185
  Normal Pulling         2m39s (x3 over 3m22s)  kubelet        Pulling image "quay.io/yugabyte/yugaware:2.20.2.1-b3"
  Warning Failed         2m38s (x3 over 3m21s)  kubelet        Failed to pull image "quay.io/yugabyte/yugaware:2.20.2.1-b3": failed to pull and unpack image "quay.io/yugabyte/yugaware:2.20.2.1-b3": failed to resolve reference "quay.io/yugabyte/yugaware:2.20.2.1-b3": unexpected status from HEAD request to https://quay.io/v2/yugabyte/yugaware/manifests/2.20.2.1-b3: 401 UNAUTHORIZED
  Warning Failed         2m38s (x3 over 3m21s)  kubelet        Error: ErrImagePull
  Normal BackOff          2m11s (x4 over 3m20s)  kubelet        Back-off pulling image "quay.io/yugabyte/yugaware:2.20.2.1-b3"
  Warning Failed         2m11s (x4 over 3m20s)  kubelet        Error: ImagePullBackoff
  Warning FailedToRetrieveImagePullSecret 117s (x8 over 3m23s)  kubelet        Unable to retrieve some image pull secrets (yugabyte-k8s-pull-secret); attempting to pull the image may not succeed.

```

Abbildung 3.36: YugabyteDB - Subscription `yugawre`

Es stellte sich auch heraus, dass wenn man YugabyteDB 4 Cores pro Node zur Verfügung geben will (je zwei für den `master` und `tserver`), der Server mehr als 4 Cores haben muss.

Andernfalls wird Kubernetes einen der beiden Pods nicht deployen, weil zuwenig Cores zur Verfügung stehen.

Bei der Konstellation rke2, Cilium und MetalLB, muss nebst dem IPAddressPool auch ein L2Advertisement für den Pool gesetzt werden.

Ansonsten kann die im YugabyteDB `values.yaml` gesetzte IP für den `tserver` von außen nicht angesprochen werden:

```

1 ---
2 apiVersion: metallb.io/v1beta1
3 kind: L2Advertisement
4 metadata:
5   name: l2adv
6   namespace: metallb-system
7 spec:
8   ipAddressPools:
9     - distributed-sql
10

```

Listing 3.1: metallb - Konfig YAML - Detail L2Advertisement

## Diplomarbeit

Dieses Problem ist schwer zu greifen und hat zwei Tage in Anspruch genommen, es zu lösen. Die Vorschläge zum Lösen des Problems reichten von deaktivieren von kube-proxy bis hin zu einer Migration zum Cilium-Loadb-Balancers.

Mit diesem funktionierte dann nicht einmal mehr die Installation von YugabyteDB.

Lösung brachte nur ein GitHub-Eintrag[27], wo oben genannter Ansatz empfohlen wurde.

### 3.1.7.4.2 Konfiguration

Damit nicht der YugabyteDB Anywhere-Service installiert wird, muss das entsprechende Image gesetzt werden:

```

1 ...
2 Image:
3   repository: "yugabytedb/yugabyte"
4   tag: 2.20.2.1-b3
5   pullPolicy: IfNotPresent
6   pullSecretName: ""
7 ...
8

```

Listing 3.2: YugabyteDB - Helm Chart Manifest - Detail Image

Die StorageClass muss im values.yaml gesetzt werden, einmal für den master und einmal für den tserver

```

1 ...
2 storage:
3   ephemeral: false # will not allocate PVs when true
4   master:
5     count: 1
6     size: 3Gi
7     storageClass: "yb-storage"
8   tserver:
9     count: 1
10    size: 3Gi
11    storageClass: "yb-storage"
12 ...
13

```

Listing 3.3: YugabyteDB - Helm Chart Manifest - Detail StorageClass

Dem node werden je 4 Cores zur Verfügung gestellt. Zwei für den master und zwei für den tserver. Beide erhalten 4GiB Memory:

```

1 ...
2 resource:
3   master:
4     requests:
5       cpu: "1"

```

## Diplomarbeit

```

6     memory: 2Gi
7   limits:
8     cpu: "1"
9       ## Ensure the 'memory' value is strictly in 'Gi' or 'G' format. Deviating
10      from these formats
11      ## may result in setting an incorrect value for the 'memory_limit_hard_bytes'
12      ## flag.
13      ## Avoid using floating numbers for the numeric part of 'memory'. Doing so
14      may lead to
15      ## the 'memory_limit_hard_bytes' being set to 0, as the function expects
16      integer values.
17     memory: 2Gi
18 tserver:
19   requests:
20     cpu: "1"
21     memory: 4Gi
22   limits:
23     cpu: "1"
24     ## Ensure the 'memory' value is strictly in 'Gi' or 'G' format. Deviating
25      from these formats
26      ## may result in setting an incorrect value for the 'memory_limit_hard_bytes'
27      ## flag.
28      ## Avoid using floating numbers for the numeric part of 'memory'. Doing so
29      may lead to
30      ## the 'memory_limit_hard_bytes' being set to 0, as the function expects
31      integer values.
32     memory: 4Gi
33 ...
34

```

Listing 3.4: YugabyteDB - Helm Chart Manifest - Detail Resources

Die Shards, oder Tablets wie sie Yugabyte nennt, sollen auf allen drei Nodes repliziert werden:

```

1 ...
2 replicas:
3   master: 3
4   tserver: 3
5     ## Used to set replication factor when isMultiAz is set to true
6   totalMasters: 3
7 ...
8

```

Listing 3.5: YugabyteDB - Helm Chart Manifest - Detail Replika

Wichtig ist auch, dass der YSQL-Dienst aktiv ist, damit PostgreSQL Abfragen abgesetzt werden können.

Deshalb muss der Dienst aktiv sein und darf nicht deaktiviert werden:

```

1 ...
2 # Disable the YSQL

```

```

3 disableYsql: false
4 ...
5

```

Listing 3.6: YugabyteDB - Helm Chart Manifest - Detail Disable YSQL

Nun muss die Domain und die Service-Endpoints konfiguriert werden.

Der Domainname bleibt vorerst `cluster.local` wie Default hinterlegt.

Die Servicenamen und Ports werden nicht angeastet, wichtig ist die LoadBalancer-IP.

Sie ist entsprechend der gewählten VirtualIP mit `10.0.20.106` zu setzen.

```

1 ...
2 domainName: "cluster.local"
3
4 serviceEndpoints:
5   - name: "yb-master-ui"
6     type: LoadBalancer
7     annotations: {}
8     clusterIP: ""
9     ## Sets the Service's externalTrafficPolicy
10    externalTrafficPolicy: ""
11    app: "yb-master"
12    loadBalancerIP: ""
13    ports:
14      http-ui: "7000"
15
16   - name: "yb-tserver-service"
17     type: LoadBalancer
18     annotations:
19       metallb.universe.tf/loadBalancerIPs: 10.0.20.106
20     clusterIP: ""
21     ## Sets the Service's externalTrafficPolicy
22     externalTrafficPolicy: ""
23     app: "yb-tserver"
24     loadBalancerIP: ""
25     ports:
26       tcp-yql-port: "9042"
27       tcp-yedis-port: "6379"
28       tcp-ysql-port: "5433"
29 ...
30

```

Listing 3.7: YugabyteDB - Helm Chart Manifest - Detail Domainname und Service-Endpoints

### 3.1.7.4.3 Connection

Auch mit einem PostgreSQL-

### 3.1.8 Testing Evaluationssysteme

#### 3.1.8.1 Patroni

Art	Test Case Nr.	Test Case	Erwartetes Ergebnis	Eingetretenes Ergebnis	Begründung
Failover	1	Automatismus	Wird der Primary Server vom Netz genommen, führt Patroni einen Failover auf einen Replika-Node	Eingetroffen	Connection-Stabilität kann nur hergestellt werden, wenn entweder die Applikation dazu in der Lage ist oder man einen Connection-Pooler wie pgBouncer einsetzt. Es wurde aber keiner eingesetzt.
Failover	2	Connection-Stabilität	Bestehende Connections dürfen nicht getrennt werden.	Nicht eingetroffen	Auch hier hängt die Stabilität an den Settings der Applikation und einem Connection-Pooler.
Failover	3	Geschwindigkeit	Der Failover muss so schnell stattfinden, dass offene Connections nicht wegen eines Timeouts geschlossen werden.	Bedingt eingetroffen	Connection-Stabilität kann nur hergestellt werden, wenn entweder die Applikation dazu in der Lage ist oder man einen Connection-Pooler wie pgBouncer einsetzt. Es wurde aber keiner eingesetzt.
Switchover	4	Skript / API	Mit der Patroni REST-API wird der Switchover ausgeführt	Eingetroffen	
Switchover	5	Skript / API	Mit dem Patroni Commandset wird er Switchover ausgeführt	Eingetroffen	
Switchover	6	Connection-Stabilität	Bestehende Connections dürfen nicht getrennt werden.	Eingetroffen	
Switchover	7	Geschwindigkeit	Der Switchover muss so schnell stattfinden, dass offene Connections nicht wegen eines Timeouts geschlossen werden.	Nicht eingetroffen	Connection-Stabilität kann nur hergestellt werden, wenn entweder die Applikation dazu in der Lage ist oder man einen Connection-Pooler wie pgBouncer einsetzt. Es wurde aber keiner eingesetzt.
Restore	9	Skript / API	Mit der Patroni REST-API wird der Primary-Node wiederhergestellt	Eingetroffen	
Restore	10	Skript / API	Mit dem Patroni Commandset der Primary-Node wiederhergestellt	Eingetroffen	
Restore	11	Skript / API	Mit der Patroni REST-API wird ein Replika-Node wiederhergestellt	Eingetroffen	
Restore	12	Skript / API	Mit dem Patroni Commandset ein Replika-Node wiederhergestellt	Eingetroffen	
Restore	13	Datensicherheit	Beim Restore des Primary-Nodes dürfen keine Daten, die seit dem Failover geschrieben wurden, darf es zu keinem Datenverlust kommen	Eingetroffen	Connection-Stabilität kann nur hergestellt werden, wenn entweder die Applikation dazu in der Lage ist oder man einen Connection-Pooler wie pgBouncer einsetzt. Es wurde aber keiner eingesetzt.
Restore	14	Connection-Stabilität	Beim Restore des Primary-Nodes dürfen keine Connections geschlossen werden.	Nicht eingetroffen	

Tabelle 3.13: Testresultate Evaluation Patroni

#### 3.1.8.2 StackGres -Citus

StackGres kann nicht alle Anforderungen erfüllen. Obwohl es mit envoy und pgBounder einen Proxy und einen Connection Pooler gibt, scheint dies nicht über die Coordinator-Nodes selbst zu gehen. Daher brechen bestehende Connections ab oder laufen irgendwann in ein Timeout, wenn Kubernetes Nodes nicht schnell genug heruntergefahren werden.

Art	Test Case Nr.	Test Case	Erwartetes Ergebnis	Eingetretenes Ergebnis	Begründung
Failover	1	Automatismus	Wird der Primary Server vom Netz genommen, führt Patroni einen Failover auf einen Replika-Node	Eingetroffen	
Failover	2	Connection-Stabilität	Bestehende Connections dürfen nicht getrennt werden.	Nicht eingetroffen	Keine. StackGres setzt envoy ein. Offensichtlich nicht bei einem ganzen Cluster
Failover	3	Geschwindigkeit	Der Failover muss so schnell stattfinden, dass offene Connections nicht wegen eines Timeouts geschlossen werden.	Nicht eingetroffen	Keine. StackGres setzt envoy ein. Offensichtlich nicht bei einem ganzen Cluster
Sharding und Datenintegrität	4	Datenkonsistenz	Daten sind konsistent und integer. Eingetroffen	Eingetroffen	
Sharding	5	Schutz vor Datenverlust	Die Daten müssen konsistent und schnell auf die Shards verteilt werden	Eingetroffen	
Self Healing	6	Node stellt sich selber wieder her	Shard Node wird automatisch synchronisiert	Eingetroffen	
Self Healing	7	Leader wird automatisch gesetzt	Leader wird entweder beibehalten oder wird neu gesetzt wenn ein Node zurückkehrt	Eingetroffen	

Tabelle 3.14: Testresultate Evaluation StackGres - Citus

Die genauen Details sind im Anhang zu finden: [Anhang - StackGres - Citus Testing](#)

### 3.1.8.3 YugabyteDB

Art	Test Case Nr.	Test Case	Erwartetes Ergebnis	Eingetretenes Ergebnis
Failover	1	Automatismus	Wird ein Node vom Netz genommen, muss es zu einem Rebalancing kommen	Eingetroffen
Failover	2	Connection-Stabilität	Bestehende Connections dürfen nicht getrennt werden.	Eingetroffen
Failover	3	Geschwindigkeit	Der Failover muss so schnell stattfinden, dass offene Connections nicht wegen eines Timeouts geschlossen werden.	Eingetroffen
Sharding und Datenintegrität	4	Datenkonsistenz	Daten sind Konsistent und Inetger. Eingetroffen	Eingetroffen
Sharding	5	Schutz vor Datenverlust	Die Daten müssen Konsistent und schnell auf die Tablets verteilt werden	Eingetroffen
Self Healing	6	Node stellt sich selber wieder her	Tablet wird automatisch synchronisiert	Eingetroffen

Tabelle 3.15: Testresultate Evaluation YugabyteDB

### 3.1.9 Gegenüberstellung der Lösungen

#### 3.1.9.0.1 YugabyteDB

Zuerst wurde für die Annährend 5GiB-DB initialisiert:

```
1 yugabyte=# create database pgbench_eval_bench;
2 CREATE DATABASE
```

Listing 3.8: YugabyteDB - Benchmarking - DB erstellen

#### 3.1.9.0.2 Patroni

Als die 250GiB DB getestet wurde, zeigte sich, das die Parameter nicht darauf optimiert waren.

#### 3.1.9.1 Benchmarks

Der vergleich zwischen den verschiedenen Varianten.

##### ☞ YugabyteDB

4GiB Memory für die tserver waren offensichtlich zu knapp bemessen. Zumindest wenn die Tabelle 120'000'000 Rows hat und ein mixed Benchmark abgesetzt wird.

Dies äusserte sich in einem Fehler (Absturz) auf zwei von drei tserver-Nodes sowie einer hohen Anzahl an Fehlern bei den mixed-Benchmarks. Daher wurde das Memory auf 8GiB erhöht und die komplette Testreihe erneut gestartet. Zudem wurden die Anzahl Fehler ebenfalls in die Benchmark-Auswertung einbezogen.

Bei den Transaktionen pro Sekunden gilt, je höher der Wert, umso besser das Ergebnis.

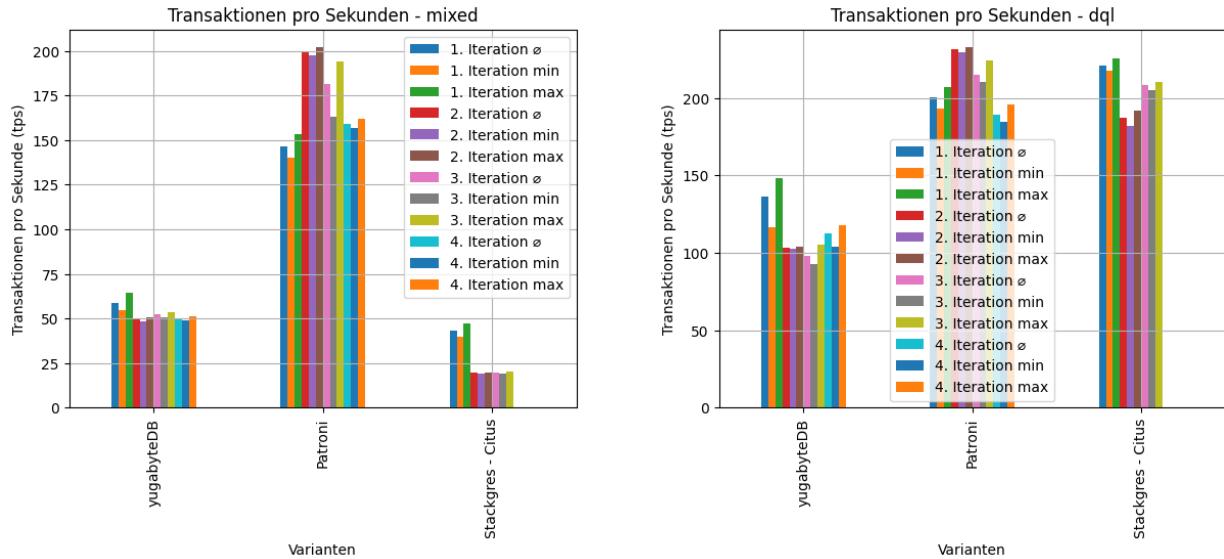


Abbildung 3.37: Benchmarks - tps

Bei der Latenz ist es genau andersrum, je höher der Wert desto schlechter schnitt die Variante ab.

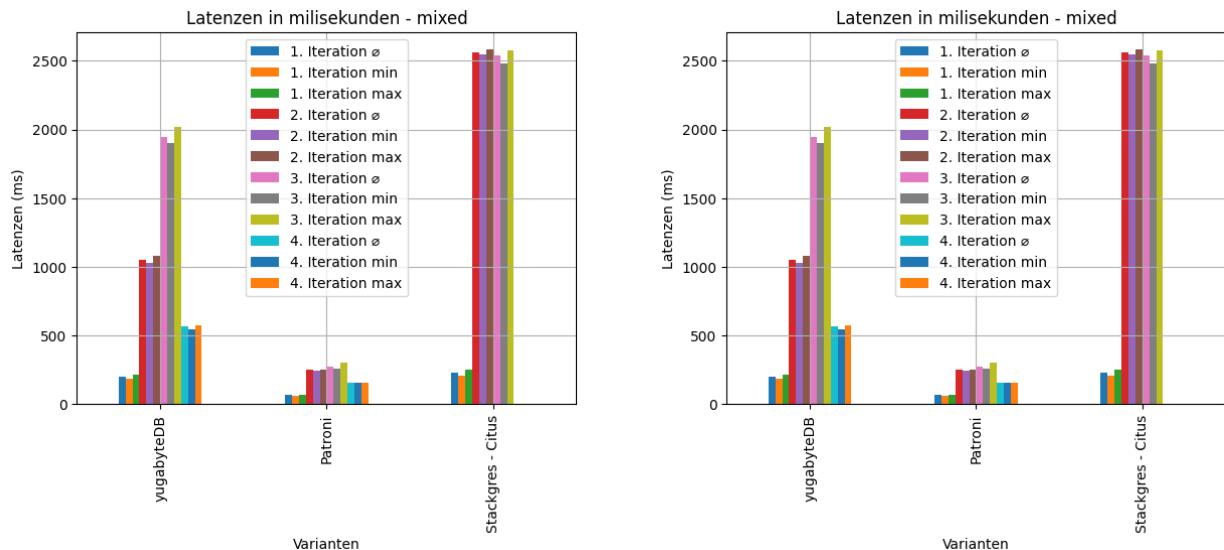


Abbildung 3.38: Benchmarks - latency

Die ersten beiden läufe mit Patroni wurde erst nur mit der Asynchronen Standard-Replikation von Patroni vorgenommen.

Später wurden die Benchmarks mit der Synchronen Replikation wiederholt.

Daraus ergab sich die Möglichkeit, beide Methoden direkt zu vergleichen:

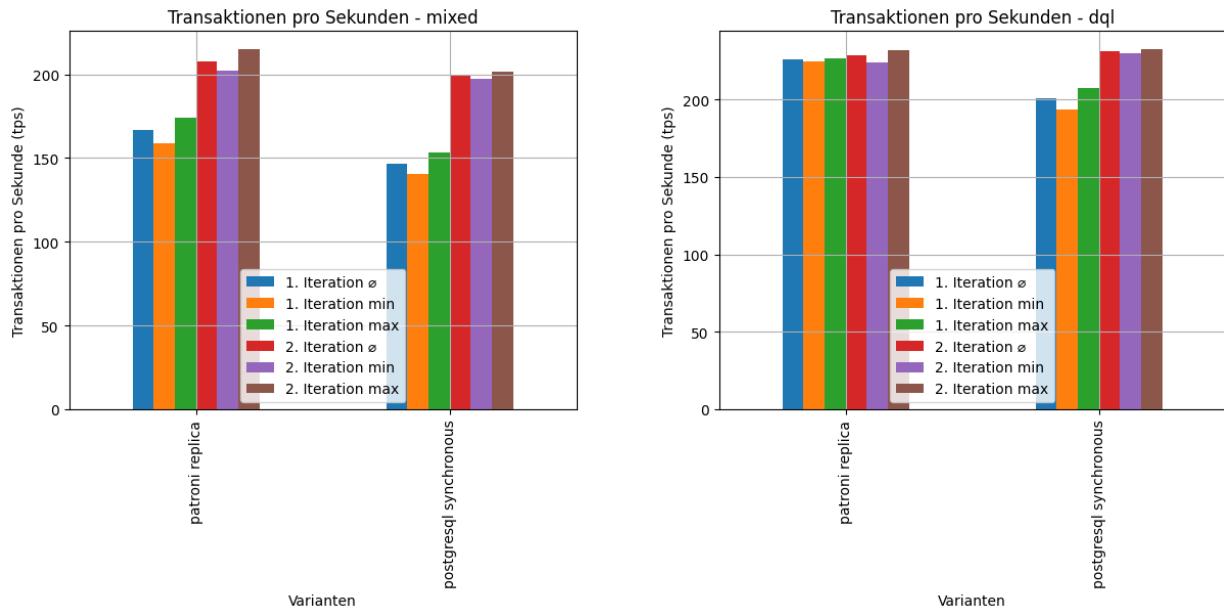


Abbildung 3.39: Benchmarks - tps Patroni Replica

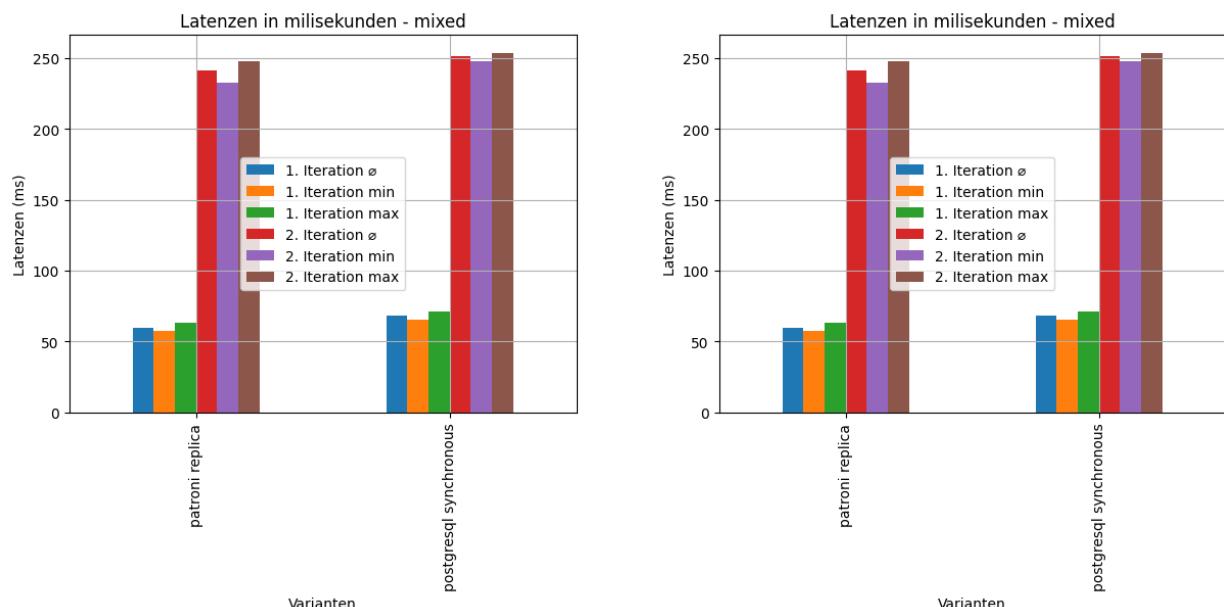


Abbildung 3.40: Benchmarks - latency Patroni Replica

Die Asynchrone Replikation ist dabei ein klein wenig schneller als die Synchrone Replikation.

Ein weiterer Benchmark sind die Fehler, die bei den DML-Transktionen beim mixed-Benchmark auftreten können.

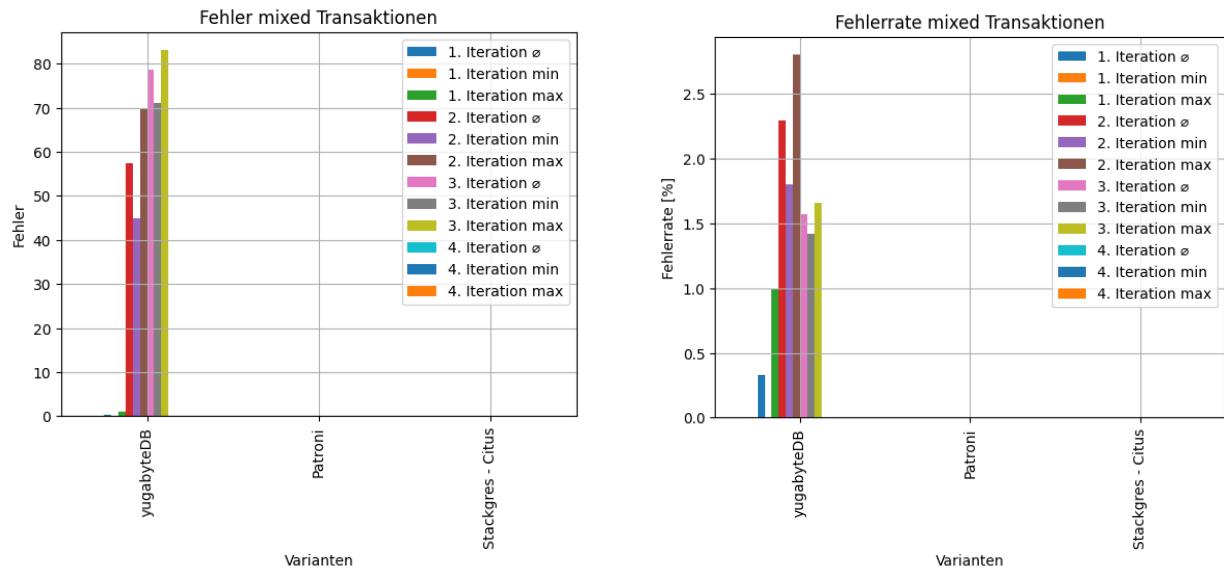


Abbildung 3.41: Benchmarks - Fehler bei mixed-Transaktionen

Ebenfalls ein wichtiger Benchmark ist die Zeit, die benötigt wird, um mittels pgbench initialisiert die Tabellen zu erstellen.

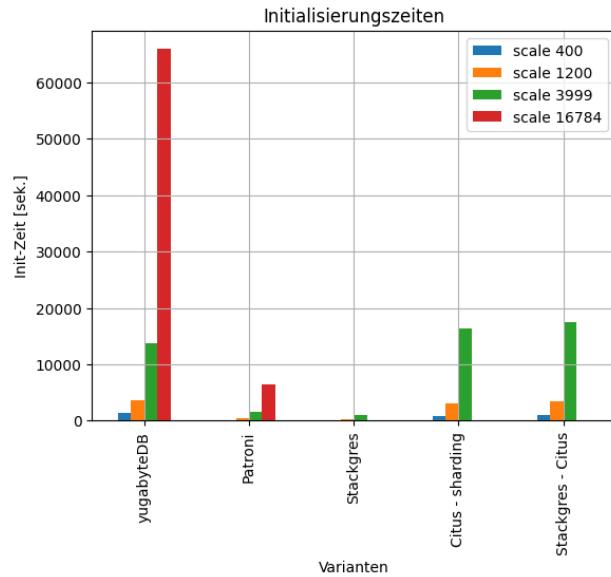


Abbildung 3.42: Benchmarks - Initialisierungszeit - sekunden

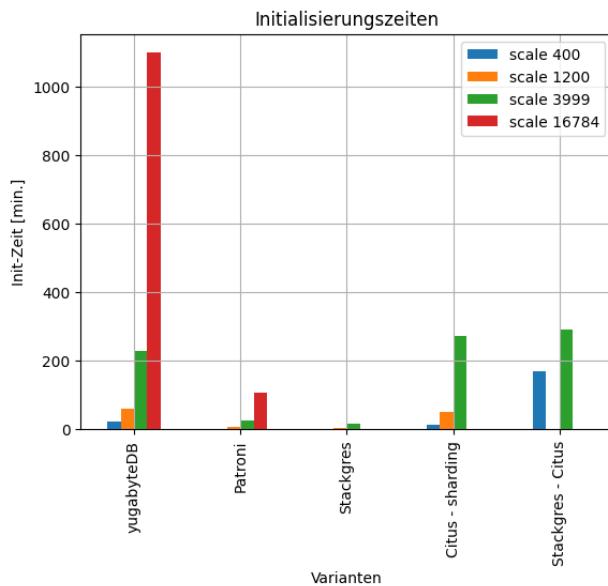


Abbildung 3.43: Benchmarks - Initialisierungszeit - Minuten

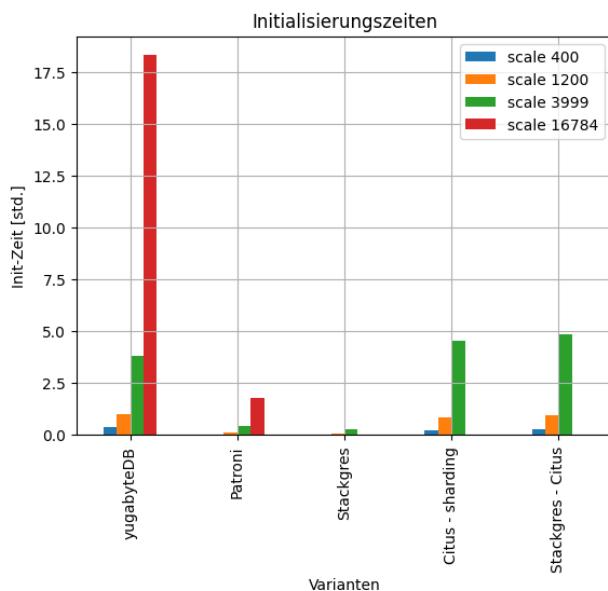


Abbildung 3.44: Benchmarks - Initialisierungszeit - Stunden

Dabei fällt auf, mit Patroni werden die Tabellen am schnellsten geladen.

StackGres selber generiert ebenfalls wesentlich schneller als YugabyteDB.

Werden dann aber die Tabellen in Shards aufgeteilt, verändert sich die Initialisierungszeit zuungunsten von StackGres - Citus.

### 3.1.9.2 Gemessene Zeiten und Kosten

## Diplomarbeit



### 3.1.9.2.1 Gegebene Parameter und Annahmen

### 3.1.9.2.2 Gemessene Zeiten

### 3.1.9.2.3 Zeitvergleiche

Phase	Subphase	Patroni - vanilla	Patroni - postgresql-cluster	StackGres - Citus	YugabyteDB
Initialer Aufwand	Basisinstallation	5.0	6.0	4.0	5.0
	Basiskonfiguration	5.0	5.0	5.0	5.0
	Backup Konfiguration	1.0	1.0	1.0	2.0
	Monitoring Konfiguration	2.0	2.0	2.0	2.0
Security Aufwand	private container registry Integration	0.0	0.0	1.0	1.0
	PKI Integration	3.0	3.0	3.0	3.0
Erweiterungsaufwand	Automatisierung Backup	1.0	1.0	4.0	4.0
	Automatisierung Skalierung	8.0	4.0	8.0	2.0
	Self-Healing	8.0	4.0	16.0	0.0
	Auto-Recovery	8.0	4.0	16.0	2.0
	DB Self-Service	16.0	16.0	16.0	16.0
Operationsaufwand / 5 Jahre	Switchover	50.0	25.0	50.0	0.0
	Node Recovery	50.0	25.0	100.0	25.0
	Backup Recovery	50.0	25.0	50.0	25.0
	Quorum erweitern	30.0	20.0	5.0	5.0
		237.0	141.0	281.0	97.0

Tabelle 3.16: Gemessene und Extrapolierte Aufwände Bsp.

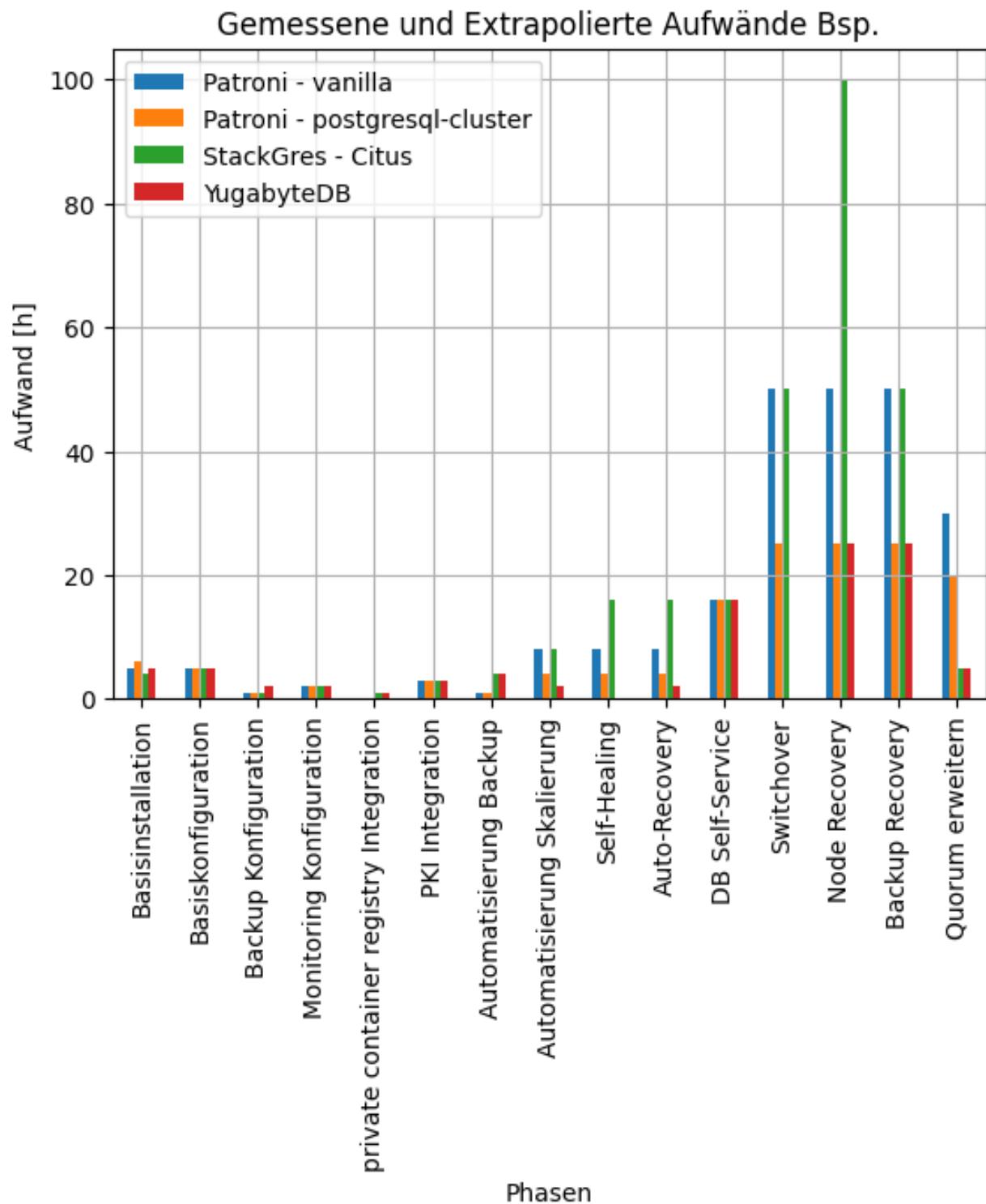


Abbildung 3.45: Zeitaufwände

## 3.1.9.2.4 Kostenvergleiche

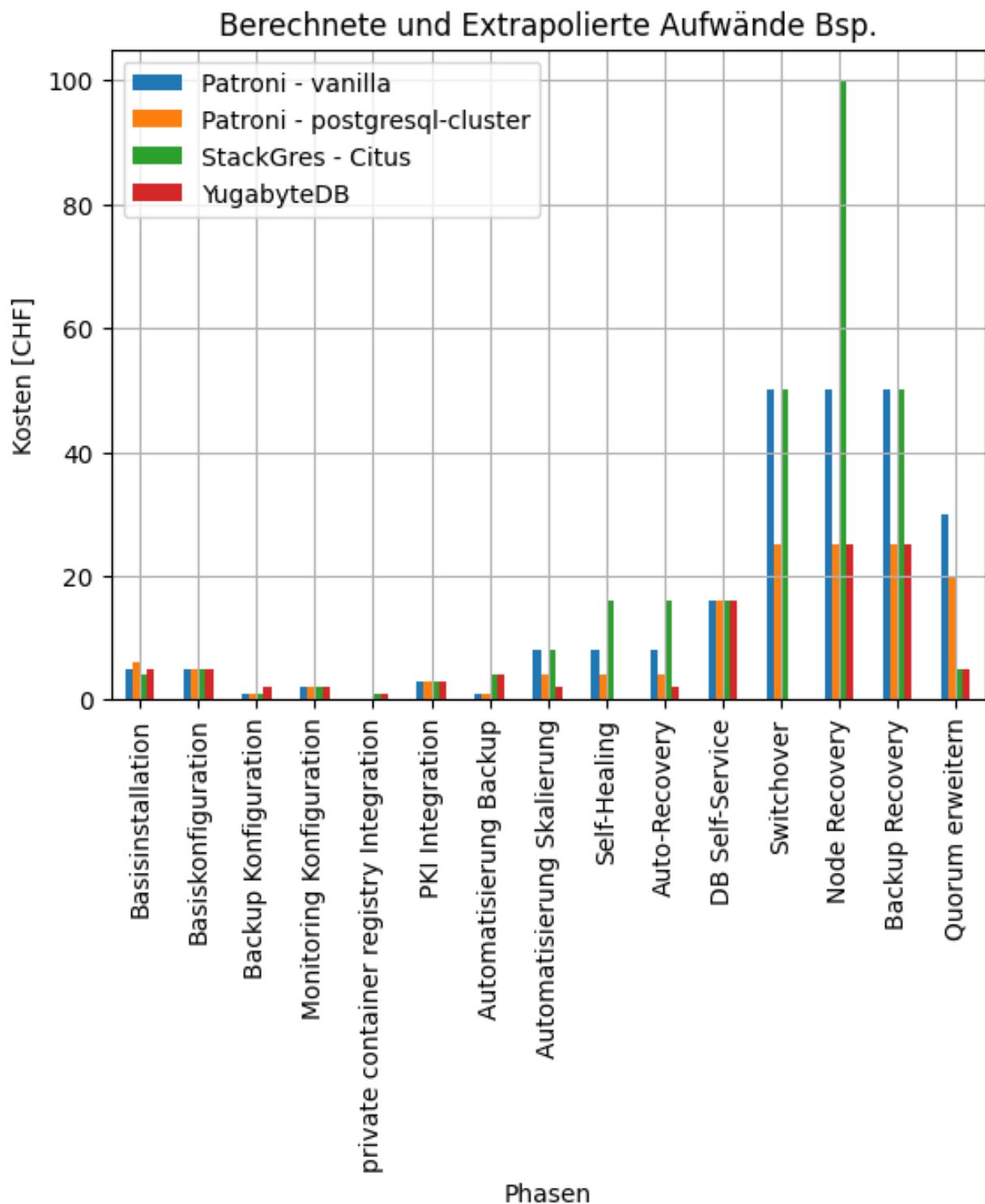


Abbildung 3.46: Kostenaufwände

### 3.1.9.3 Kosten-Nutzen

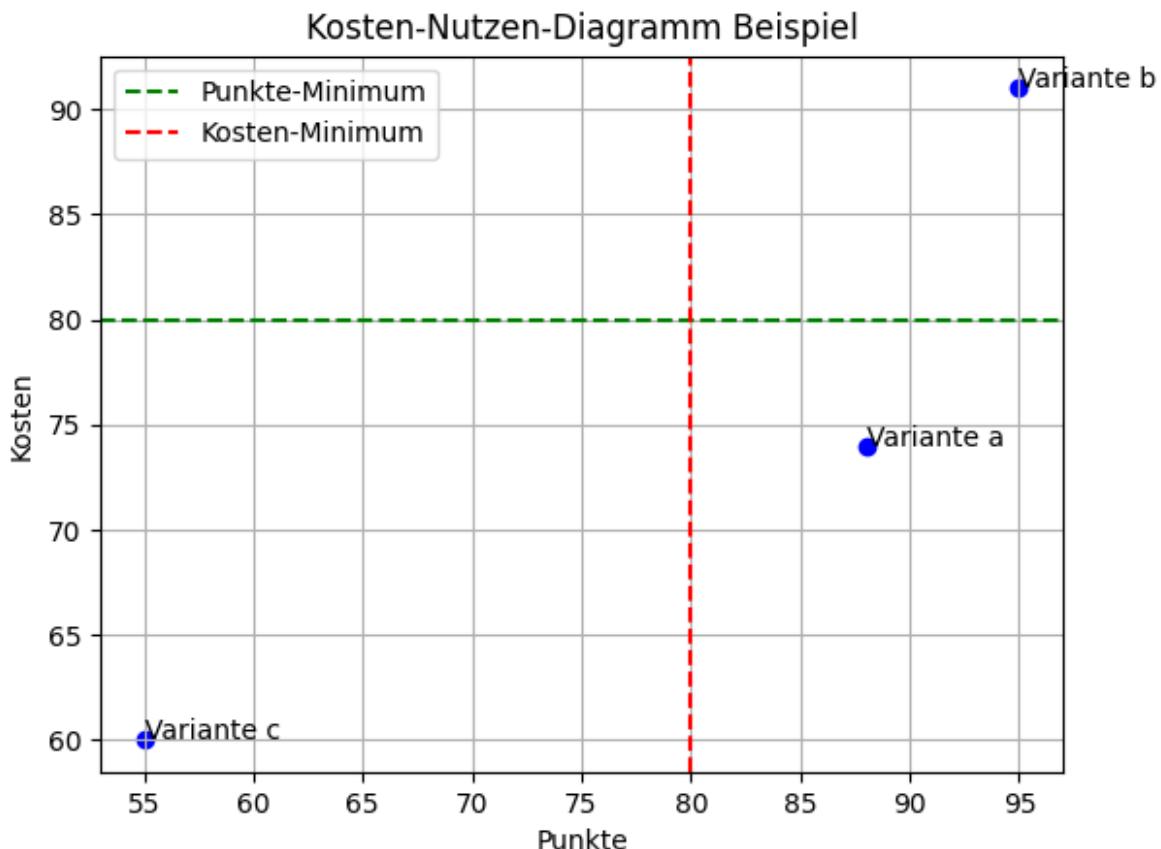


Abbildung 3.47: Kosten-Nutzen-Analyse

### 3.1.10 Entscheid

- 3.2 Aufbau und Implementation Testsystem
- 3.2.1 Bereitstellen der Grundinfrastruktur
- 3.2.2 Installation und Konfiguration PostgreSQL HA Cluster
- 3.2.3 Technical Review der Umgebung
- 3.3 Testing
- 3.3.1 Testing
- 3.3.2 Protokollierung

## Diplomarbeit

3.3.3      Review und Auswertung

3.4          Troubleshooting und Lösungsfindung

- 4           Resultate**
- 4.1       Zielüberprüfung**
- 4.2       Schlussfolgerung**
- 4.3       Weiteres Vorgehen / offene Arbeiten**
- 4.4       Persönliches Fazit**

## Abbildungsverzeichnis

1.1	Spitalregionen Kanton Graubünden[53]	1
1.2	Wahlkreise Kanton St. Gallen[78]	2
1.3	Spitalregionen / Spitalstrategie Kanton St. Gallen[47]	3
1.4	Organigramm Kantonsspital Graubünden	4
1.5	Organigramm Departement 10 - ICT	5
1.6	Datenbanken - Aufgeschlüsselt nach RDBMS	8
1.7	Datenbanken - Aufgeschlüsselt nach Betriebssystem	9
1.8	Risikomanagement PostgreSQL	13
1.9	Systemabgrenzung	18
2.1	Risikomanagement Projekt	23
2.2	Riskikomatrix - Assessment 21.03.2024	27
2.3	Projektcontrolling	29
3.1	Sharding - Vertikale Partitionierung	33
3.2	Sharding - Horizontales Partitionierung	34
3.3	Monolithische vs. verteilte SQL Systeme	36
3.4	CAP-Theorem	38
3.5	Datenbankskalierung	39
3.6	Präferenzmatrix	44
3.7	Testing - ERD DB self_healing_test	47
3.8	Benchmark Settings - Zabbix - Systeminformationen	48
3.9	Benchmark Settings - Zabbix - Connections per Seconds	49
3.10	Benchmark Settings - Zabbix - Queries per Seconds	49
3.11	Benchmark Settings - Zabbix - Client Queries per Seconds	49
3.12	Benchmark Settings - Zabbix - DB Size	50
3.13	Benchmark Settings - Anzahl Records / Skalierungsfaktor	52
3.14	pg_auto_failover-Architektur - Single Standby	56
3.15	pg_auto_failover-Architektur - Multi-Node Standby	56
3.16	pg_auto_failover-Architektur - Citus	57
3.17	CloudNativePG - Kubernetes - PostgreSQL	58
3.18	CloudNativePG - Kubernetes - Read-write workloads	59
3.19	CloudNativePG - Kubernetes - Read-only workloads	59
3.20	Patroni-Architektur	62
3.21	Stackgres - Grundarchitektur	65
3.22	Citus - Coordinator und Workers	66

## Diplomarbeit

3.23 Citus - Row-Based-Sharding . . . . .	66
3.24 Citus - Schema-Based-Sharding . . . . .	67
3.25 StackGres-Citus - Shard-Replikation . . . . .	68
3.26 YugabyteDB - Grundkonzept . . . . .	69
3.27 YugabyteDB - Architektur . . . . .	70
3.28 YugabyteDB - Sharding . . . . .	71
3.29 YugabyteDB - Tablet - Leader und Follower . . . . .	71
3.30 YugabyteDB - Tablet - Replikationsfaktor . . . . .	72
3.31 YugabyteDB - Zonen . . . . .	73
3.32 YugabyteDB - Zone outage Tolerance . . . . .	74
3.33 Evaluationssystem - Distributed SQL / Shards . . . . .	76
3.34 Patroni - Evaluationsarchitektur . . . . .	77
3.35 Stackgres - Citus - Evaluationsarchitektur . . . . .	78
3.36 YugabyteDB - Subscription Yugaware . . . . .	79
3.37 Benchmarks - tps . . . . .	85
3.38 Benchmarks - latency . . . . .	85
3.39 Benchmarks - tps Patroni Replica . . . . .	86
3.40 Benchmarks - latency Patroni Replica . . . . .	86
3.41 Benchmarks - Fehler bei mixed-Transaktionen . . . . .	87
3.42 Benchmarks - Initialisierungszeit - sekunden . . . . .	87
3.43 Benchmarks - Initialisierungszeit - minuten . . . . .	88
3.44 Benchmarks - Initialisierungszeit - stunden . . . . .	88
3.45 Zeitaufwände . . . . .	90
3.46 Kostenaufwände . . . . .	91
3.47 Kosten-Nutzen-Analyse . . . . .	92
I CloudNativePG - Pulse . . . . .	vi
II CloudNativePG - Code Frequency . . . . .	vi
III CloudNativePG - Community Standards . . . . .	vii
IV CloudNativePG - Contributors Commits . . . . .	vii
V CloudNativePG - Contributors Deletions . . . . .	viii
VI CloudNativePG - Contributors Additions . . . . .	viii
VII CloudNativePG - Commit Activity . . . . .	viii
VIII CloudNativePG - Network Graph . . . . .	ix
IX Patroni - Pulse . . . . .	ix
X Patroni - Code Frequency . . . . .	x
XI Patroni - Community Standards . . . . .	x
XII Patroni - Contributors Commits . . . . .	xi
XIII Patroni - Contributors Deletions . . . . .	xi

## Diplomarbeit

XIV	Patroni - Contributors Additions . . . . .	xi
XV	Patroni - Commit Activity . . . . .	xii
XVI	Patroni - Network Graph . . . . .	xii
XVII	Stackgres - Pulse . . . . .	xiii
XVIII	Citus - Pulse . . . . .	xiii
XIX	Stackgres - Code Frequency . . . . .	xiii
XX	Citus - Code Frequency . . . . .	xiv
XXI	Stackgres - Community Standards . . . . .	xiv
XXII	Citus - Community Standards . . . . .	xv
XXIII	Stackgres - Contributors Commits . . . . .	xv
XXIV	Stackgres - Contributors Deletations . . . . .	xvi
XXV	Stackgres - Contributors Additions . . . . .	xvi
XXVI	Citus - Contributors Commits . . . . .	xvi
XXVII	Citus - Contributors Deletions . . . . .	xvii
XXVIII	Citus - Contributors Additions . . . . .	xvii
XXIX	Stackgres - Commit Activity . . . . .	xvii
XXX	Citus - Commit Activity . . . . .	xviii
XXXI	Stackgres - Network Graph . . . . .	xviii
XXXII	Citus - Network Graph . . . . .	xix
XXXIII	MugabyteDB - Pulse . . . . .	xix
XXXIV	MugabyteDB - Code Frequency . . . . .	xx
XXXV	MugabyteDB - Community Standards . . . . .	xx
XXXVI	MugabyteDB - Contributors . . . . .	xxi
XXXVII	MugabyteDB - Commit Activity . . . . .	xxi
XXXVIII	MugabyteDB - Network Graph . . . . .	xxii
XXXIX	StackGres Testing - Node sks1184 down . . . . .	xlv
XL	StackGres Testing - Pods Down . . . . .	xlvi
XLI	StackGres Testing - Patroni Übersicht . . . . .	xlvi
XLII	StackGres Testing - DB Zugriff . . . . .	xlvii
XLIII	StackGres Testing - Connection Timeout . . . . .	xlviii

## Tabellenverzeichnis

1.1	Inventarisierte Datenbanksysteme . . . . .	7
1.2	Datenbankinventar . . . . .	8
1.3	Datenbankinventor - Nach Betriebssystemen aufgeschlüsselt . . . . .	9
1.4	Risiko-Matrix aktuelle Situation PostgreSQL Datenbanken . . . . .	12
1.5	Administrative Aufgaben . . . . .	14
1.6	Automatisierung Administrativer Aufgaben . . . . .	15
1.7	Ziele . . . . .	16
1.8	Gegebene Systeme . . . . .	17
1.9	Abhängigkeiten . . . . .	19
2.1	Risiko-Matrix der Diplomarbeit . . . . .	22
2.2	Neu Erkannte / Erfasste Risiken . . . . .	24
2.3	Risiko-Assessment 21.03.2024 . . . . .	26
2.4	Projektcontrolling . . . . .	28
2.5	Fachgespräche . . . . .	32
3.1	Quorum Beispiele . . . . .	37
3.2	Anforderungskatalog . . . . .	42
3.3	Stakeholder . . . . .	43
3.4	Benchmark Settings - Mixed Transaktionen . . . . .	50
3.5	Benchmark Settings - DQL Transaktionen . . . . .	50
3.6	Benchmark Settings - Skalierungsfaktoren . . . . .	51
3.7	Benchmark Settings - Datenbankgrößen / Skalierungsfaktor . . . . .	51
3.8	Benchmark Settings - Tabellengrößen / Skalierungsfaktor . . . . .	51
3.9	Vorauswahl - Ausgeschieden . . . . .	75
3.10	Vorauswahl - Evaluation . . . . .	75
3.11	Evaluationssystems . . . . .	76
3.12	Evaluationssystem - Distributed SQL / Sharding . . . . .	76
3.13	Testresultate Evaluation Patroni . . . . .	83
3.14	Testresultate Evaluation StackGres - Citus . . . . .	83
3.15	Testresultate Evaluation YugabyteDB . . . . .	84
3.16	Gemessene und Extrapolierte Aufwände Bsp. . . . .	89
I	Arbeitsrapport . . . . .	i
II	Fachgespräche - Protokoll . . . . .	ii
III	Kommentare - Anmerkung . . . . .	v

## Listings

3.1	metallb - Konfig YAML - Detail L2Advertisement . . . . .	79
3.2	yugabyteDB - Helm Chart Manifest - Detail Image . . . . .	80
3.3	yugabyteDB - Helm Chart Manifest - Detail StorageClass . . . . .	80
3.4	yugabyteDB - Helm Chart Manifest - Detail Resources . . . . .	80
3.5	yugabyteDB - Helm Chart Manifest - Detail Replica . . . . .	81
3.6	yugabyteDB - Helm Chart Manifest - Detail Disable YSQL . . . . .	81
3.7	yugabyteDB - Helm Chart Manifest - Detail Domainname und Service-Endpoints . . . . .	82
3.8	yugabyteDB - Benchmarking - DB erstellen . . . . .	84
1	Proxy Settings . . . . .	xxii
2	rke2 server - Verzeichnis erstellen . . . . .	xxiii
3	rke2 server - config.yaml . . . . .	xxiii
4	rke2 server - cilium-config.yaml . . . . .	xxiii
5	rke2 server installieren . . . . .	xxiii
6	rke2 agenten installieren . . . . .	xxiii
7	rke2 agent - config.yaml . . . . .	xxiv
8	-rke2 agent service restart . . . . .	xxiv
9	rke2 server proxy . . . . .	xxiv
10	rke2 server proxy kopieren . . . . .	xxiv
11	rke2 server cilium installieren . . . . .	xxiv
12	rke2 server cilium aktivieren . . . . .	xxiv
13	rke2 server starten . . . . .	xxiv
14	iptables entries server . . . . .	xxv
15	rke2 server token . . . . .	xxv
16	local-path-storage auf Linux Bereitstellen . . . . .	xxvi
17	local-path-provisioner definieren . . . . .	xxvi
18	local-path-storage aktualisieren . . . . .	xxvii
19	MetallB installieren . . . . .	xxvii
20	MetallB konfigurieren . . . . .	xxvii
21	MetallB Konfiguration einspielen . . . . .	xxvii
22	yugabyteDB - StorageClass setzen . . . . .	xxviii
23	yugabyteDB - StorageClass / PersistentVolume aktivieren . . . . .	xxviii
24	yugabyteDB - Namespace . . . . .	xxix
25	yugabyteDB - Helm Chart Manifest . . . . .	xxix
26	yugabyteDB - Installation . . . . .	xli

## Diplomarbeit

27	StackGres-Citus - StorageClass setzen . . . . .	xli
28	yugabyteDB - StorageClass / PersistentVolume aktivieren . . . . .	xlii
29	sks9016 - Download YugabyteDB On-Premise . . . . .	xlii
30	sks9016 - Installation YugabyteDB On-Premise . . . . .	xlii
31	sks9016 - Check YugabyteDB On-Premise . . . . .	xliii
32	Patroni - Proxy Settings . . . . .	xliii
33	Patroni - apt-Proxy Settings . . . . .	xliv
34	Patroni - PostgreSQL einbinden . . . . .	xliv
35	Python LaTex - zotero.py - Zotero BibLaTex Importer . . . . .	xlviii
36	Python LaTex - zotero_bibtex_configuration.yaml - Konfigurationsdatei - Zotero Bi- bLaTex Importer . . . . .	liv
37	Python LaTex - zotero_biblatex_keystore.yaml - x-y-Achse Konfigurationsdatei - Zo- tero BibLaTex Importer . . . . .	lv
38	Python LaTex - riskmatrix.py - Risikomatrizen . . . . .	lxii
39	Python LaTex - riskmatrix_plotter_conf.yaml - Konfigurationsdatei - Risikomatrizen	lxvi
40	Python LaTex - riskmatrix_xy_axis_tuple_matrix.yaml - Konfigurationsdatei - Risi- komatrizen - X-Y-Achsen Tuples . . . . .	lxix
41	Python LaTex - cost_benefit_diagram.py - Kosten-Nutzen-Diagramm . . . . .	lxx
42	Python LaTex - cost_benefit_diagram_plotter_conf.yaml - Konfigurationsdatei - Kosten- Nutzen-Diagramm . . . . .	lxxii
43	Python LaTex - pandas_dataframe_to_latex_table.py CSV - LaTex Tabelle . . . . .	lxxiii
44	Python LaTex - csv_to_latex_diplomarbeit.yaml - Konfigurationsdatei - CSV - LaTex- Tabelle . . . . .	lxxxii
45	Python LaTex - pandas_data_chart_plotter.py CSV - Diagramm . . . . .	cii
46	Python LaTex - pandas_data_chart_plotter_conf.yaml - Konfigurationsdatei - CSV - Diagramme . . . . .	cix

## Literatur

- [1] *About pgbench-tools.* <https://github.com/gregs1104/pgbench-tools>. original-date: 2010-02-17T13:33:28Z. 2023.
- [2] Satyadeep Ashwathnarayana und Inc. Netdata. *How to monitor and fix Database bloats in PostgreSQL? | Netdata Blog.* <https://blog.netdata.cloud/postgresql-database-bloat/>. 2022.
- [3] unknown author. *#1 Backup-Lösung für Kubernetes.* <https://www.veeam.com/de/kubernetes-native-backup.html?ck=1697900263871>.
- [4] unknown author. *API Reference - CloudNativePG.* <https://cloudnative-pg.io/documentation/1.22/cloudnative-pg.v1/>.
- [5] unknown author. *API reference (for YSQL and YCQL).* <https://docs.yugabyte.com/preview/api/>.
- [6] unknown author. *Architecture Basics — pg\_auto\_failover 2.0 documentation.* <https://pg-auto-failover.readthedocs.io/en/main/architecture.html>.
- [7] unknown author. *Benefits of using YugabyteDB.* <https://docs.yugabyte.com/preview/features/>. Section: preview.
- [8] unknown author. *Choosing Distribution Column - Citus 12.1 documentation.* [https://docs.citusdata.com/en/v12.1/sharding/data\\_modeling.html#distributed-data-modeling](https://docs.citusdata.com/en/v12.1/sharding/data_modeling.html#distributed-data-modeling).
- [9] unknown author. *Cilium - Cloud Native, eBPF-based Networking, Observability, and Security.* <https://cilium.io>.
- [10] unknown author. *Citus Replication Model: Today and Tomorrow - Replication Groups.* <https://www.citusdata.com/blog/2016/12/15/citus-replication-model-today-and-tomorrow/>.
- [11] unknown author. *Citus Support — pg\_auto\_failover 2.0 documentation.* <https://pg-auto-failover.readthedocs.io/en/main/citus.html>.
- [12] unknown author. *CLIs and command line tools.* <https://docs.yugabyte.com/preview/admin/>.
- [13] unknown author. *CloudNativePG - Main Features.* <https://cloudnative-pg.io/documentation/1.22/#main-features>.
- [14] unknown author. *Cluster Management - Citus 12.1 documentation - worker-node-failure.* [https://docs.citusdata.com/en/v12.1/admin\\_guide/cluster\\_management.html#worker-node-failure](https://docs.citusdata.com/en/v12.1/admin_guide/cluster_management.html#worker-node-failure).
- [15] unknown author. *Cluster Management — Citus Docs 7.2 documentation.* [https://docs.citusdata.com/en/v7.2/admin\\_guide/cluster\\_management.html](https://docs.citusdata.com/en/v7.2/admin_guide/cluster_management.html).

- [16] unknown author. *Concepts - Citus 12.1 documentation - row-based-sharding*. [https://docs.citusdata.com/en/v12.1/get\\_started/concepts.html#row-based-sharding](https://docs.citusdata.com/en/v12.1/get_started/concepts.html#row-based-sharding).
- [17] unknown author. *Dynamic Configuration Settings — Patroni 3.2.2 documentation*. [https://patroni.readthedocs.io/en/latest/dynamic\\_configuration.html](https://patroni.readthedocs.io/en/latest/dynamic_configuration.html).
- [18] unknown author. *EDB-Home*. <https://enterprisedb.com/>.
- [19] unknown author. *Envoy proxy - home*. <https://www.envoyproxy.io/>.
- [20] unknown author. *etcd*. <https://etcd.io/>.
- [21] unknown author. *Features - StackGres Documentation*. <https://stackgres.io/doc/latest/features/>.
- [22] unknown author. *HAProxy Documentation Converter*. <https://docs.haproxy.org/>.
- [23] unknown author. *HAProxy version 2.9.6 - Starter Guide*. <https://docs.haproxy.org/2.9/intro.html#3.2>.
- [24] unknown author. *Introduction | RKE2*. <https://docs.rke2.io/>. 2024.
- [25] unknown author. *Introduction to Cilium & Hubble — Cilium 1.15.3 documentation*. <https://docs.cilium.io/en/stable/overview/intro/#what-is-cilium>.
- [26] unknown author. *Manual Pages — pg\_auto\_failover 2.0 documentation*. <https://pg-auto-failover.readthedocs.io/en/main/ref/manual.html>.
- [27] unknown author. *MetallB provides Services with IP Addresses but doesn't ARP for the address · Issue #1154 · metallb/metallb*. <https://github.com/metallb/metallb/issues/1154>.
- [28] unknown author. *MetallB, bare metal load-balancer for Kubernetes*. <https://metallb.universe.tf/>.
- [29] unknown author. *Multi-node Architectures — pg\_auto\_failover 2.0 documentation*. <https://pg-auto-failover.readthedocs.io/en/main/architecture-multi-standby.html>.
- [30] unknown author. *Percona Software for PostgreSQL*. <https://www.percona.com/postgresql/software>.
- [31] unknown author. *PgBouncer - lightweight connection pooler for PostgreSQL*. <https://www.pgbouncer.org/>.
- [32] unknown author. *Replication in DocDB - Zone Fault Tolerance*. <https://docs.yugabyte.com/preview/architecture/docdb-replication/replication/>. Section: preview.
- [33] unknown author. *Support and Services for PostgreSQL*. <https://www.percona.com/postgresql/support-and-services>.
- [34] unknown author. *Tuning PostgreSQL with pgbench*. <https://www.cloudbees.com/blog/tuning-postgresql-with-pgbench>. 2017.

- [35] unknown author. *YB-TServer service*. <https://docs.yugabyte.com/preview/architecture/concepts/yb-tserver/>. Section: preview.
- [36] GitLab B.V. und GitLab Inc. *The DevSecOps Platform | GitLab*. <https://about.gitlab.com/>.
- [37] Fernando Laudares Camargos Avinash Vallarapu. *Tuning PostgreSQL for sysbench-tpcc*. <https://www.percona.com/blog/tuning-postgresql-for-sysbench-tpcc/>. 2018.
- [38] Alexandre Cassen und Read the Docs. *Introduction — Keepalived 1.2.15 documentation*. <https://keepalived.readthedocs.io/en/latest/introduction.html>. 2017.
- [39] Microsoft Corporation. *Azure SQL-Datenbank – ein verwalteter Clouddatenbankdienst | Microsoft Azure*. <https://azure.microsoft.com/de-de/products/azure-sql/database>. 2023.
- [40] Microsoft Corporation. *Datenbank-Software und Datenbankanwendungen | Microsoft Access*. <https://www.microsoft.com/de-de/microsoft-365/access>. 2023.
- [41] Microsoft Corporation. *Microsoft Data Platform | Microsoft*. <https://www.microsoft.com/de-ch/sql-server>.
- [42] Varun Dhawan und data-nerd.blog. *PostgreSQL-Diagnostic-Queries – data-nerd.blog*. <https://data-nerd.blog/2018/12/30/postgresql-diagnostic-queries/>.
- [43] Elektronik-Kompendium.de und Schnabel Schnabel. *SAN - Storage Area Network*. <https://www.elektronik-kompendium.de/sites/net/0906071.htm>. 2023.
- [44] DB-Engines und solidIT consulting & software development gmbh. *DB-Engines Ranking*. <https://db-engines.com/en/ranking>.
- [45] DB-Engines und solidIT consulting & software development gmbh. *relationale Datenbanken - DB-Engines Enzyklopädie*. <https://db-engines.com/de/article/relationale+Datenbanken?ref=RDBMS>.
- [46] The Linux Foundation. *Harbor*. <https://goharbor.io/>. 2023.
- [47] Kanton St. Gallen - Amt für Gesundheitsversorgung und Staatskanzlei Kanton St. Gallen - Dienststelle Kommunikation. *Weiterentwicklung der Strategie der St.Galler Spitalverbunde / sg.ch*. <https://www.sg.ch/gesundheit-soziales/gesundheit/gesundheitsversorgung--spitaeler-spitaeler-kliniken/spitalzukunft.html>.
- [48] Git. *About - Git*. <https://git-scm.com/about>.
- [49] IBM Deutschland GmbH. *Was ist das CAP-Theorem? | IBM*. <https://www.ibm.com/de-de/topics/cap-theorem>. 2023.
- [50] IBM Deutschland GmbH. *Was ist OLAP? | IBM*. <https://www.ibm.com/de-de/topics/olap>.
- [51] Jedox GmbH. *Was ist OLAP? Online Analytical Processing im Überblick*. <https://www.jedox.com/de/blog/was-ist-olap/>. Section: Knowledge.

## Diplomarbeit

- [52] Pure Storage Germany GmbH. *Was ist ein Storage Area Network (SAN)?* | Pure Storage. <https://www.purestorage.com/de/knowledge/what-is-storage-area-network.html>.
- [53] Gesundheitsamt Graubünden, Uffizi da sanadad dal Grischun und Ufficio dell'igiene pubblica dei Grigioni. *Kenndaten 2016 Spitäler und Kliniken September 2018*. <https://www.gr.ch/DE/institutionen/verwaltung/djsg/ga/InstitutionenGesundeitswesens/Spitaeler/Dok%20Spitler/Kenndaten%202016%20Spit%C3%A4ler.pdf>.
- [54] The Pgpool Global Development Group. *What is Pgpool-II?* <https://www.pgpool.net/docs/44/en/html/intro-whatis.html>. 2023.
- [55] The PostgreSQL Global Development Group. *25.1. Routine Vacuuming*. <https://www.postgresql.org/docs/16/routine-vacuuming.html>. 2023.
- [56] The PostgreSQL Global Development Group. *pgbench*. <https://www.postgresql.org/docs/16/pgbench.html>. 2023.
- [57] CYBERTEC Guest. *A formula to calculate pgbench scaling factor for target DB size*. <https://www.cybertec-postgresql.com/en/a-formula-to-calculate-pgbench-scaling-factor-for-target-db-size/>. 2018.
- [58] Michael Haag. *Built-in Connection Manager Turns Key PostgreSQL Weakness into a Strength*. <https://www.yugabyte.com/blog/connection-pooling-management/>. 2023.
- [59] Inc. HashiCorp. *Terraform by HashiCorp*. <https://www.terraform.io/>.
- [60] Patrick Hunt, Mahadev Konar, Flavio P Junqueira und Benjamin Reed. „ZooKeeper: Wait-free coordination for Internet-scale systems“. In: (2010).
- [61] Splunk Inc. *Splunk / Der Schlüssel zu einem resiliентen Unternehmen*. [https://www.splunk.com/de\\_de](https://www.splunk.com/de_de). 2023.
- [62] Sebastian Insaurti. *Scaling PostgreSQL for Large Amounts of Data*. <https://severalnines.com/blog/scaling-postgresql-large-amounts-data/>. 2019.
- [63] Shiv Iyer und MinervaDB. *PostgreSQL DBA Daily Checklist*. <https://minervadb.xyz/postgresql-dba-daily-checklist/>. 2020.
- [64] jobinau/pg\_gather. [https://github.com/jobinau/pg\\_gather](https://github.com/jobinau/pg_gather). original-date: 2021-01-19T08:12:07Z. 2024.
- [65] Unmesh Joshi. *Quorum*. <https://martinfowler.com/articles/patterns-of-distributed-systems/quorum.html>. 2020.
- [66] Martin Keen und IBM Deutschland GmbH. *IBM Db2*. <https://www.ibm.com/de-de/products/db2>.
- [67] Pasha Kostohrys. *Database replication — an overview*. <https://medium.com/@pkostohrys/database-replication-an-overview-f7ade110477>. 2020.
- [68] Anatoli Kreyman. *Was ist eigentlich Splunk?* <https://www.kreyman.de/index.php/splunk/76-was-ist-eigentlich-splunk-big-data-platform-monitoring-security>.

- [69] Pankaj Kushwaha und Unit 3D North Point House. *POSTGRESQL DATABASE MAINTENANCE. Routine backup of daily database... / by Pankaj kushwaha / Medium.* <https://pankajconnect.medium.com/postgresql-database-maintenance-66cd638d25ab>.
- [70] Red Hat Limited. *Was ist Ansible?* <https://www.redhat.com/de/technologies/management/ansible/what-is-ansible>.
- [71] Red Hat Limited. *Was ist CI/CD? Konzepte und CI/CD Tools im Überblick.* <https://www.redhat.com/de/topics/devops/what-is-ci-cd>.
- [72] Switzerland Linuxfabrik GmbH Zurich. *Keepalived — Open Source Admin-Handbuch der Linuxfabrik.* <https://docs.linuxfabrik.ch/software/keepalived.html>. 2023.
- [73] Nico Litzel, Stefan Luber und Vogel IT-Medien GmbH. *Was ist Elasticsearch?* <https://www.bigdata-insider.de/was-ist-elasticsearch-a-939625/>. 2020.
- [74] Hewlett Packard Enterprise Development LP. *Was ist SAN-Speicher? / Glossar.* <https://www.hpe.com/ch/de/what-is/san-storage.html>.
- [75] Julian Markwort. „Benchmarking four Different Replication Solutions“. In: ()..
- [76] Diego Ongaro. „Consensus: Bridging Theory and Practice“. In: (2014).
- [77] Bruno Queirós und LinkedIn Ireland Unlimited Company. *Postgresql replication with automatic failover.* <https://www.linkedin.com/pulse/postgresql-replication-automatic-failover-bruno-c3%b3s>. 2020.
- [78] Kanton St. Gallen - Dienst für politische Rechte und Staatskanzlei Kanton St. Gallen - Dienststelle Kommunikation. *Wahlkreise für Kantonsratswahlen / sg.ch.* <https://www.sg.ch/politik-verwaltung/abstimmungen-wahlen/wahlen/Wahlkreise-im-Kanton-SG.html>.
- [79] Ed Reckers und SnapLogic Inc. *Was ist die Snowflake-Datenplattform?* <https://www.snaplogic.com/de/blog/snowflake-data-platform>. 2023.
- [80] IONOS SE. *Apache Cassandra: Verteilte Verwaltung großer Datenbanken.* <https://www.ionos.de/digitalguide/hosting/hosting-technik/apache-cassandra-vorgestellt/>. 2021.
- [81] IONOS SE. *Datenbankmanagementsystem (DBMS) erklärt.* <https://www.ionos.de/digitalguide/hosting/hosting-technik/datenbankmanagementsystem-dbms-erklaert/>. 2020.
- [82] IONOS SE. *MongoDB – die flexible und skalierbare NoSQL-Datenbank.* <https://www.ionos.de/digitalguide/websites/web-entwicklung/mongodb-vorstellung-und-vergleich-mit-mysql>. 2019.
- [83] IONOS SE. *SQLite: Die bekannte Programmiersbibliothek im Detail vorgestellt.* <https://www.ionos.de/digitalguide/websites/web-entwicklung/sqlite/>. 2023.
- [84] IONOS SE. *Terraform.* <https://www.ionos.de/digitalguide/server/tools/was-ist-terraform/>. 2020.
- [85] IONOS SE. *Was ist Redis? Die Datenbank vorgestellt.* <https://www.ionos.de/digitalguide/hosting/hosting-technik/was-ist-redis/>. 2020.

- [86] IONOS SE. *Was ist SIEM (Security Information and Event Management)?* <https://www.ionos.de/digitalguide/server/sicherheit/was-ist-siem/>. 2020.
- [87] Naveed Shaikh. *It's All About Replication Lag in PostgreSQL.* <https://www.percona.com/blog/replication-lag-in-postgresql/>. 2023.
- [88] Sami Ahmed Siddiqui. *Distributed SQL 101.* <https://www.yugabyte.com/distributed-sql/>.
- [89] Inc. Snowflake. *Datenbanken, Tabellen und Ansichten – Überblick | Snowflake Documentation.* <https://docs.snowflake.com/de/guides-overview-db>.
- [90] Thomas-Krenn.AG. *Git Grundlagen – Thomas-Krenn-Wiki.* [https://www.thomas-krenn.com/de/wiki/Git\\_Grundlagen](https://www.thomas-krenn.com/de/wiki/Git_Grundlagen).
- [91] Mahmut Can Uçanefe. *Pgbench Load Test.* <https://medium.com/@c.ucanefe/pgbench-load-test-166b2023>.
- [92] vitabaks/postgresql\_cluster. [https://github.com/vitabaks/postgresql\\_cluster](https://github.com/vitabaks/postgresql_cluster). original-date: 2019-06-04T13:26:17Z. 2024.
- [93] Rainer Züst. „Einstieg ins Systems Engineering“. In: (2002).

## **Abkürzungen**

ICT	information and communications technology
ibW	ibW Höhere Fachschule Südostschweiz
KSGR	Kantonsspital Graubünden
KPS	KSGR Provisioning System
RDBMS	Relational Database Management System
DBMS	Database Management System
k8s	Kubernetes
HPE	Hewlett Packard Enterprise
HP-UX	Hewlett Packard UNIX
SAP	Systemanalyse Programmentwicklung
SQL	Structured Query Language
DBA	Database Administrator / Datenbankadministrator
HA	High Availability
PRTG	Paessler Router Traffic Grapher
SAN	Storage Area Network
SIEM	Security Information and Event Management
CI/CD	Continuous Integration/Continuous Delivery
SWOT-Analyse	Strengths, Weaknesses, Opportunities, Threats
OLAP	Online Analytical Processing
IaC	Infrastructure as Code
IPERKA	Informieren, Planen, Entscheiden, Realisieren, Kontrollieren, Auswerten
BSI	Bundesamt für Sicherheit in der Informationstechnik
VRRP	Virtual Router Redundancy Protocol

## Diplomarbeit

PKI	Private Key Infrastructure
DCS	Distributed Configuration Store
DQL	Data Query Language
DML	Data Manipulation Language
ACID	Atomicity, Consistency, Isolation und Durability
EDB	EnterpriseDB
CRD	Custom Resource Definition
rke2	Ranger Kubernetes Engine 2
BGP	Border Gateway Protocol
NTP	Network Time Protocol
BSD	Berkeley Software Distribution

## Glossar

**Ansible** Ansible ist ein Open-Source Automatisierungstool zur Provisionierung, Konfiguration, Deployment und Orchestrierung. Ansible verbindet sich auf die Zielgeräte und führt dort die hinterlegten Module aus. Oft werden die verschiedenen Aufgaben in einem Skript, in einem sogenannten Playbook geschrieben werden[70].. 17

**AUTOVACUUM** Der AUTOVACUUM Job räumt die Tablespaces und Data Files innerhalb von PostgreSQL sowie auf dem Filesystem nach Lösch- und Manipulations-Transaktionen auf, aktualisiert Datenbank interne Statistiken und verhindert Datenverlust von selten genutzten Datensätzen[55].. 15, 16, 47

**Cassandra** Cassandra ist eine Spaltenorganisierte NoSQL-Datenbank die 2008 veröffentlicht[80] wurde.. 7, 69, 75

**CI/CD** Continuous Integration/Continuous Delivery bedeutet, dass Anpassungen kontinuierlich in die Entwicklungsumgebungen integriert und auf die Zielplattformen verteilt werden[71].. 107

**Cilium** Cilium ist ein Netzwerk-Connector zwischen den Services von Container-Applikationen und Container Management-Systemen wie Docker oder k8s. Nebst simplen Networking bietet Cilium auch Netzwerk-Policies, Load-Balancer und andere Features[9, 25].. 79, 80

**DBMS** Ein Database Management System regelt und organisiert die Datenbasis einer Datenbank[81].. 107

**DCS** Der DCS ist eine Kernkomponente von Patroni [17]. Realisiert wird der DCS bei Patroni mit Etcd.. 108

**Debian** Debian gehört nebst Slackware Linux zu den ältesten Linux Distribution die noch immer gepflegt und eingesetzt werden. Sie wurde im August 1993 gestartet und brachte im Laufe der Zeit einige der beliebtesten Distributionen wie Ubuntu hervor.. 17

**Elasticsearch** Elasticsearch ist eine 2010 veröffentlichte Open-Source Suchmaschine die auf Basis von JSON-Dokumenten und einer NoSQL-Datenbank arbeitet[73].. 7

**etcd** etcd ist [20]. 61, 109, xliv

**Failover** In einem Fehlerfall wird in einem HA-System meist ein Primary Node auf den Secondary ungeplant geswitched.. 16, 36, 37, 53, 110

**Foreman** Foreman ist ein Lifecycle Management und Provisioning System für Virtuelle und Physische Server. Ab Version 6 basierte der Red Hat Satellite auf Foreman. 17, 22, xlvi

## Diplomarbeit

**Git** Git ist eine Versionierungssoftware und bietet die Möglichkeit, Repositories erstellen zu können. Die Repositories sind dabei nicht zentral sondern dezentral organisiert und arbeiten daher mit Working Copies von Repositories[48, 90].. 110

**GitHub** GitHub ist ein Git basierendes System für die Versionierung und bietet dabei auch noch Dienste für CI/CD. GitHub, dass mittlerweile zum Microsoft-Konzern gehört, kann sowohl als Online Dienst als auch als On-premises Service konsumiert werden. Hierfür ist der GitHub Enterprise Server notwendig[R398TJSHB, UL2FJNU].. xliv

**GitLab** GitLab ist ein Git basierendes System für die Versionierung und bietet dabei auch noch Dienste für CI/CD. GitLab kann sowohl als Online Dienst als auch als On-premises Service konsumiert werden[36].. 16, 53

**HAProxy** HAProxy [23]. 55, 60

**Harbor** Harbor ist ein Open-Source-Tool zur Registrierung von Richtlinien rollenbasierten Zugriffssteuerung[46]. Harbor wird beim KSGR zur Verwaltung der Kubernetes-Plattform verwendet.. 16, 53

**HP-UX** Dieses UNIX-Derivat ist ein abkömmling von System III, System V R3 und System V R4 und wurde von HP zum ersten Mal 1982 veröffentlicht.. 5, 9, 22, 107

**IBM DB2** IBM DB2 ist eine Relationale Datenbank[66] deren Vorläufer System-R von IBM zwischen 1975 und 1979 entwickelt wurde. DB2 selber wurde 1983 von IBM veröffentlicht.. 7, 38

**keepalived** keepalived nutzt VRRP um eine leichtgewichtige Lösung für ein HA-Failover zu realisieren. keepalived benötigt dazu keinen dritten Node, also einen Quorum-Node. Wenn die definierte sekundärseite keine Antwort mehr von der primären Seite nach einer definierten Anzahl versuchen in einem bestimmten Interval mehr bekommt, oder ein per Skript definiertes Event auf der primären Seite eintrifft, wird ein Failover auf die sekundäre Seite ausgeführt. Je nach Konfiguration kann der Restore auf die primäre Seite eingeleitet werden wenn diese wieder verfügbar ist oder der Restore unterbunden werden[72, 38].. 53

**Key-Value-orientierte** Siehe Key-Value-Datenbank. 113

**Key-Value-Datenbank** Eine Key-Value-Datenbank ist ein Typ derNoSQL Datenbanken. Diese Datenbanken haben einen Primary Key und oft mindestens einen Sort Key. Key-Value-Datenbanken können auch Objekte mit Subitems resp. Referenzen dazu speichern. Eine bekannte Key-Value-Datenbank ist Redis. 110, 112

**Key-Value-Store** Siehe Key-Value-Datenbank. 51, 69

## Diplomarbeit

**Kubernetes** Kubernetes, oder k8s, ist eine Open-Source Containerplattform die ursprünglich von Google 2014 für die Bereitstellung und Orchestrierung von Containern entwickelt wurde aber 2015 an eine Tochter Foundation der Linux Foundation gespendet. Kubernetes kommt aus dem Griechischen und bedeutet Steuermann.. 9, 17, 83, 107, 110

**Linux** Linux ist ein Open-Source Betriebssystem, welches von Linus Torvalds 1991 in seiner frühesten Form entwickelt wurde und löse vom UNIX Derivat MINIX inspiriert war. Linux besteht heute aus einer enorm grossen Anzahl an Distributionen und läuft auf einer grossen Anzahl von Plattformen.. 5, 111

**MariaDB** MariaDB ist ein MySQL Fork des ehemaligen MySQL Mitbegründers Michael Widenius, wobei sich der Name Maria aus dem Vornamen einer seiner Töchter ableitet. Nach dem Fork 2009 blieb MariaDB für eine Zeitlang sehr ähnlich mit MySQL und behielt ein ähnliches Versionierungsschema bei. Dies änderte sich 2012 wo dann direkt mit der Version 10 weitergefahren wurde. Beide Datenbanken entfernen sich im Lauf der Zeit immer mehr voneinander und sind nicht mehr in jedem Fall kompatibel oder beliebig austauschbar. Auf den Linux Distributionen trat MariaDB die Nachfolge von MySQL als Standard Datenbank an.. 5, 7, 9, 52

**MetallLB** MetallLB ist ein für Bare-Metal k8s Systeme ausgelegter Load-Balancer. Er kann sowohl auf Layer 2, mit OS-Boardmitteln arbeiten, bietet aber auch BGP-Routing an, so dass Pods direkt von Routern angesteuert werden können, ohne via Host gehen zu müssen[28].. 79

**Microsoft Azure SQL Database** Microsoft Azure SQL Database oder auch Azure SQL ist eine Relationale Datenbank die von Microsoft für die Azure Cloud optimiert 2010 Entwickelt wurde[39].. 7

**Microsoft Access** Access wurde 1992 veröffentlicht und ist Entwicklungsumgebung, Front- und Backend-Software und Relationale Datenbank in einem[40].. 7

**Microsoft SQL Server** MS SQL Server ist das RDBMS von Microsoft[41]. Nebst Microsoft Windows und Windows Server lässt es sich seit Version 2014 ebenfalls auf Linux Betreiben. In der Wirtschaft ist die primäre Plattform aber Windows Server.. 5, 7, 112

**MongoDB** MongoDB ist eine dokumentenorientierte NoSQL-Datenbank, die zum ersten Mal 2007 veröffentlicht wurde[82].. 7

**MySQL** Die Datenbank MySQL wurde ursprünglich als reine Relationale Open-Source Datenbank von Firma MySQL AB 1994 entwickelt. Der Name My leitet sich vom Namen My der Tochter des Mitbegründers Michael Widenius ab. Als Sun Microsystems 2008 MySQL übernahm, hielt sich die Option frei, bei einem Kauf von Sun Microsystems durch Oracle gründen zu dürfen. Seit Oracle Sun Microsystems 2010 gekauft hat, wurden immer mehr Funktionalitäten von der Community Edition zu der Enterprise Edition verschoben worden. Aus diesem

## Diplomarbeit

Grund hat heute der MySQL Fork MariaDB MySQL mehrheitlich aus allen Linux Distributionen als Standard Datenbank verdrängt.. 5, 7, 9, 52

**NoSQL** NoSQL steht für Not only SQL. Das heisst, Relationale Datenbanken haben Komponenten wie Dokumentendatenbanken, Graphendatenbanken, Key-Value-Datenbanken und Spaltenorientiert Datenbanken. Viele der grossen Datenbanklösungen wie Oracle Database oder Microsoft SQL Server sind NoSQL Datenbanken resp. bieten diese option an.. 7, 109, 110, 111, 113, 114

**OLAP** Eine Online Analytical Processing, kurz OLAP, ist eine Multirelationale resp. Multidimensionale Datenbanklösung. Sie wird oft in Form eines Datenwürfels erklärt, kann aber auf verschiedene Arten umgesetzt werden[51, 50]. OLAP-Systeme bieten eine Hochperformante Analyse grosser Datenmengen und sind oftmals zentraler Teil eines Data-Warehouses.. 7, 107

**Oracle Linux** Oracle Linux ist eine RHEL-Distribution der Firma Oracle und ist mit RHL Binär-kompatibel. Sie wird primär für den Betrieb von Oracle Datenbanken verwendet und kommt auf den Oracle Eigenen Appliances ODA und Exadata zum Einsatz. Für den Zweck als DB Plattform kann ein für Oracle Datenbanken optimimierter Kernel verwendet werden. Zu Oracle Linux kann ein kostenpflichtiger Support bezogen werden, allerdings ist die Distribution anders als RHEL auch ohne Lizenz erhältlich.. 17

**Oracle Database** Die erste verfügbare Version der Oracle Datenbank kam im Jahr 1979 mit Version 2 (statt Version 1) heraus, damals allerdings nur mit den Basisfunktionen. Im Laufe der Zeit wuchs der Funktionsumfang sehr stark an, die Grundlage des Client-Server-Designs kam erstmals im Jahr 1985 mit Version auf den Markt und hat sich im Prinzip bis heute gehalten. Mit der mit Version 8/8i 1997 erschienen Optimizer und mit der Version 9i 2001 erschienene Flashback-Funktionalität (die ein schnelles Online Recovery sowie einen Blick in die Vergangenheit ermöglichen) konnte Oracle sich stark von der Konkurrenz absetzen. Heute gilt die Datenbank als erste Wahl, wenn es um Hochverfügbare Systeme, hohe Performance oder grosse Datenmengen geht.. 5, 7, 9, 38, 112

**PKI** . 108

**PostgreSQL** Die OpenSource Datenbank PostgreSQL wurde in Form von POSTGRES zum ersten Mal 1986 von der University of California at Berkeley veröffentlicht. und zählt zu den beliebtesten OpenSource Datenbanken. Zudem besteht in vielen bereichen eine gewisse Ähnlichkeit zu Oracles Oracle Database.. 5, 7, 9, 10, 14, 38, 53, 68

**PostgreSQL HA Cluster** Der HA Cluster des PostgreSQL Clusters. 16

**PostgreSQL Cluster** Ein PostgreSQL Cluster entspricht einer Instanz bei MS SQL oder einer Container Database wie Oracle.. 15, 16, 53, 112

## Diplomarbeit

**PRTG** Das Monitoring System Paessler Router Traffic Grapher der Firma Paessler wurde 2003 zum erstmals veröffentlicht und war ebenfalls als Netzwerkmonitoring System konzipiert. Wie bei Zabbix lässt sich heute damit ebenfalls fast jedes IT-System damit Überwachen. Reichen die Zahlreich vorhanden Standard Sensoren nicht, können eigene Sensoren geschrieben werden. PRTG ist nicht Open-Source, man bezahlt anhand gewisser Sensor Packages.. 5, 15, 17, 107

**Quorum** In verteilten Systemen resp. Cluster muss sichergestellt werden, das bei einem Ausfall oder ein Netzwerk trennung zwischen den Nodes es zu keiner Split-brain-Situation kommt. Hierzu wird i.d.R. ein Quorum verwendet. I.d.R. wird jener Teil des Quorums zum Primary oder alleinigen Node, der mit der die Mehrheit aller Nodes vereint. Daraus ergeben sich bestimmte größen, mit 5 Nodes braucht es 3 Nodes um aktiv zu bleiben und mit 3 Nodes deren 2. Bei diesen Konstellationen wird daher darauf geachtet, eine ungerade Anzahl Nodes im Cluster zu halten um keine Pat-Situation zu provozieren. Im Kapitel [Unterabschnitt 3.1.1.4](#) wird genauer auf die Mechanik eines Quorums eingegangen. . 54, 110

**RDBMS** Ein RDBMS ist ein Datenbankmanagementsystem für eine Relationale Datenbank. Relationale Datenbanken sind Tabellenorganierte Datenmodelle die auf Relationen aufbauen, deren Schematas sich Normalisieren lassen. Dabei müssen Relationale Datenbanken müssen dabei auch Mengenoperationen, Selektion, Projektion und Joins erfüllen um als Relationale Datenbanken zu gelten[45].. 69, 107

**RedHat Enterprise Linux (RHEL)** RHEL wurde in seiner Ursprüglichen Form Red Hat Linux (RHL) bis in den Oktober 1994 zurück, wobei die erste Version von RHEL wie es heute existiert im Jahr 2002 erfolgte. RHEL ist auf lange Wartungszyklen von fünf Jahren und grosskunden ausgelegt. Ohne entsprechenden Supportvertrag kann keine ISO-Datei bezogen werden. Somit hebt sich RHEL stark von anderen Linux Distributionen ab.. 17

**Redis** Redis ist eine Key-Value-orientierte NoSQL In-Memory-Datenbank, dh. die Daten liegen Primär im Memory und nicht auf dem Storage[85]. Redis wurde 2009 zum ersten Mal veröffentlicht.. 7, 110

**rke2** rke2 ist eine Leichtgewichtige Kubernetes Distribution, die alles notwendige mitbringt, um einen k8s Cluster zu betreiben[24].. 76, 79

**Rocky Linux** Rocky Linux basierte auf der offen zugänglichen Linux Distribution CentOS welche RHEL Binärkompatibel war und gilt als inoffizieller Nachfolger von CentOS.. 17

**SAN** Ein Storage Area Network ist ein dediziertes Netzwerk aus Storage Komponenten. SAN Systeme bieten redundante Pools an Speicher. Die Physischen Festplatten werden zu Virtuellen Lunes, also logischen Einheiten, zusammengefasst. Dies werden nach aussen den Konsumenten präsentiert[43, 74, 52]. 5, 17, 22, 107

## Diplomarbeit

**SIEM** Ein sammelt Daten aus verschiedenen Netzwerkkomponenten oder Geräten von Agents oder Logs. Diese Daten werden permanent analysiert und mit einem definierten Regelwerk gegeprüft. Ziel ist es, verdächtige Events zu erkennen und einem Angriff zuvorzukommen oder ihn möglichst früh zu unterbinden[86].. 17, 107

**Snowflake** Snowflake ist eine Big Data Plattform die Data Warehousing, Data Lakes, Data Engineering und Data Science in einem Service vereint. Die Daten werden in eigenen internen Relationalen und NoSQL-Datenbanken gespeichert[89, 79]. 7

**Split-brain** Im Kapitel ?? werden die ursachen und folgenden eines Split-brains genauer besprochen. . 37, 113

**Splunk** Splunk ist Big Data Plattform, Monitoring- und Security-Tool in einem[61, 68]. . 7

**SQLite** SQLite ist eine Relationale Embedded Datenbank welche seit 2000 existiert. Sie verzichtet auf eine Client-Server-Architektur und kann in vielen Frameworks eingebunden werden[83].. 7

**Switchover** In einem Maintenance-Fall in einem HA-System meist ein Primary Node auf den Secondary geplant geswitched.. 16

**SWOT-Analyse** Eine SWOT-Analyse soll die Stärken (Strengths), Schwächen (Weaknesses), Chancen (Opportunities) und Risiken (Threads) für ein Unternehmen oder ein Projekt aufzueigen. Anhand einer SWOT-Analyse werden i.d.R. anschliessend Strategien abgeleitet um mit den Stärken und Chancen die Schwächen und Risiken abzufangen oder anzumildern.. 107

**Terraform** Terraform ist ein Werkzeug für die Verwaltung von Infrastruktur mit Software zu steuern, sogenanntes Infrastructure as Code. Terraform wird sehr oft dafür benutzt um Container- und Cloudinfrastruktur ansteuern und verwalten zu können[84, 59].. 17

**Transaktion** Eine Transaktion ist beinhaltet Schreib-, Lese-, Mutatations- oder Löschoperationen auf Daten.. 48, 50

**UNIX** Die erste Version von UNIX wurde im Jahr 1969 in den Bell Labs entwickelt und übernahm viele Komponenten aus dem gescheiterten Multics-Projekt. Aus dem Ursprünglichen UNIX enstanden im Laufe der Zeit viele offene und Proprietäre Derivate deren Einfluss weit über die Welt der Informatik reicht.. 107

**VRRP** VRRP . 107, 110

**Zabbix** Das 2001 veröffentlichte Open-Source Monitoring System Zabbix gilt zwar als Netzwerk-Monitoring System, allerdings kann heute nahezu jedes IT-System damit überwacht werden.

## Diplomarbeit

Zabbix speichert die Metriken und nicht die Auswertungen, das heisst, solange die Daten vorhanden sind können Grafiken zu jedem Zeitpunkt generiert werden. Zabbix ist grundsätzlich Open-Source, man kann allerdings Supportverträge Abschliessen.. 9, 17

## **Selbstständigkeitserklärung**

Ich versichere, dass die vorliegende Arbeit von den Autoren selbstständig und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt wurde. Alle Inhalte dieser Arbeit, dazu gehören neben Texten auch Grafiken, Programmcode, etc., die wörtlich oder sinngemäß aus anderen Quellen stammen, sind als solche eindeutig kenntlich gemacht und korrekt im Quellenverzeichnis gelistet. Dies gilt auch für einzelne Auszüge aus fremden Quellen.

Die Arbeit ist in gleicher oder ähnlicher Form noch nicht veröffentlicht und noch keiner Prüfungsbehörde vorgelegt worden.

---

Ort, Datum, Unterschrift

## **Haftungsausschluss**

Der vorliegende Bericht wurde von Studierenden im Rahmen einer Diplomarbeit erarbeitet. Es muss an dieser Stelle darauf hingewiesen werden, dass die Arbeit nicht im Rahmen eines Auftragsverhältnisses erstellt wurde. Weder der Ersteller noch die ibW Höhere Fachhochschule Südostschweiz können deshalb für Aktivitäten auf der Basis dieser Diplomarbeit eine Haftung übernehmen.

## I

## Arbeitsrapport

Datum	Von	Bis	Dauer [h]	Phase	Subphase	Tätigkeit	Bemerkung	Schwierigkeit	Lösungen
14.02.2024	19:00	20:00	1.0	1. Expertengespräch	1. Expertengespräch				
21.02.2024	15:00	16:00	1.0	Evaluation	Anforderungskatalog	Anforderungskatalog erarbeiten			
22.02.2024	16:00	17:30	1.5	Evaluation	Anforderungskatalog	Anforderungskatalog erarbeiten			
27.02.2024	10:00	11:30	1.5	Dokumentation	Dokumentation	Dokumentation erweitern			
27.02.2024	13:00	16:00	3.0	Dokumentation	Dokumentation	Dokumentation erweitern	Viele LaTeX Tabellen.		Generator mit python pandas gebaut für alle möglichen Tabellen. Inkl. Aggregation und Pivot-Mechaniken
28.02.2024	09:00	11:00	2.0	Dokumentation	Dokumentation	Dokumentation erweitern	Viele LaTeX Tabellen.		Generator mit python pandas gebaut für alle möglichen Tabellen. Inkl. Aggregation und Pivot-Mechaniken
01.03.2024	07:00	09:00	2.0	Dokumentation	Dokumentation	Dokumentation Exkurs Architektur	Um Entscheidungen Transparent zu machen, müssen Grundlegende Konzepte aufgezeigt werden. Nicht alle Konzepte wie z.B. Distributed SQL sind bekannt resp. das zusammenspiel mit Kubernetes.	Konzepte wie Distributed SQL sind nicht einfach zu erklären.	
08.03.2024	07:00	09:00	2.0	Evaluation	Anforderungskatalog	Anforderungskatalog erarbeiten			
11.03.2024	07:00	11:30	4.5	Evaluation	Analyse PostgreSQL HA Cluster Lösungen	Informationen Sammeln	pgpool II	pgpool II hat kein GitHub Repository.	pgpool II fällt somit direkt aus der betrachtung raus, da kein vergleich möglich ist.
11.03.2024	12:00	13:30	1.5	Dokumentation	Dokumentation	Dokumentation erweitern			
11.03.2024	16:45	17:30	0.5	Dokumentation	Dokumentation	Dokumentation erweitern	Stakeholder erfassen		
13.03.2024	17:45	19:45	2.0	Evaluation	Analyse PostgreSQL HA Cluster Lösungen	Stackgres und Citus analysieren	Citus row-based-sharding	Citus Dokumentation stark Textlastig. Wenig Abbildungen, vieles muss selber gezeichnet werden.	
14.03.2024	19:45	20:45	1.0	Evaluation	Analyse PostgreSQL HA Cluster Lösungen	Projektcontrolling Arbeiten	Citus row-based-sharding		
14.03.2024	20:45	21:30	0.8	Dokumentation	Dokumentation	Dokumentation erweitern	Citus row-based-sharding Dokumentieren		
16.03.2024	17:45	18:30	0.8	Dokumentation	Dokumentation	Dokumentation erweitern	Zweiter Statusbericht verfassen		
17.03.2024	14:45	16:30	1.8	Dokumentation	Dokumentation	Dokumentation erweitern	ACID Exkurs erfassen		
17.03.2024	19:30	20:00	0.5	Dokumentation	Dokumentation	Dokumentation erweitern	Listings sauber machen.		
17.03.2024	20:15	21:00	0.8	Dokumentation	Dokumentation	Dokumentation erweitern	Neue Listing-Sprache für yaml-Files erstellt, da noch einige folgen werden.		
18.03.2024	14:00	16:00	2.0	Dokumentation	Dokumentation	Dokumentation erweitern	Statusbericht 2 fertig Schreiben und Mail an Norman Süssstrunk senden		
18.03.2024	20:20	21:50	1.5	Evaluation	Vorbereitung Benchmarking	pgbench analysieren	Percona ist Dein Freund		
19.03.2024	08:00	10:00	2.0	Evaluation	Analyse PostgreSQL HA Cluster Lösungen	yugabytedb		Veeam Kast K10 wird nicht vor Angabe Diplomarbeit fertig sein	Backups lokal speichern. Veeam Integration wird im Nachgang implementiert.
19.03.2024	10:00	10:30	0.5	Evaluation	Analyse PostgreSQL HA Cluster Lösungen	Backup Abhandlungen			
19.03.2024	14:00	16:00	2.0	Dokumentation	Dokumentation	Dokumentation erweitern	yugabytedb		
20.03.2024	16:00	16:15	0.2	Dokumentation	Dokumentation	Termin für 2. Fachgespräch organisieren			
21.03.2024	18:00	20:00	2.0	Dokumentation	Dokumentation	Dokumentation erweitern	Projektcontrolling gemacht.		
22.03.2024	09:00	11:00	2.0	Evaluation	Vorbereitung Benchmarking	zabbix analysieren			
22.03.2024	13:00	14:30	1.5	Evaluation	Vorbereitung Benchmarking	Benchmark Settings setzen			
22.03.2024	14:30	15:30	1.0	Dokumentation	Dokumentation	Dokumentation erweitern	Projektcontrolling und Dokumentation		
22.03.2024	16:45	19:00	2.8	Dokumentation	Dokumentation	Dokumentation erweitern	Analyse gängiger PostgreSQL HA Cluster Lösungen - Patroni, Stackgres, CloudNativePG dokumentieren / Benchmarking		
24.03.2024	14:30	17:30	3.0	Dokumentation	Dokumentation	Dokumentation erweitern	Analyse gängiger PostgreSQL HA Cluster Lösungen - Patroni, Stackgres, CloudNativePG dokumentieren		
24.03.2024	19:30	22:30	3.0	Dokumentation	Dokumentation	Dokumentation erweitern	Analyse gängiger PostgreSQL HA Cluster Lösungen - Patroni, Stackgres, CloudNativePG dokumentieren / Arbeitsrapport		
25.03.2024	08:00	11:00	3.0	Evaluation	Analyse PostgreSQL HA Cluster Lösungen	rke2 - local-path-provisioner Installieren	Anforderungen recht hoch. Es wird ein guillermotleftprivate container registry guillermotright verlangt. Das KSGR hat Harbor im Einsatz, allerdings ist dieses nicht für das Evaluationssetting gedacht.		
25.03.2024	13:00	14:45	2.8	Evaluation	Analyse PostgreSQL HA Cluster Lösungen	yugabytedb			Eine mögliche Lösung könnte sein, rke2 als Registry zu verwenden.
26.03.2024	12:00	13:00	1.0	Evaluation	Analyse PostgreSQL HA Cluster Lösungen	yugabytedb Installation	Norman verspätete sich wegen eines Privaten Notfalls.		
26.03.2024	14:45	15:00	0.2	2. Expertengespräch	2. Expertengespräch		Aus versehen YugabyteDB Anywhere (Repository yugaware) installiert.	Termin wird auf morgen verschoben	
26.03.2024	15:00	16:00	1.0	Evaluation	Analyse PostgreSQL HA Cluster Lösungen	yugabytedb Installation	Diese Installation benötigt zwingend eine Subscription.	YugabyteDB (Repository yugabyte) verwenden.	
							Wäre ein No-Go	Dies ist nach wie vor Open-Source	

Tabelle I: Arbeitsrapport

## II Protokoll - Fachgespräche

Fachgespräch	Datum	Fachexperte	Nebenexperte	Studenten	Fragen	Antworten	Sonstige Themen	Bemerkungen
1	14.02.2024	Norman Süssstrunk	-	Michael Graber Curdin Roffler	- Darf eine Vorauswahl stattfinden, um den Aufwand zur reduzieren?  - Muss das Protokoll des Fachgesprächs jeweils Zeitnah freigegeben werden? - Hat Norman ggf. noch Vorschläge zu PostgreSQL Clustern gefunden? - Soll ich die Gewichtung mit 100 Punkten machen oder 1000? Im Moment haben diverse Punkte eine sehr kleine Punktzahl - Soll die Disposition in den Anhang? Diese ist 50 Seiten lang	- Eine Vorauswahl ist sinnvoll und in diesem Rahmen fast zwingend notwendig, da sonst viel zuviel Zeit investiert werden müsste	- Vorstellung Norman Süssstrunk, Curdin Roffler und Michael Graber - Kontaktdaten shared  - Bei Fragen jederzeit an Norman wenden - Norman braucht aber mindestens 1 Woche vorlaufzeit - Norman wird sich spätestens zur Halbzeit melden. - Norman wird sic	- Es wurden zwar für alle Studenten von Norman Süssstrunk Zoom-Räume bereitgestellt, aus effizienzgründen nahmen Curdin Roffler und ich beide am selben Meeting teil
2	26.03.2024	Norman Süssstrunk	-	Michael Graber			- Protokoll genehmigen	

Tabelle II: Fachgespräche - Protokoll

==:

## Diplomarbeit

III Statusbericht

III.I Status Report 1

III.II Status Report 2

## IV Kommentare / Anmerkungen

Hier werden Kommentare und Anmerkungen, welche für das Fazit wichtig sein könnten, gesammelt.

Woche	Beschreibung / Event / Problem
KW10	<p>Vier ganze Tage war ich in Thalwil für die Oracle Multitenant-Schulung für das ExaCC Projekt (Ablösung HP-UX).</p> <p>Am Freitag war ich ebenfalls fast den ganzen Tag dran.</p> <p>Weitere Termine werden folgen, das Risiko durch das Projekt tritt langsam ein.</p> <p>Projekt Zeitlich im Verzug.</p>
KW11	<p>Nebst dem HP-UX Ablösungsprojekt schlagen auch diverse Betriebsthemen ein.</p> <p>Die analyse der PostgreSQL HA Cluster nimmt ebenfalls mehr Zeit in Anspruch, als erwartet.</p> <ul style="list-style-type: none"> <li>- HP-UX Probleme am Montag.</li> </ul> <p>Backups sind über das Weekend nicht durchgelaufen.</p>
KW12	<p>Die ganze Montagsplanung wurde über den Haufen geworfen.</p> <ul style="list-style-type: none"> <li>- Besprechung bezüglich Backup.</li> </ul> <p>Veeam Kasten steht noch nicht zur Verfügung.</p> <ul style="list-style-type: none"> <li>- Mittwochvormittag in Zürich, am Nachmittag Probleme mit dfs-Shares.</li> </ul>
< KW12	<p>So wenig Zeit.</p> <ul style="list-style-type: none"> <li>- Mit Norman Termin für nächste Woche Fachgespräch organisiert.</li> </ul> <p>Freue mich darauf.</p>
KW12	<ul style="list-style-type: none"> <li>- Alle Gängigen PostgreSQL HA Lösungen dokumentiert. Aufwand für Die Dokumentation weit grösser als erwartet.</li> <li>- YugabyteDB entpuppt sich als recht fordernd.</li> </ul>
KW13	<p>Es benötigt eine «private container registry», mir fehlt die Expertise dazu.</p> <ul style="list-style-type: none"> <li>- Der Aufbau der Projektplanung entpuppt sich begrenzt nutzbar.</li> </ul> <p>Das erstellen der Evaluationsinfrastruktur</p> <ul style="list-style-type: none"> <li>- Das Problem mit dem «private container registry» rührte daher,</li> </ul>
KW13	<p>dass das YugabyteDB Anywhere (Repository Yugaware) verwendet wurde.</p> <p>Kurz ein Schock, dass YugabyteDB ausgeschieden ist.</p> <p>Später bemerkte ich, dass man das Repository Yugabyte verwendet muss.</p>
KW13	<ul style="list-style-type: none"> <li>- Bereits jetzt viel gerlernt über Kubernetes, Ranger (rke2) und Helm</li> </ul>

Tabelle III: Kommentare - Anmerkung

## V Evaluation

### V.I Maintenance - CloudNativePG

Das Projekt hat eine relativ hohe Anzahl an aktiven Issues, wobei viele neue dazugekommen sind:

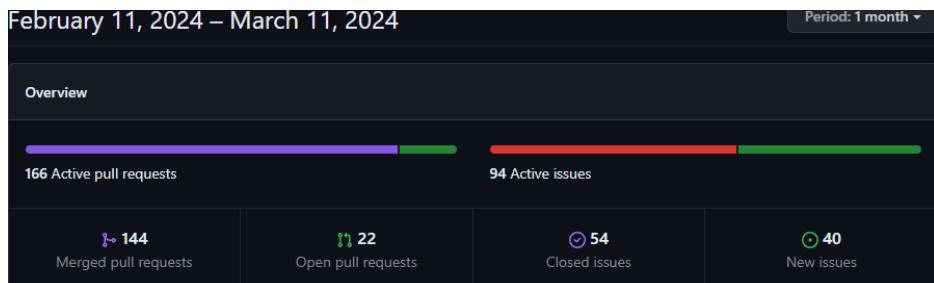


Abbildung I: CloudNativePG - Pulse

Der Code ist aber recht gut gepflegt, Code wird nicht nur regelmässig hinzugefügt, sondern auch entfernt. Auffällig ist, das im April 2022 eine grosse Menge Code entfernt wurde:

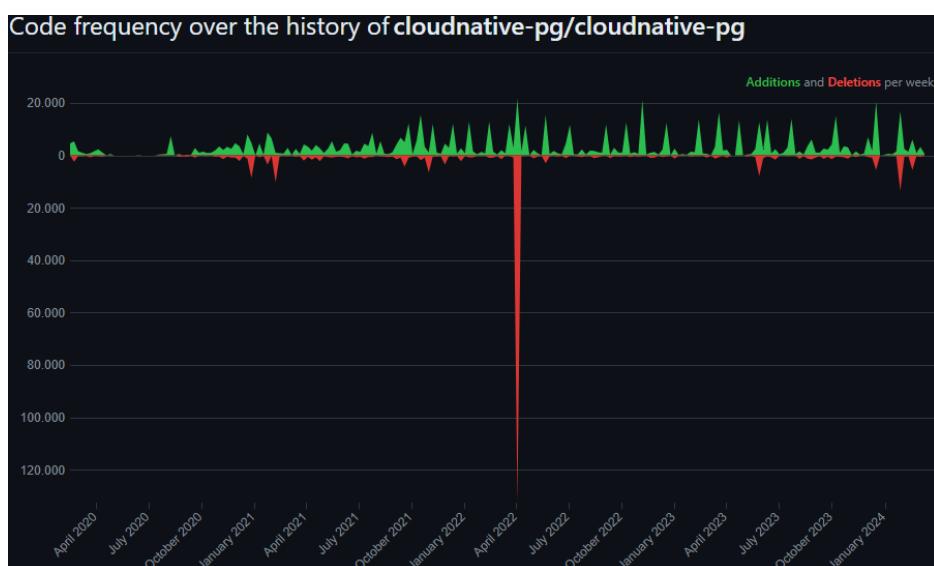


Abbildung II: CloudNativePG - Code Frequency

Das Projekt hält die meisten Standards von GitHub ein:

## Diplomarbeit



Community Standards

Here's how this project compares to [recommended community standards](#).

**Checklist**

- ✓ Description
- ✓ README
- ✓ Code of conduct
- ✓ Contributing
- ✓ License
- ✓ Security policy
- ✓ Issue templates
- Pull request template
- Repository admins accept content reports

Abbildung III: CloudNativePG - Community Standards

Die Contributors committen zwar Regelmässig auf das Projekt, allerdings fügen sie ungleich mehr dazu als sie alten Code bereinigen.

Das führt dann dazu, dass es dann zu grösseren Aufräumarbeiten kommt wie im April 2022.  
Es kann der Eindruck gewonnen werden, dass der Code wenig aufgeräumt wird und viel Balast mit sich schleppt,  
was ein Sicherheitsrisiko darstellen kann:

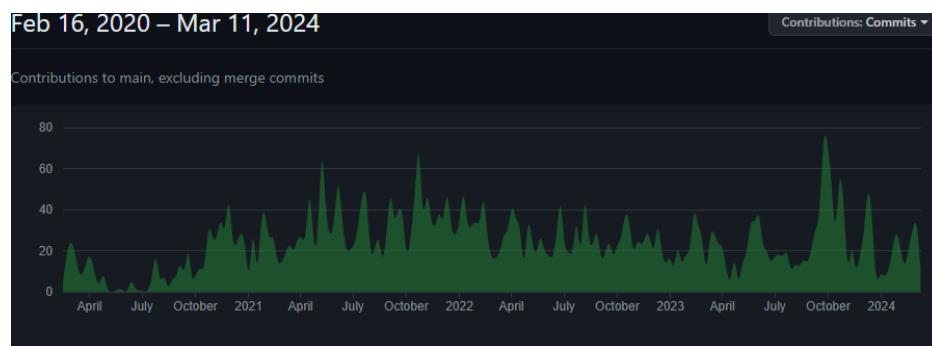


Abbildung IV: CloudNativePG - Contributors Commits

## Diplomarbeit



Abbildung V: CloudNativePG - Contributors Deletations



Abbildung VI: CloudNativePG - Contributors Additions

Commits werden regelmässig abgesetzt, allerdings gibt es immer wieder gehäufte Commits.  
Oft um die Monatswechsel herum:



Abbildung VII: CloudNativePG - Commit Activity

Nebst dem Projekt cloudnative-pg der «© The CloudNativePG Contributors» ist CloudNativePG-Gründer EDB noch immer ein grosser Contributor.

# Diplomarbeit

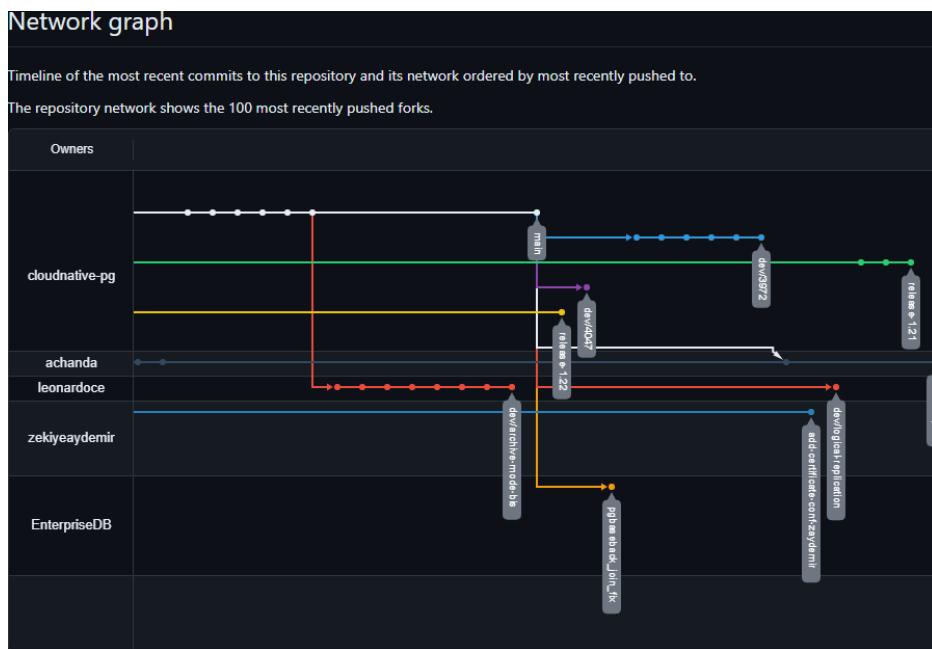


Abbildung VIII: CloudNativePG - Network Graph

V.II Maintenance - Patroni

Patroni wird von Zalando regelmäßig gepflegt. Das Projekt hat eine überschaubare Anzahl an Issues, wird aber Regelmässig

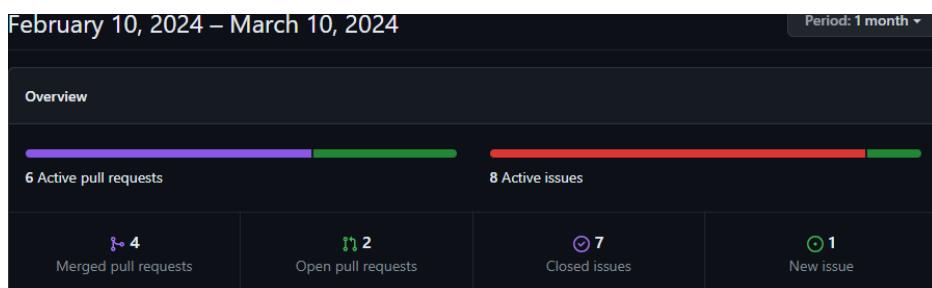


Abbildung IX: Patroni - Pulse

Code wird Regelmässig hinzugefügt und entfernt:

## Diplomarbeit

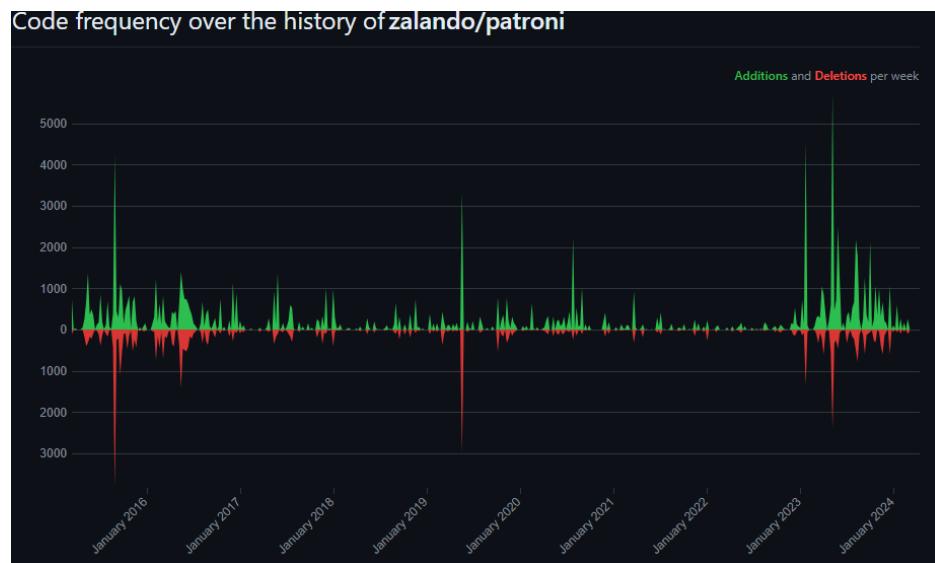


Abbildung X: Patroni - Code Frequency

Das Projekt hält auch die gängigen Standards auf Github ein:

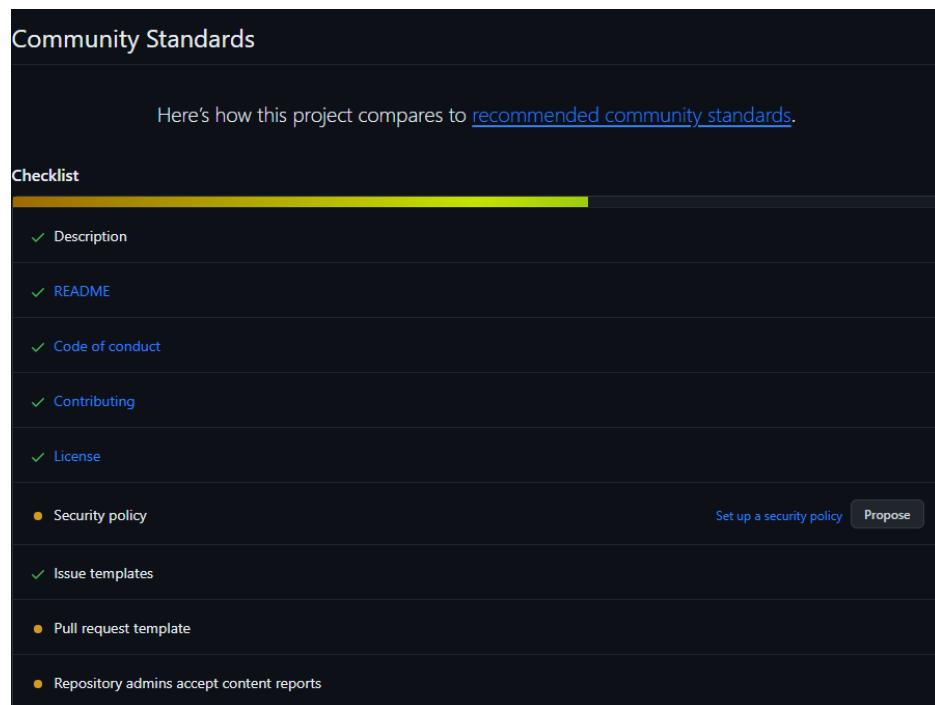


Abbildung XI: Patroni - Community Standards

Die Contributors commiten, löschen und erweitern Patroni Regelmässig:

## Diplomarbeit



Abbildung XII: Patroni - Contributors Commits



Abbildung XIII: Patroni - Contributors Deletations



Abbildung XIV: Patroni - Contributors Additions

Commits werden nach wie vor immer noch Regelmässig eingespielt, auch wenn die Frequenz etwas nachgelassen hat:

## Diplomarbeit

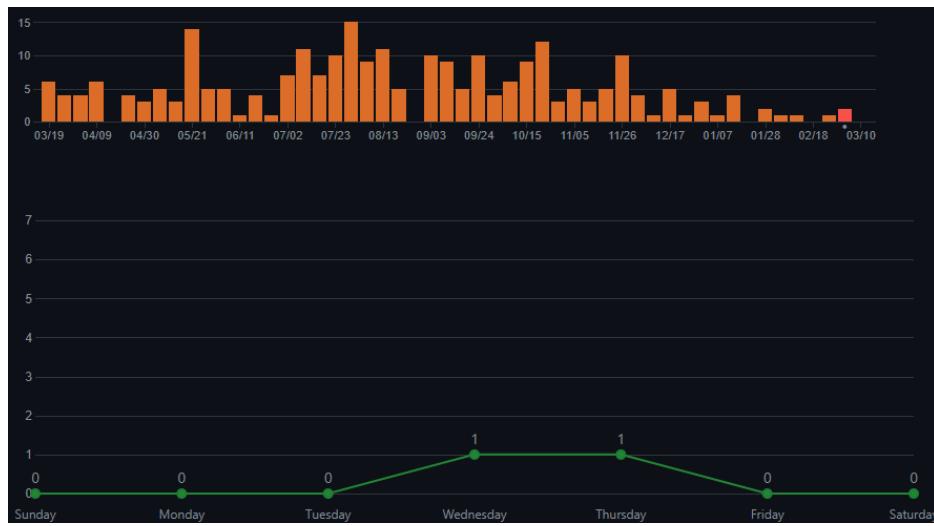


Abbildung XV: Patroni - Commit Activity

Nebst Zalando selbst hat auch EnterpriseDB [[LNF967SI](#)] ein grösseres Repository eingebunden. Dies weil EnterpriseDB stark auf Patroni setzt.

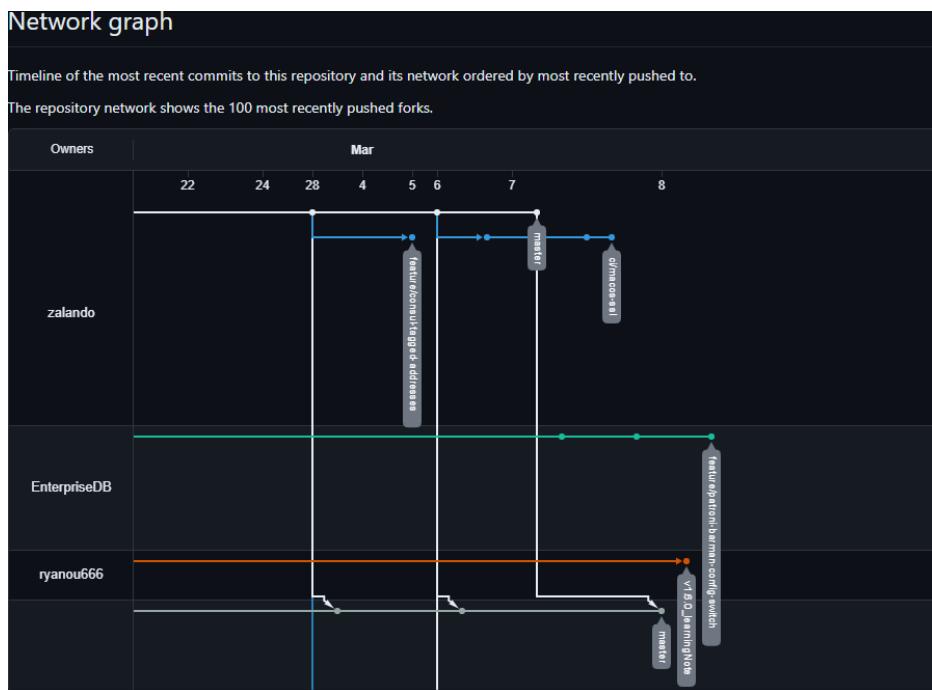


Abbildung XVI: Patroni - Network Graph

## V.III Maintenance - StackGres - Citus

Bei StackGres gab es im letzten Monat keine wirkliche Bewegung:

## Diplomarbeit

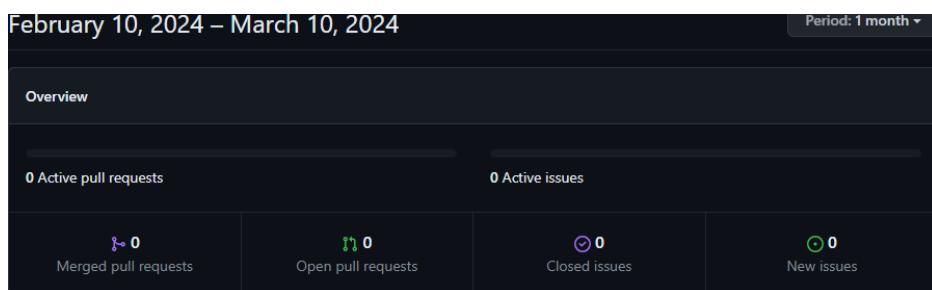


Abbildung XVII: Stackgres - Pulse

Anders sieht es bei Citus aus, die Firma die mittlerweile zu Microsoft gehört, schliesst Issues rasch und hat eine verhältnismässig hohe Requistrate:

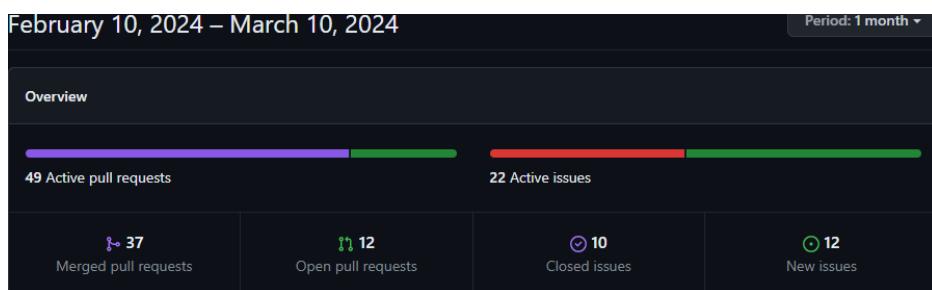


Abbildung XVIII: Citus - Pulse

Bei Stackgres wird sehr viel Code hinzugefügt oder gelöscht, beim älteren Citus wurden weniger änderungen verzeichnetnet:

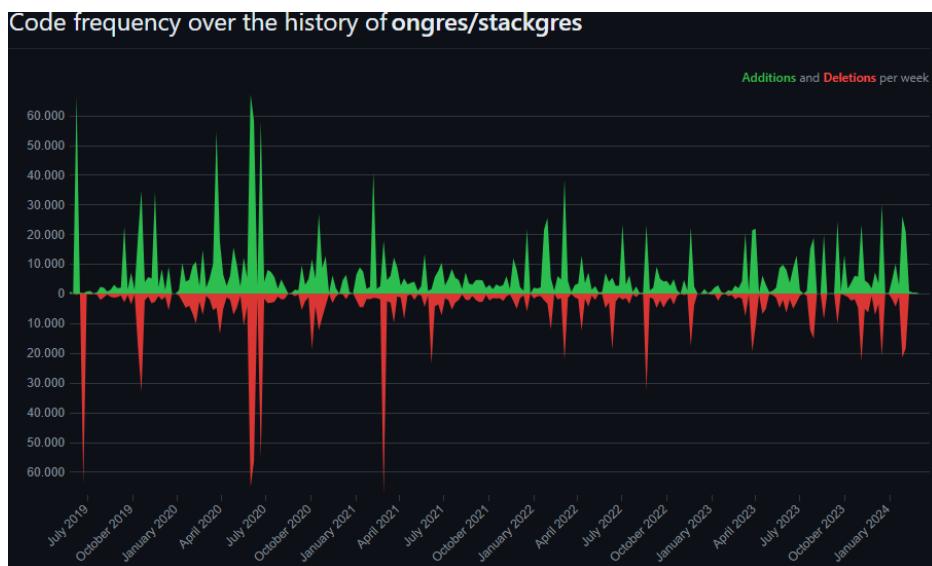


Abbildung XIX: Stackgres - Code Frequency

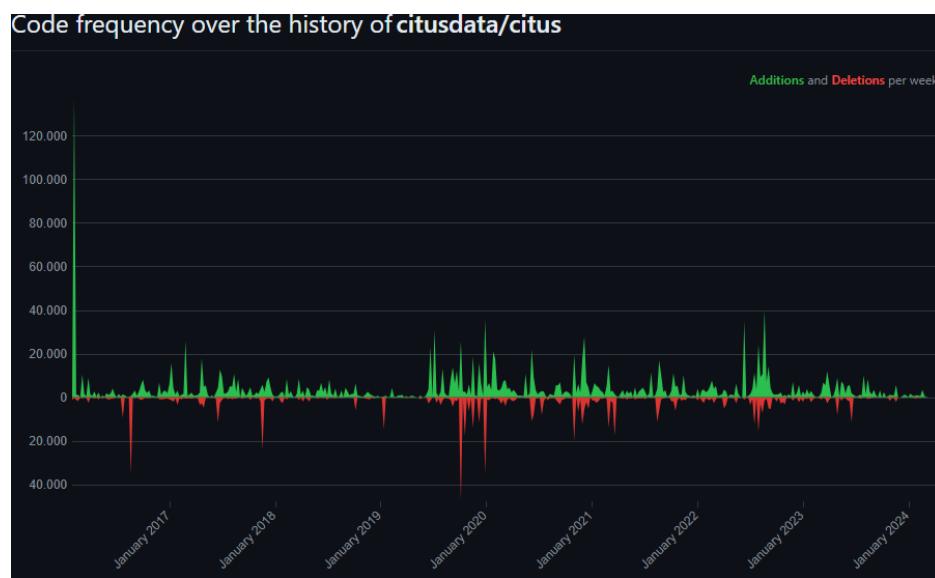


Abbildung XX: Citus - Code Frequency

Citus legt einen hohen Stellenwert auf die Community-Standards, Stackgres selbst schneidet hier nur Mittelmässig ab:

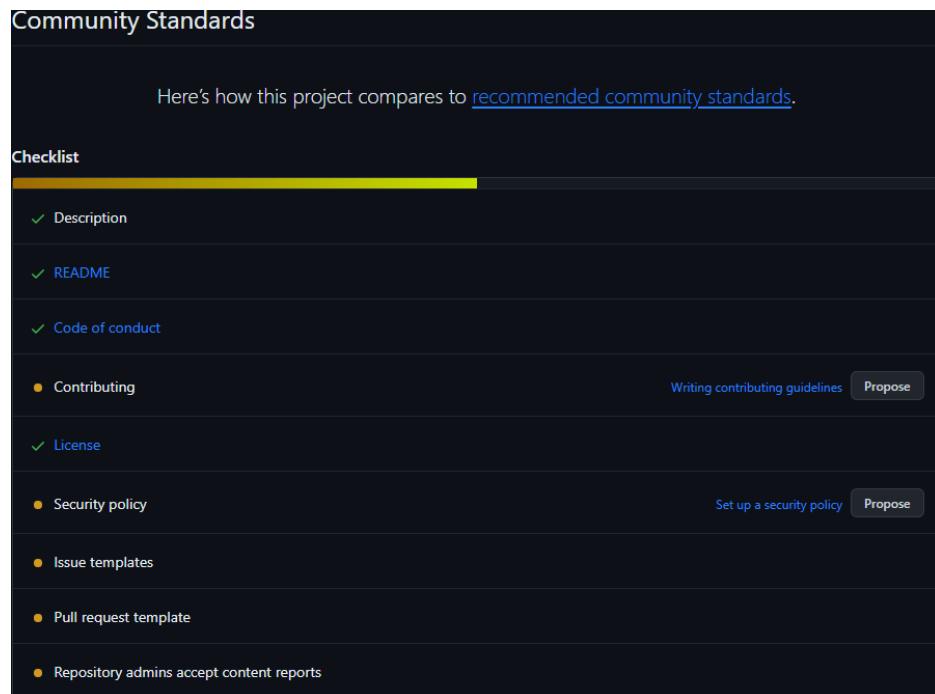


Abbildung XXI: Stackgres - Community Standards

Community Standards

Here's how this project compares to [recommended community standards](#).

**Checklist**

Item	Status
Description	✓
README	✓
Code of conduct	✓
Contributing	✓
License	✓
Security policy	✓
Issue templates	●
Pull request template	✓
Repository admins accept content reports	●

Abbildung XXII: Citus - Community Standards

Die Stackgres Contributors pflegen aktiv Additions ein, löschen Regelmässig und Commiten ebenfalls auf die main-Branch. Citus, dessen Repository länger Committed wird, hat weniger bewegung auf die main-Branch.

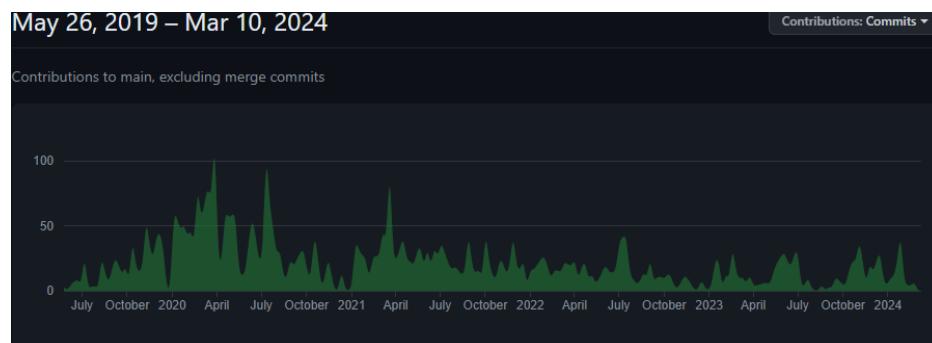


Abbildung XXIII: Stackgres - Contributors Commits

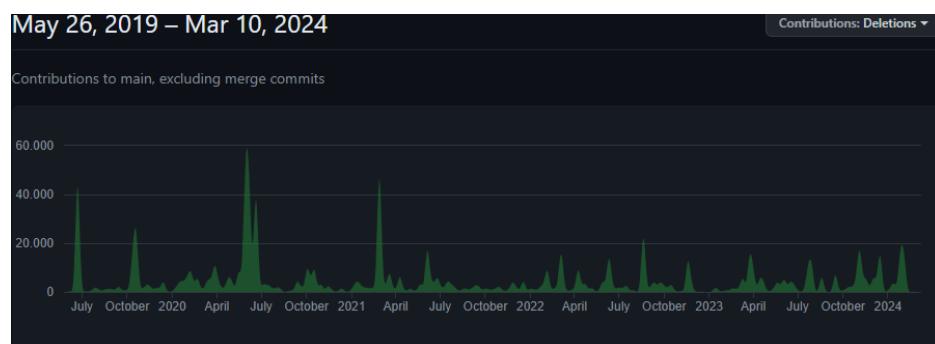


Abbildung XXIV: Stackgres - Contributors Deletions

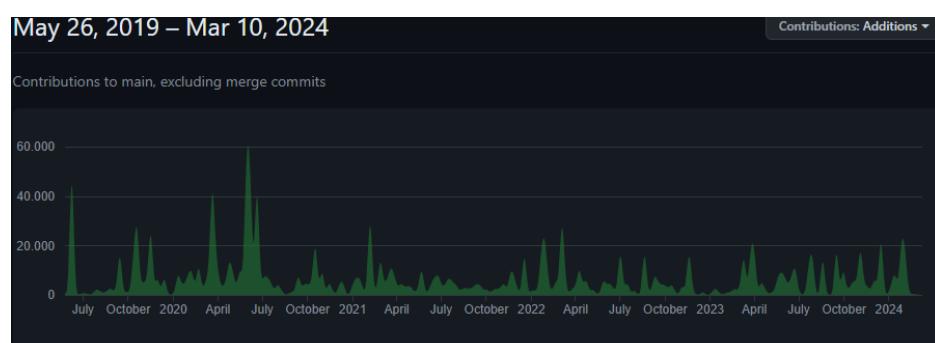


Abbildung XXV: Stackgres - Contributors Additions

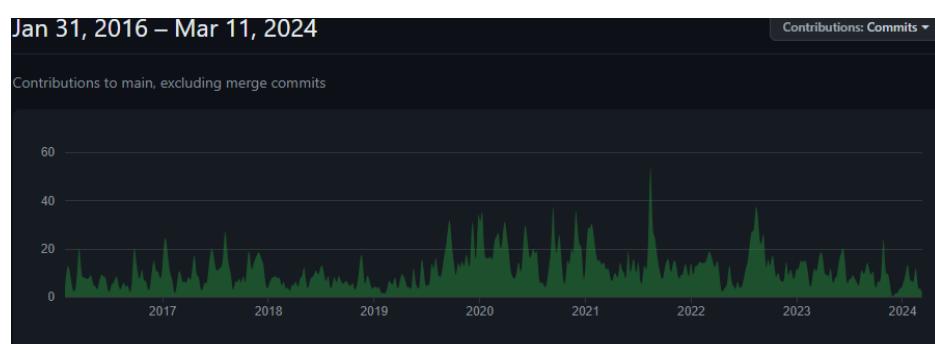


Abbildung XXVI: Citus - Contributors Commits

## Diplomarbeit

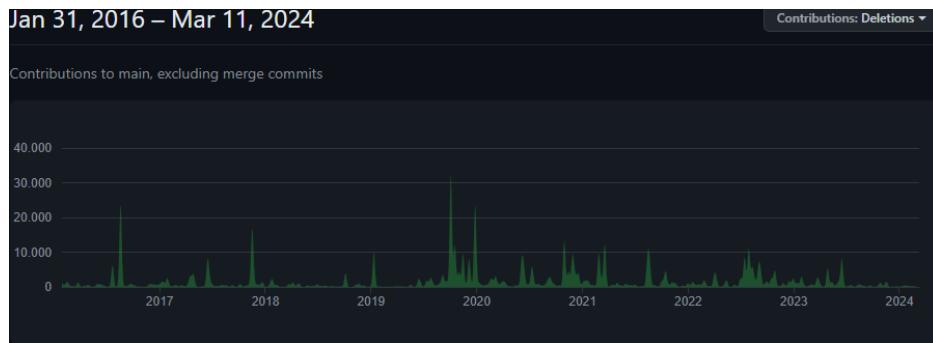


Abbildung XXVII: Citus - Contributors Deletions



Abbildung XXVIII: Citus - Contributors Additions

Gerade Ende Januar gab es bei Stackgres eine grössere Anzahl Commits, anhand der statistik wird ersichtlich, dass i.d.R. einmal pro Monat grössere Mengen an Commits eingespielt werden. Bei Citus gibt es ebenfalls Regelmässig grössere Mengen an Commits, allerdings scheint bei citusdata mehr mit kürzeren Sprints gearbeitet zu werden als bei ongres denn die Commits sind Regelmässiger:

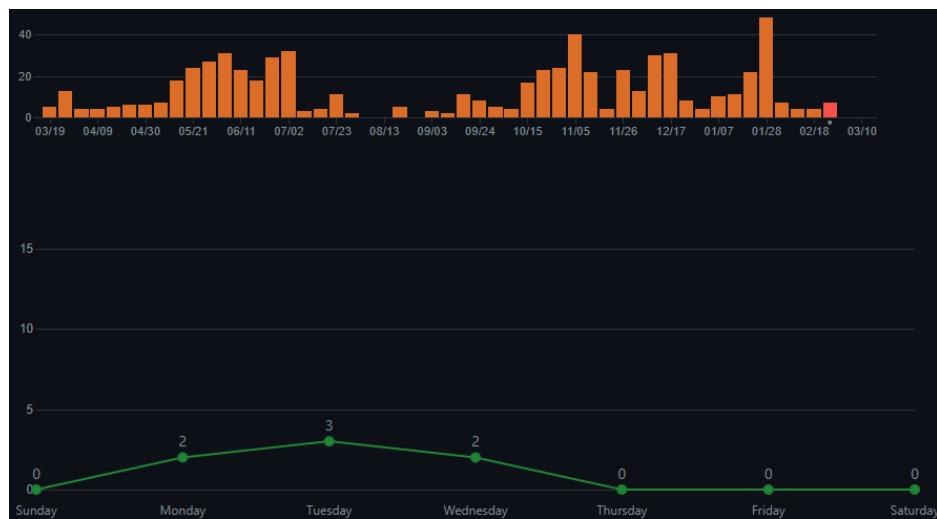


Abbildung XXIX: Stackgres - Commit Activity

## Diplomarbeit

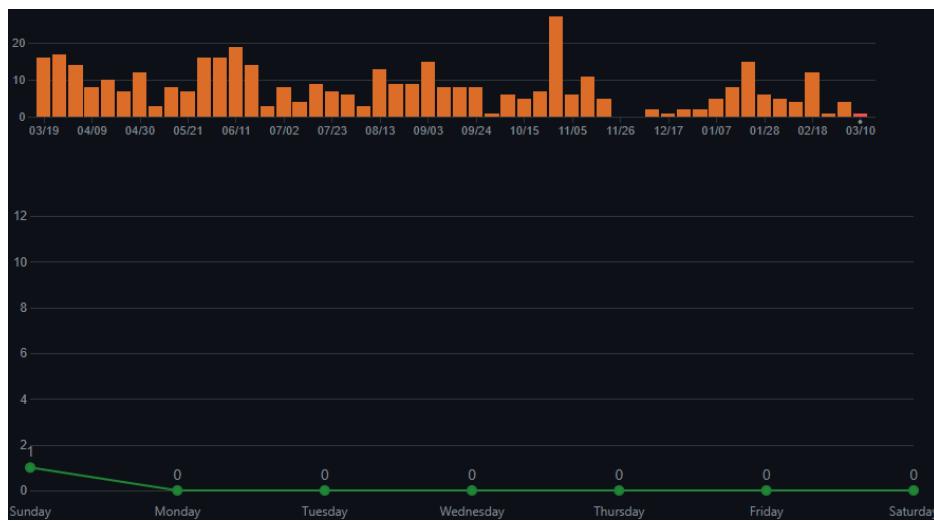


Abbildung XXX: Citus - Commit Activity

In letzter Zeit haben nur ongres, der Entwickler von Stackgres, als auch citusdata, grössere Commits auf das Repository gefahren. Andere grössere Entwickler wie EnterpriseDB sind abwesend.



Abbildung XXXI: Stackgres - Network Graph



Abbildung XXXII: Citus - Network Graph

#### V.IV Maintenance - YugabyteDB

Das Projekt hat eine sehr hohe Anzahl an aktiven Issues, wobei viele neue dazugekommen sinned:

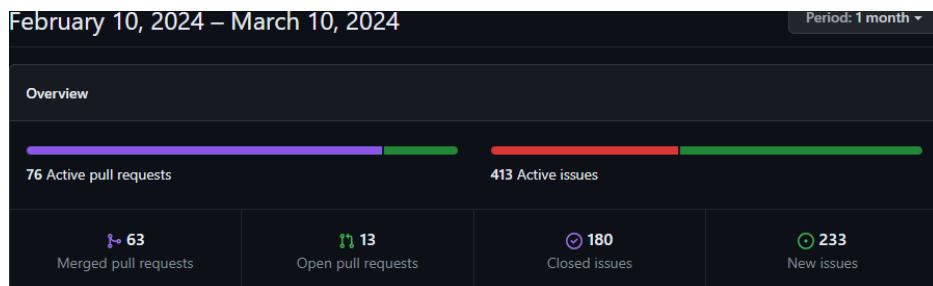


Abbildung XXXIII: YugabyteDB - Pulse

Die Code Frequency kann nicht ausgegeben werden, es gab zu viele Commits:



Abbildung XXXIV: YugabyteDB - Code Frequency

Das Projekt hält nur die wichtigsten Community Standards ein:

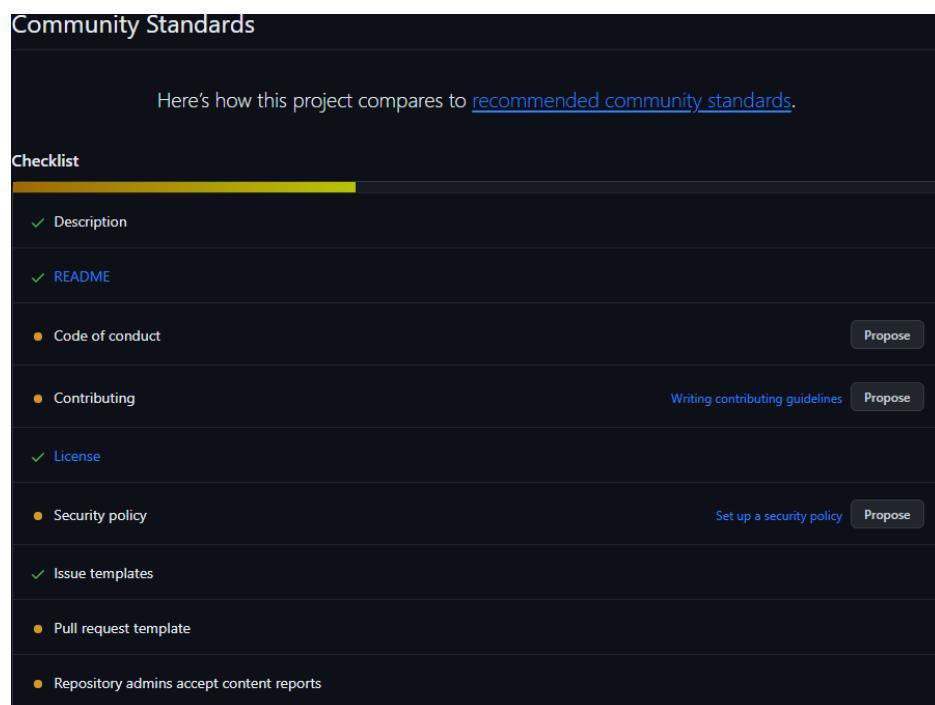


Abbildung XXXV: YugabyteDB - Community Standards

Es werden immer wieder Commits abgesetzt, allerdings sind diese nicht weiter aufgeteilt in Commits, Additions und Deletations:

## Diplomarbeit

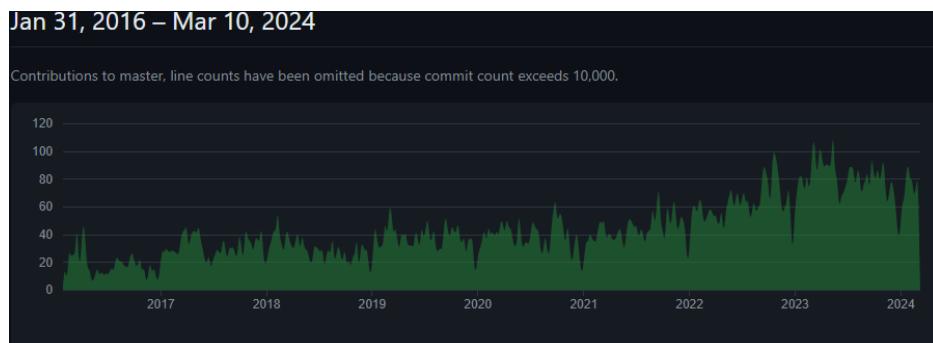


Abbildung XXXVI: YugabyteDB - Contributors

Die Commits wiederum werden Regelmässig ausgeführt, es wird scheinbar in kurzen Sprints gearbeitet:



Abbildung XXXVII: YugabyteDB - Commit Activity

YugabyteDB ist der Maintainer seines Produkts.

Es gibt keine anderen Grossen Contributors:

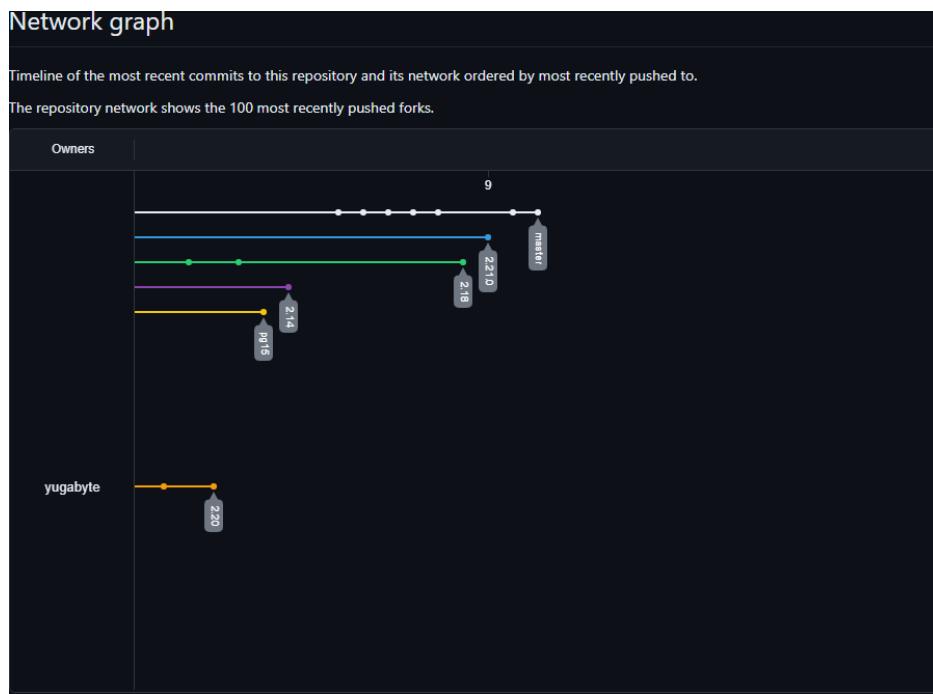


Abbildung XXXVIII: YugabyteDB - Network Graph

## VI Evaluationssysteme - Installation

### VI.I rke2

#### VI.I.I Vorbereitung

Da Package aus WAN-Repositories geladen werden müssen, muss eine Proxy-Connection nach aussen gemacht werden können:

```

1 sudo nano /etc/profile.d/proxy.sh
2
3 export https_proxy=http://sproxy.sivc.first-it.ch:8080
4 export HTTPS_PROXY=http://sproxy.sivc.first-it.ch:8080
5 export http_proxy=http://sproxy.sivc.first-it.ch:8080
6 export HTTP_PROXY=http://sproxy.sivc.first-it.ch:8080
7 export no_proxy=localhost,127.0.0.0/8,::1,10.0.0.0/8,172.16.0.0/12,192.168.0.0/16
8 export NO_PROXY=localhost,127.0.0.0/8,::1,10.0.0.0/8,172.16.0.0/12,192.168.0.0/16
9
10 source /etc/profile.d/proxy.sh

```

Listing 1: Proxy Settings

#### VI.I.II Installation

##### VI.I.II.I server - sks1183

Es gibt kein apt-Package. Daher muss zuerst das tarball-Package heruntergeladen werden.

## Diplomarbeit

Zuerst wird das Verzeichnis für rke2 erstellt:

```
1 mkdir -p /etc/rancher/rke2/
2 mkdir -p /var/lib/rancher/rke2/server/manifests/
```

Listing 2: rke2 server - Verzeichnis erstellen

```
1 # /etc/rancher/rke2/
2 cluster-cidr: "198.18.0.0/16"
3 service-cidr: "198.19.0.0/16"
4 cni:
5   - cilium
6 disable:
7   - rke2-canal
```

Listing 3: rke2 server - config.yaml

Cilium muss separat manifestiert werden:

```
1 # /var/lib/rancher/rke2/server/manifests/rke2-cilium-config.yaml
2 ---
3 apiVersion: helm.cattle.io/v1
4 kind: HelmChartConfig
5 metadata:
6   name: rke2-cilium
7   namespace: kube-system
8 spec:
9   valuesContent: |
10     eni:
11       enabled: true
```

Listing 4: rke2 server - cilium-config.yaml

Das Package kann nun installiert und aktiviert werden:

```
1 curl -sfL https://get.rke2.io | INSTALL_RKE2_VERSION=v1.29.0+rke2r1 sh -
2 systemctl enable rke2-server.service
3 systemctl start rke2-server.service
```

Listing 5: rke2 server installieren

## VI.I.II.II agents - sks1184 / sks1185

Der Agent muss direkt heruntergeladen, installiert und aktiviert werden:

```
1 curl -sfL https://get.rke2.io | INSTALL_RKE2_TYPE="agent" INSTALL_RKE2_VERSION=v1
  .29.0+rke2r1 sh -
2 systemctl enable rke2-agent.service
3 mkdir -p /etc/rancher/rke2/
```

Listing 6: rke2 agenten installieren

## Diplomarbeit

Die Konfiguration muss nun konfiguriert werden. Dem Agents müssen den Server und den Server Token erhalten:

```
1 # /etc/rancher/rke2/config.yaml
2 server: https://10.0.20.97:9345
3 token: K1042bf32f28282edad37cbac4b77ccfa1cd44a26f0ea2c19111ed664013954a326::server
   :7a430a28b29501b778543f0882a156b8
```

Listing 7: rke2 agent - config.yaml

Nun muss der Dienst restartet werden

```
1 systemctl start rke2-agent.service
```

Listing 8: -rke2 agent service restart

### VI.I.II.III Cluster Konfiguration

#### VI.I.II.III.I server

Auch für Kubernetes und die Pots müssen die Proxy-Einstellungen gemacht werden:

```
1 nano /etc/default/rke2-server
2 HTTPS_PROXY=http://sproxy.sivc.first-it.ch:8080
3 HTTP_PROXY=http://sproxy.sivc.first-it.ch:8080
4 NO_PROXY=localhost,127.0.0.0/8,::1,10.0.0.0/8,172.16.0.0/12,192.168.0.0/16
5
6 CONTAINERD_HTTPS_PROXY=http://sproxy.sivc.first-it.ch:8080
7 CONTAINERD_HTTP_PROXY=http://sproxy.sivc.first-it.ch:8080
8 CONTAINERD_NO_PROXY=localhost
   ,127.0.0.0/8,::1,10.0.0.0/8,172.16.0.0/12,192.168.0.0/16
```

Listing 9: rke2 server proxy

Dieses File muss entsprechend in das Homeverzeichnis gespeichert werden:

Listing 10: rke2 server proxy kopieren

Für den Netzwerkteil muss nun Cilium installiert werden:

Listing 11: rke2 server cilium installieren

Cilium muss nun aktiviert werden:

Listing 12: rke2 server cilium aktivieren

Der rke2-Server muss nun mit der entsprechenden Config gestartet werden, anschliessend muss Cilium noch in die Conig und diese mittels Service reboot aktiviert werden:

Listing 13: rke2 server starten

Entsprechend muss die Firewall gesetzt werden:

```

1 nano /etc/iptables/rules.v4
2
3 # Generated by iptables-save v1.8.9 (nf_tables)
4 *filter
5 :INPUT DROP [0:0]
6 :FORWARD ACCEPT [0:0]
7 :OUTPUT ACCEPT [0:0]
8 -A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
9 -A INPUT -p udp -m udp --sport 53 -j ACCEPT
10 -A INPUT -p icmp -j ACCEPT
11 -A INPUT -i lo -j ACCEPT
12 -A INPUT -s 10.0.0.0/8 -p tcp -m tcp --dport 22 -j ACCEPT
13 -A INPUT -s 10.0.9.115/32 -p udp -m udp --dport 161 -m comment --comment "Allow
    SNMP for probe 10.0.9.115" -j ACCEPT
14 -A INPUT -s 10.0.9.76/32 -p udp -m udp --dport 161 -m comment --comment "Allow
    SNMP for probe 10.0.9.76" -j ACCEPT
15 -A INPUT -s 10.0.36.147/32 -p udp -m udp --dport 161 -m comment --comment "Allow
    SNMP for probe 10.0.36.147" -j ACCEPT
16 -A INPUT -s 10.0.9.35/32 -p udp -m udp --dport 161 -m comment --comment "Allow
    SNMP for probe 10.0.9.35" -j ACCEPT
17 -A INPUT -s 10.0.9.37/32 -p udp -m udp --dport 161 -m comment --comment "Allow
    SNMP for probe 10.0.9.37" -j ACCEPT
18 -A INPUT -s 10.0.9.74/32 -p udp -m udp --dport 161 -m comment --comment "Allow
    SNMP for probe 10.0.9.74" -j ACCEPT
19 -A INPUT -s 10.0.9.75/32 -p udp -m udp --dport 161 -m comment --comment "Allow
    SNMP for probe 10.0.9.75" -j ACCEPT
20 -A INPUT -s 10.0.9.36/32 -p udp -m udp --dport 161 -m comment --comment "Allow
    SNMP for probe 10.0.9.36" -j ACCEPT
21 -A INPUT -s 10.0.9.14/32 -p udp -m udp --dport 161 -m comment --comment "Allow
    SNMP for probe 10.0.9.14" -j ACCEPT
22 -A INPUT -s 10.0.0.0/8 -p icmp -m icmp --icmp-type 8 -j ACCEPT
23 -A INPUT -s 10.0.0.0/8 -p tcp -m tcp --dport 6443 -j ACCEPT
24 -A INPUT -s 10.0.0.0/8 -p tcp -m tcp --dport 9345 -j ACCEPT
25 COMMIT
26 # Completed
27
28 systemctl restart iptables

```

Listing 14: iptables entries server

Für den Connect der Agents muss noch ein Token generiert werden:

Listing 15: rke2 server token

## VI.I.II.IV agents

## VI.I.II.V local-path-provisioner

Zuerst mussten auf den drei Servern der Storage bereitgestellt werden:

```

1 root@sks1183:~# mkdir /var/local-path-provisioner
2 root@sks1183:~# chmod -R 777 /var/local-path-provisioner/
3
4 root@sks1184:~# mkdir /var/local-path-provisioner
5 root@sks1184:~# chmod -R 777 /var/local-path-provisioner/
6
7 root@sks1185:~# mkdir /var/local-path-provisioner
8 root@sks1185:~# chmod -R 777 /var/local-path-provisioner/

```

Listing 16: local-path-storage auf Linux Bereitstellen

Anschliessend musste rke2 entsprechend angepasst werden. Damit Automatisch der local-path auf das Verzeichnis /var/local-path-provisioner/ geht, muss dies in einem entsprechenden Manifest geschrieben werden:

```

1 kind: ConfigMap
2 apiVersion: v1
3 metadata:
4   name: local-path-config
5   namespace: local-path-storage
6 data:
7   config.json: |-
8     {
9       "nodePathMap": [
10         {
11           "node": "DEFAULT_PATH_FOR_NON_LISTED_NODES",
12           "paths": ["/var/local-path-provisioner"]
13         }
14       ]
15     }
16   setup: |-
17     #!/bin/sh
18     set -eu
19     mkdir -m 0777 -p "$VOL_DIR"
20   teardown: |-
21     #!/bin/sh
22     set -eu
23     rm -rf "$VOL_DIR"
24   helperPod.yaml: |-
25     apiVersion: v1
26     kind: Pod
27     metadata:
28       name: helper-pod
29     spec:
30       priorityClassName: system-node-critical
31       tolerations:

```

## Diplomarbeit

```

32      - key: node.kubernetes.io/disk-pressure
33          operator: Exists
34          effect: NoSchedule
35
36      containers:
37          - name: helper-pod
            image: busybox

```

Listing 17: local-path-provisioner definieren

Zuerst mussten auf den drei Servern der Storage bereitgestellt werden:

```

1 kubectl apply -f /home/gramic/PycharmProjects/rke2_settings/rke2/local-path-
    provisioner.yaml

```

Listing 18: local-path-storage aktualisieren

## VI.I.II.VI MetallB - Proxy / Load Balancer

MetallB musste installiert werden:

```

1 kubectl apply -f https://raw.githubusercontent.com/metallb/metallb/v0.14.4/config/
    manifests/metallb-native.yaml

```

Listing 19: MetallB installieren

Das Konfigurationsmanifest wurde eingespielt:

```

1 apiVersion: metallb.io/v1beta1
2 kind: IPAddressPool
3 metadata:
4   name: distributed-sql
5   namespace: metallb-system
6 spec:
7   addresses:
8     - 10.0.20.106-10.0.20.106
9     - 10.0.20.150-10.0.20.155
10---
11 apiVersion: metallb.io/v1beta1
12 kind: L2Advertisement
13 metadata:
14   name: l2adv
15   namespace: metallb-system
16 spec:
17   ipAddressPools:
18     - distributed-sql

```

Listing 20: MetallB konfigurieren

Das Manifest musste danach eingespielt werden:

## Diplomarbeit

```
1 kubectl apply -f /home/gramic/PycharmProjects/rke2_settings/rke2/metallb-values.yaml
```

Listing 21: MetallB Konfiguration einspielen

## VI.II        yugabyteDB

### VI.II.I      Prerequisites

#### VI.II.I.I     Prometheus Daten

#### VI.II.I.II    StorageClass setzen

Zuerst muss die StorageClass und das PersistentVolume gesetzt werden:

```
1 apiVersion: storage.k8s.io/v1
2 kind: StorageClass
3 metadata:
4   name: yb-storage
5 provisioner: rancher.io/local-path
6 parameters:
7   nodePath: /var/local-path-provisioner
8 volumeBindingMode: WaitForFirstConsumer
9 reclaimPolicy: Delete
10 --
11 apiVersion: v1
12 kind: PersistentVolume
13 metadata:
14   name: yb-storage-pv
15   labels:
16     type: local
17 spec:
18   accessModes:
19     - ReadWriteOnce
20   capacity:
21     storage: 3Gi
22   storageClassName: "yb-storage"
23   hostPath:
24     path: /var/local-path-provisioner
```

Listing 22: yugabyteDB - StorageClass setzen

Die Storage Class und das PersistentVolume muss aktiviert werden:

```
1 gramic@cks4040:~$ kubectl apply -f /home/gramic/PycharmProjects/rke2_settings/
2   yugabytedb/yugabytedb/storageclass.yaml
3 storageclass.storage.k8s.io/yb-storage unchanged
3 persistentvolume/yb-storage-pv created
```

Listing 23: yugabyteDB - StorageClass / PersistentVolume aktivieren

## VI.II.II Installation

Zuerst muss ein Namespace erstellt werden:

```
1 kubectl create namespace yb-platform
```

Listing 24: YugabyteDB - Namespace

```

1 # Default values for Yugabyte.
2 # This is a YAML-formatted file.
3 # Declare variables to be passed into your templates.
4 Component: "yugabytedb"
5
6 fullnameOverride: ""
7 nameOverride: ""
8
9 Image:
10   repository: "yugabytedb/yugabyte"
11   tag: 2.20.2.1-b3
12   pullPolicy: IfNotPresent
13   pullSecretName: ""
14
15 storage:
16   ephemeral: false # will not allocate PVs when true
17   master:
18     count: 1
19     size: 3Gi
20     storageClass: "yb-storage"
21   tserver:
22     count: 1
23     size: 3Gi
24     storageClass: "yb-storage"
25
26 resource:
27   master:
28     requests:
29       cpu: "1"
30       memory: 2Gi
31     limits:
32       cpu: "1"
33       ## Ensure the 'memory' value is strictly in 'Gi' or 'G' format. Deviating
34       ## from these formats
35       ## may result in setting an incorrect value for the 'memory_limit_hard_bytes'
36       ## flag.
37       ## Avoid using floating numbers for the numeric part of 'memory'. Doing so
38       ## may lead to
39       ## the 'memory_limit_hard_bytes' being set to 0, as the function expects
40       ## integer values.
41       memory: 2Gi
42   tserver:

```

## Diplomarbeit

```
39 requests:
40   cpu: "1"
41   memory: 4Gi
42 limits:
43   cpu: "1"
44   ## Ensure the 'memory' value is strictly in 'Gi' or 'G' format. Deviating
45   ## from these formats
46   ## may result in setting an incorrect value for the 'memory_limit_hard_bytes'
47   ## flag.
48   ## Avoid using floating numbers for the numeric part of 'memory'. Doing so
49   ## may lead to
50   ## the 'memory_limit_hard_bytes' being set to 0, as the function expects
51   ## integer values.
52   memory: 4Gi
53
54 replicas:
55   master: 3
56   tserver: 3
57   ## Used to set replication factor when isMultiAz is set to true
58   totalMasters: 3
59
60 partition:
61   master: 0
62   tserver: 0
63
64 updateStrategy:
65   type: RollingUpdate
66
67 # Used in Multi-AZ setup
68 masterAddresses: ""
69
70 isMultiAz: false
71 AZ: ""
72
73 disableYsql: false
74
75 tls:
76   # Set to true to enable the TLS.
77   enabled: false
78   nodeToNode: true
79   clientToServer: true
80   ## Set to false to disallow any service with unencrypted communication from
81   ## joining this cluster
82   insecure: false
83   ## Set enabled to true to use cert-manager instead of providing your own rootCA
84   certManager:
85     enabled: false
```

```
82      # Will create own ca certificate and issuer when set to true
83      bootstrapSelfsigned: true
84      # Use ClusterIssuer when set to true, otherwise use Issuer
85      useClusterIssuer: false
86      # Name of ClusterIssuer to use when useClusterIssuer is true
87      clusterIssuer: cluster-ca
88      # Name of Issuer to use when useClusterIssuer is false
89      issuer: Yugabyte-ca
90      certificates:
91          # The lifetime before cert-manager will issue a new certificate.
92          # The re-issued certificates will not be automatically reloaded by the
93          # service.
94          # It is necessary to provide some external means of restarting the pods.
95          duration: 2160h # 90d
96          renewBefore: 360h # 15d
97          algorithm: RSA # ECDSA or RSA
98          # Can be 2048, 4096 or 8192 for RSA
99          # Or 256, 384 or 521 for ECDSA
100         keySize: 2048
101
102     ## When certManager.enabled=false, rootCA.cert and rootCA.key are used to
103     # generate TLS certs.
104     ## When certManager.enabled=true and bootstrapSelfsigned=true, rootCA is ignored.
105     ## When certManager.enabled=true and bootstrapSelfsigned=false, only rootCA.cert
106     # is used
107     ## to verify TLS certs generated and signed by the external provider.
108     rootCA:
109         cert: "LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tCk1JSUM2VENDQWRHZ0F3SUJBZ01CQVRBTkJna3Foa2lHOXcwQkFRc02"
110         key: "LS0tLS1CRUdJTiBSU0EgUFJJVkfURSBLRVktLS0tLQpNSUlFcEFJQkFBSONBUUVBdU4xYXVpZzhvalUwczQ5cXdBeG"
111
112     ## When tls.certManager.enabled=false
113     ## nodeCert and clientCert will be used only when rootCA.key is empty.
114     ## Will be ignored and genSignedCert will be used to generate
115     ## node and client certs if rootCA.key is provided.
116     ## cert and key are base64 encoded content of certificate and key.
117     nodeCert:
118         cert: ""
119         key: ""
120     clientCert:
121         cert: ""
122         key: ""
```

```
123 tserver: {}
124 #   use_cassandra_authentication: false
125
126 PodManagementPolicy: Parallel
127
128 enableLoadBalancer: true
129
130 ybc:
131   enabled: false
132   ## https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/#resource-requests-and-limits-of-pod-and-container
133   ## Use the above link to learn more about Kubernetes resources configuration.
134   # resources:
135   #   requests:
136   #     cpu: "1"
137   #     memory: 1Gi
138   #   limits:
139   #     cpu: "1"
140   #     memory: 1Gi
141
142 ybCleanup: {}
143   ## https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/#resource-requests-and-limits-of-pod-and-container
144   ## Use the above link to learn more about Kubernetes resources configuration.
145   # resources:
146   #   requests:
147   #     cpu: "1"
148   #     memory: 1Gi
149   #   limits:
150   #     cpu: "1"
151   #     memory: 1Gi
152
153 domainName: "cluster.local"
154
155 serviceEndpoints:
156 - name: "yb-master-ui"
157   type: LoadBalancer
158   annotations: {}
159   clusterIP: ""
160   ## Sets the Service's externalTrafficPolicy
161   externalTrafficPolicy: ""
162   app: "yb-master"
163   loadBalancerIP: ""
164   ports:
165     http-ui: "7000"
166
167 - name: "yb-tserver-service"
168   type: LoadBalancer
```

```

169   annotations:
170     metallb.universe.tf/loadBalancerIPs: 10.0.20.106
171   clusterIP: ""
172   ## Sets the Service's externalTrafficPolicy
173   externalTrafficPolicy: ""
174   app: "yb-tserver"
175   loadBalancerIP: ""
176   ports:
177     tcp-yql-port: "9042"
178     tcp-yedis-port: "6379"
179     tcp-ysql-port: "5433"
180
181 Services:
182 - name: "yb-masters"
183   label: "yb-master"
184   skipHealthChecks: false
185   memory_limit_to_ram_ratio: 0.85
186   ports:
187     http-ui: "7000"
188     tcp-rpc-port: "7100"
189
190 - name: "yb-tservers"
191   label: "yb-tserver"
192   skipHealthChecks: false
193   ports:
194     http-ui: "9000"
195     tcp-rpc-port: "9100"
196     tcp-yql-port: "9042"
197     tcp-yedis-port: "6379"
198     tcp-ysql-port: "5433"
199     http-ycql-met: "12000"
200     http-yedis-met: "11000"
201     http-ysql-met: "13000"
202     grpc-ybc-port: "18018"
203
204
205 ## Should be set to true only if Istio is being used. This also adds
206 ## the Istio sidecar injection labels to the pods.
207 ## TODO: remove this once
208 ## https://github.com/yugabyte/yugabyte-db/issues/5641 is fixed.
209 ##
210 istioCompatibility:
211   enabled: false
212
213 ## Settings required when using multicluster environment.
214 multicluster:
215   ## Creates a ClusterIP service for each yb-master and yb-tserver
216   ## pod.

```

```
217  createServicePerPod: false
218  ## creates a ClusterIP service whos name does not have release name
219  ## in it. A common service across different clusters for automatic
220  ## failover. Useful when using new naming style.
221  createCommonTserverService: false
222
223  ## Enable it to deploy YugabyteDB in a multi-cluster services enabled
224  ## Kubernetes cluster (KEP-1645). This will create ServiceExport.
225  ## GKE Ref - https://cloud.google.com/kubernetes-engine/docs/how-to/multi-
226  ## cluster-services#registering_a_service_for_export
227  ## You can use this gist for the reference to deploy the YugabyteDB in a multi-
228  ## cluster scenario.
229  ## Gist - https://gist.github.com/baba230896/78cc9bb6f4ba0b3d0e611cd49ed201bf
230  createServiceExports: false
231
232  ## Mandatory variable when createServiceExports is set to true.
233  ## Use: In case of GKE, you need to pass GKE Hub Membership Name.
234  ## GKE Ref - https://cloud.google.com/kubernetes-engine/docs/how-to/multi-
235  ## cluster-services#enabling
236  kubernetesClusterId: ""
237
238  mcsApiVersion: "multicluster.x-k8s.io/v1alpha1"
239
240  serviceMonitor:
241    ## If true, two ServiceMonitor CRs are created. One for yb-master
242    ## and one for yb-tserver
243    ## https://github.com/coreos/prometheus-operator/blob/master/Documentation/api.
244    ## md#servicemonitor
245    ##
246    enabled: false
247    ## interval is the default scrape_interval for all the endpoints
248    interval: 30s
249    ## extraLabels can be used to add labels to the ServiceMonitors
250    ## being created
251    extraLabels: {}
252    ## release: prom
253
254  ## Configurations of ServiceMonitor for yb-master
255  master:
256    enabled: true
257    port: "http-ui"
258    interval: ""
259    path: "/prometheus-metrics"
260
261  ## Configurations of ServiceMonitor for yb-tserver
262  tserver:
```

## Diplomarbeit

```

261     enabled: true
262     port: "http-ui"
263     interval: ""
264     path: "/prometheus-metrics"
265   ycql:
266     enabled: true
267     port: "http-ycql-met"
268     interval: ""
269     path: "/prometheus-metrics"
270   ysql:
271     enabled: true
272     port: "http-ysql-met"
273     interval: ""
274     path: "/prometheus-metrics"
275   yedis:
276     enabled: true
277     port: "http-yedis-met"
278     interval: ""
279     path: "/prometheus-metrics"
280
281 commonMetricRelabelings:
282 # https://git.io/JJW5p
283 # Save the name of the metric so we can group_by since we cannot by __name__
284 # directly...
285 - sourceLabels: ["__name__"]
286   regex: "(.*"
287   targetLabel: "saved_name"
288   replacement: "$1"
289 # The following basically retrofit the handler_latency_* metrics to label format
290 #
291 - sourceLabels: ["__name__"]
292   regex: "handler_latency_(yb_[^_]*)([^_]*)([^_]*)(.*"
293   targetLabel: "server_type"
294   replacement: "$1"
295 - sourceLabels: ["__name__"]
296   regex: "handler_latency_(yb_[^_]*)([^_]*)([^_]*)(.*"
297   targetLabel: "service_type"
298   replacement: "$2"
299 - sourceLabels: ["__name__"]
300   regex: "handler_latency_(yb_[^_]*)([^_]*)([^_]*)(_sum|_count)?"
301   targetLabel: "service_method"
302   replacement: "$3"
303 - sourceLabels: ["__name__"]
304   regex: "handler_latency_(yb_[^_]*)([^_]*)([^_]*)(_sum|_count)?"
305   targetLabel: "__name__"
306   replacement: "rpc_latency$4"
307
308 resources: {}

```

```

307
308 nodeSelector: {}
309
310 affinity: {}
311
312 statefulSetAnnotations: {}
313
314 networkAnnotation: {}
315
316 commonLabels: {}
317
318 master:
319     ## Ref: https://kubernetes.io/docs/reference/generated/kubernetes-api/v1.22/# affinity-v1-core
320     ## This might override the default affinity from service.yaml
321     # To successfully merge, we need to follow rules for merging nodeSelectorTerms
322     # that kubernentes
323     # has. Each new node selector term is ORed together, and each match expression
324     # or match field in
325     # a single selector is ANDed together.
326     # This means, if a pod needs to be scheduled on a label 'custom_label_1' with a
327     # value
328     # 'custom_value_1', we need to add this 'subterm' to each of our pre-defined
329     # node affinity
330     # terms.
331     #
332     # Pod anti affinity is a simpler merge. Each term is applied separately, and the
333     # weight is tracked.
334     # The pod that achieves the highest weight is selected.
335     ## Example.
336     # affinity:
337     #   podAntiAffinity:
338     #     requiredDuringSchedulingIgnoredDuringExecution:
339     #       - labelSelector:
340     #         matchExpressions:
341     #           - key: app
342     #             operator: In
343     #             values:
344     #               - "yb-master"
345     #     topologyKey: kubernetes.io/hostname
346     #
347     # For further examples, see examples/yugabyte/affinity_overrides.yaml
348 affinity: {}

349     ## Extra environment variables passed to the Master pods.
350     ## Ref: https://kubernetes.io/docs/reference/generated/kubernetes-api/v1.22/# envvar-v1-core
351     ## Example:

```

```
348 # extraEnv:  
349 # - name: NODE_IP  
350 #   valueFrom:  
351 #     fieldRef:  
352 #       fieldPath: status.hostIP  
353 extraEnv: []  
354  
355 # secretEnv variables are used to expose secrets data as env variables in the  
356 # master pod.  
357 # TODO Add namespace also to support copying secrets from other namespace.  
358 # secretEnv:  
359 # - name: MYSQL_LDAP_PASSWORD  
360 #   valueFrom:  
361 #     secretKeyRef:  
362 #       name: secretName  
363 #       key: password  
364 secretEnv: []  
365  
366 ## Annotations to be added to the Master pods.  
367 podAnnotations: {}  
368  
369 ## Labels to be added to the Master pods.  
370 podLabels: {}  
371  
372 ## Tolerations to be added to the Master pods.  
373 ## Ref: https://kubernetes.io/docs/reference/generated/kubernetes-api/v1.22/#toleration-v1-core  
374 ## Example:  
375 # tolerations:  
376 # - key: dedicated  
377 #   operator: Equal  
378 #   value: experimental  
379 #   effect: NoSchedule  
380 tolerations: []  
381  
382 ## Extra volumes  
383 ## extraVolumeMounts are mandatory for each extraVolumes.  
384 ## Ref: https://kubernetes.io/docs/reference/generated/kubernetes-api/v1.22/#volume-v1-core  
385 ## Example:  
386 # extraVolumes:  
387 # - name: custom-nfs-vol  
388 #   persistentVolumeClaim:  
389 #     claimName: some-nfs-claim  
390 extraVolumes: []  
391  
392 ## Extra volume mounts  
393 ## Ref: https://kubernetes.io/docs/reference/generated/kubernetes-api/v1.22/#
```

```

393     volumemount-v1-core
394     ## Example:
395     # extraVolumeMounts:
396     # - name: custom-nfs-vol
397     #   mountPath: /home/yugabyte/nfs-backup
398     extraVolumeMounts: []
399
400     ## Set service account for master DB pods. The service account
401     ## should exist in the namespace where the master DB pods are brought up.
402     serviceAccount: ""
403
404     ## Memory limit hard % (between 1-100) of the memory limit.
405     memoryLimitHardPercentage: 85
406
407 tserver:
408     ## Ref: https://kubernetes.io/docs/reference/generated/kubernetes-api/v1.22/#affinity-v1-core
409     ## This might override the default affinity from service.yaml
410     # To successfully merge, we need to follow rules for merging nodeSelectorTerms
411     # that kubernentes
412     # has. Each new node selector term is ORed together, and each match expression
413     # or match field in
414     # a single selector is ANDed together.
415     # This means, if a pod needs to be scheduled on a label 'custom_label_1' with a
416     # value
417     # 'custom_value_1', we need to add this 'subterm' to each of our pre-defined
418     # node affinity
419     # terms.
420     #
421     # Pod anti affinity is a simpler merge. Each term is applied separately, and the
422     # weight is tracked.
423     # The pod that achieves the highest weight is selected.
424     ## Example.
425     # affinity:
426     #   podAntiAffinity:
427     #     requiredDuringSchedulingIgnoredDuringExecution:
428     #       - labelSelector:
429     #         matchExpressions:
430     #           - key: app
431     #             operator: In
432     #             values:
433     #               - "yb-tserver"
434     #     topologyKey: kubernetes.io/hostname
435     # For further examples, see examples/yugabyte/affinity_overrides.yaml
436     affinity: {}
437
438     ## Extra environment variables passed to the TServer pods.

```

## Diplomarbeit

```

434     ## Ref: https://kubernetes.io/docs/reference/generated/kubernetes-api/v1.22/#  
435     envvar-v1-core  
436  
437     ## Example:  
438     # extraEnv:  
439     # - name: NODE_IP  
440     #   valueFrom:  
441     #     fieldRef:  
442     #       fieldPath: status.hostIP  
443     extraEnv: []  
444  
445     ## secretEnv variables are used to expose secrets data as env variables in the  
446     ## tserver pods.  
447     ## If namespace field is not specified we assume that user already  
448     ## created the secret in the same namespace as DB pods.  
449     ## Example  
450     # secretEnv:  
451     # - name: MYSQL_LDAP_PASSWORD  
452     #   valueFrom:  
453     #     secretKeyRef:  
454     #       name: secretName  
455     #       namespace: my-other-namespace-with-ldap-secret  
456     #       key: password  
457     secretEnv: []  
458  
459     ## Annotations to be added to the TServer pods.  
460     podAnnotations: {}  
461  
462     ## Labels to be added to the TServer pods.  
463     podLabels: {}  
464  
465     ## Tolerations to be added to the TServer pods.  
466     ## Ref: https://kubernetes.io/docs/reference/generated/kubernetes-api/v1.22/#  
467     # toleration-v1-core  
468     ## Example:  
469     # tolerations:  
470     # - key: dedicated  
471     #   operator: Equal  
472     #   value: experimental  
473     #   effect: NoSchedule  
474     tolerations: []  
475  
476     ## Sets the --server_broadcast_addresses flag on the TServer, no  
477     ## preflight checks are done for this address. You might need to add  
478     ## 'use_private_ip: cloud' to the gflags.master and gflags.tserver.  
479     serverBroadcastAddress: ""  
480  
481     ## Extra volumes  
482     ## extraVolumeMounts are mandatory for each extraVolumes.  
483

```

## Diplomarbeit

```
479 ## Ref: https://kubernetes.io/docs/reference/generated/kubernetes-api/v1.22/#  
480     volume-v1-core  
481 ## Example:  
482 # extraVolumes:  
483 # - name: custom-nfs-vol  
484 #   persistentVolumeClaim:  
485 #     claimName: some-nfs-claim  
486 extraVolumes: []  
487  
488 ## Extra volume mounts  
489 ## Ref: https://kubernetes.io/docs/reference/generated/kubernetes-api/v1.22/#  
490     volumemount-v1-core  
491 ## Example:  
492 # extraVolumeMounts:  
493 # - name: custom-nfs-vol  
494 #   path: /home/yugabyte/nfs-backup  
495 extraVolumeMounts: []  
496  
497 ## Set service account for tserver DB pods. The service account  
498 ## should exist in the namespace where the tserver DB pods are brought up.  
499 serviceAccount: ""  
500  
501  
502  
503 helm2Legacy: false  
504  
505 ip_version_support: "v4_only" # v4_only, v6_only are the only supported values at  
      the moment  
506  
507 # For more https://docs.yugabyte.com/latest/reference/configuration/yugabyted/#  
      environment-variables  
508 authCredentials:  
509   ysql:  
510     user: "yadmin"  
511     password: "TES2&Daggerfall"  
512     database: ""  
513   ycql:  
514     user: ""  
515     password: ""  
516     keyspace: ""  
517  
518 oldNamingStyle: true  
519  
520 preflight:  
521   # Set to true to skip disk IO check, DNS address resolution, and  
   # port bind checks
```

```

523 skipAll: false
524 # Set to true to skip port bind checks
525 skipBind: false
526
527 ## Set to true to skip ulimit verification
528 ## SkipAll has higher priority
529 skipUlimit: false
530
531 ## Pod securityContext
532 ## Ref: https://kubernetes.io/docs/reference/kubernetes-api/workload-resources/pod-
      -v1/#security-context
533 ## The following configuration runs YB-Master and YB-TServer as a non-root user
534 podSecurityContext:
535   enabled: false
536   ## Mark it false, if you want to stop the non root user validation
537   runAsNonRoot: true
538   fsGroup: 10001
539   runAsUser: 10001
540   runAsGroup: 10001
541
542 ## Added to handle old universe which has volume annotations
543 ## K8s universe <= 2.5 to >= 2.6
544 legacyVolumeClaimAnnotations: false

```

Listing 25: YugabyteDB - Helm Chart Manifest

Die Installation erfolgt dann wie folgt:

```

1 helm install yw-test Yugabytedb/yugabyte --version 2.19.3 -n yb-platform -f /home/
      gramic/PycharmProjects/rke2_settings/yugabytedb/yugabytedb/values.yaml

```

Listing 26: YugabyteDB - Installation

**VI.II.III** Rekonfiguration mit 300GiB Storage

**VI.III** Stackgres mit Citus

**VI.III.I** Prerequisites

**VI.III.I.I** StorageClass setzen

Zuerst muss die StorageClass und das PersistentVolume gesetzt werden:

```

1 # https://docs.yugabyte.com/preview/yugabyte-platform/install-yugabyte-platform/
      prepare-environment/kubernetes/#configure-storage-class
2 # https://github.com/rancher/local-path-provisioner
3 apiVersion: storage.k8s.io/v1
4 kind: StorageClass
5 metadata:
6   name: stackgres-storage

```

## Diplomarbeit

```

7 provisioner: rancher.io/local-path
8 parameters:
9   nodePath: /var/local-path-provisioner
10 volumeBindingMode: WaitForFirstConsumer
11 reclaimPolicy: Delete
12---
13 apiVersion: v1
14 kind: PersistentVolume
15 metadata:
16   name: stackgres-storage-pv
17   labels:
18     type: local
19 spec:
20   accessModes:
21     - ReadWriteOnce
22   capacity:
23     storage: 3Gi
24   storageClassName: "stackgres-storage"
25   hostPath:
26     path: /var/local-path-provisioner

```

Listing 27: StackGres-Citus - StorageClass setzen

Die Storage Class und das PercistenVolume muss aktiviert werden:

```

1 gramic@cks4040:~$ kubectl apply -f /home/gramic/PycharmProjects/rke2_settings/
      stackgres_citus/stackgres_citus/storageclass.yaml
2 storageclass.storage.k8s.io/stackgres-storage created
3 persistentvolume/stackgres-storage-pv created

```

Listing 28: YugabyteDB - StorageClass / PersistentVolume aktivieren

## VI.IV sks9016 - YugabyteDB

### VI.IV.I YugabyteDB - Download und Installation YugabyteDB

Ohne YugabyteDB zu installieren, lässt sich ysql\_bench nicht ausführen. Daher muss das ganze Package erst heruntergeladen werden:

```

1 root@sks9016:~# wget https://downloads.yugabyte.com/releases/2.21.0.0/yugabyte
      -2.21.0.0-b545-linux-x86_64.tar.gz
2

```

Listing 29: sks9016 - Download YugabyteDB On-Premise

Im nächsten Schritt wird es im /opt entpackt und das post\_install.sh-Skript ausgeführt:

```

1 root@sks9016:/opt# tar xvfz yugabyte-2.21.0.0-b545-linux-x86_64.tar.gz && cd
      yugabyte-2.21.0.0/
2 ...

```

## Diplomarbeit

```
3 root@sks9016:/opt/yugabyte-2.21.0.0# ./bin/post_install.sh  
4
```

Listing 30: sks9016 - Installation YugabyteDB On-Premise

Um nun zu Testen, ob das ganze Funktioniert, kann eine Verbindung zum Evaluationssystem hergestellt werden:

```
1 root@sks9016:/opt/yugabyte-2.21.0.0# cd /opt/yugabyte-2.21.0.0/postgres/bin/  
2 root@sks9016:/opt/yugabyte-2.21.0.0/postgres/bin# ./ysqlsh "host=10.0.20.106 user=  
    yadmin"  
3 Password for user yadmin:  
4 ysqlsh (11.2-YB-2.21.0.0-b0)  
5 Type "help" for help.  
6  
7 No entry for terminal type "xterm-256color";  
8 using dumb terminal settings.  
9 yugabyte=# exit  
10
```

Listing 31: sks9016 - Check YugabyteDB On-Premise

Damit ist der Benchmarking-Server ready.

## VI.V Patroni

### VI.V.I Prerequisites

Zuerst muss der Proxy gesetzt werden:

```
1 # sks1232 / sks1233 / sks1234  
2 # Proxy setzen  
3 # nano /etc/profile.d/proxy.sh  
4 export https_proxy=http://sproxy.sivc.first-it.ch:8080  
5 export HTTPS_PROXY=http://sproxy.sivc.first-it.ch:8080  
6 export http_proxy=http://sproxy.sivc.first-it.ch:8080  
7 export HTTP_PROXY=http://sproxy.sivc.first-it.ch:8080  
8 export no_proxy=localhost,127.0.0.0/8,::1,10.0.0.0/8,172.16.0.0/12,192.168.0.0/16  
9 export NO_PROXY=localhost,127.0.0.0/8,::1,10.0.0.0/8,172.16.0.0/12,192.168.0.0/16  
10 # source /etc/profile.d/proxy.sh
```

Listing 32: Patroni - Proxy Settings

Damit das PostgreSQL-Repository eingebunden werden kann, muss dem apt-Proxy gesetzt werden.

Da via Foreman installiert wurde, muss dieser ausgenommen werden:

## Diplomarbeit

```
1 # sks1232 / sks1233 / sks1234
2 # apt-Proxy setzen
3 # nano /etc/apt/apt.conf.d/proxy.conf
4 Acquire::http::Proxy "http://sproxy.sivc.first-it.ch:8080";
5 Acquire::https::Proxy "http://sproxy.sivc.first-it.ch:8080";
6 Acquire::http::proxy::foreman.ksgr.ch "DIRECT";
```

Listing 33: Patroni - apt-Proxy Settings

Im nächsten Schritt kann das PostgreSQL-Repository eingebunden werden.

 Achtung, die von PostgreSQL beschriebene Variante wurde in Debian 10 als Deprecated gesetzt, mit Debian 13 wird diese Repository-Integration einen Fehler werden.

```
1 # sks1232 / sks1233 / sks1234
2 # PostgreSQL Repository einbinden
3 sudo sh -c 'echo "deb https://apt.postgresql.org/pub/repos/apt $(lsb_release -cs)-pgdg main" > /etc/apt/sources.list.d/pgdg.list'
4 wget --quiet -O - https://www.postgresql.org/media/keys/ACCC4CF8.asc | sudo apt-key add -
5
6 # Ausloggen und wieder einloggen
7 apt update
```

Listing 34: Patroni - PostgreSQL einbinden

etcd wird als nächstes installiert.

Hierzu muss zuerst das Repository von GitHub heruntergeladen werden:

## VI.V.II Installation

## VI.V.III Konfiguration

## Diplomarbeit

### VII Evaluationssysteme - Testing

#### VII.I Patroni

#### VII.II StackGres - Citus

Der Node ging down, als der Server `sk51184` heruntergefahren wurde:



Abbildung XXXIX: StackGres Testing - Node `sk51184` down

Entsprechend wurden die Pods ebenfalls auf terminating gesetzt:

Pods( <code>sg-platform</code> ) <code>(ls)</code>													
NAME	PF	READY	STATUS	RESTARTS	CPU	MEM	SCPU/L	XMEM/R	XMEM/L	IP	NODE	AGE	
<code>sg-healing-test-coord-0</code>	●	6/6	Terminating	0	0	0	n/a	0	n/a	198.18.1.60	<code>sk51184</code>	3h8m	
<code>sg-healing-test-coord-1</code>	●	6/6	Running	7	25	363	2	n/a	n/a	198.18.0.59	<code>sk51183</code>	3h7m	
<code>sg-healing-test-coord-2</code>	●	6/6	Running	7	305	394	30	n/a	n/a	198.18.2.90	<code>sk51185</code>	3h7m	
<code>sg-healing-test-shard0-0</code>	●	6/6	Running	7	447	359	22	n/a	17	n/a	<code>sk51184</code>	3h7m	
<code>sg-healing-test-shard0-1</code>	●	6/6	Terminating	0	0	0	n/a	0	n/a	198.18.1.213	<code>sk51184</code>	3h7m	
<code>sg-healing-test-shard0-2</code>	●	6/6	Running	0	378	414	18	n/a	20	n/a	198.18.2.3	<code>sk51185</code>	3h7m
<code>sg-healing-test-shard1-0</code>	●	6/6	Terminating	0	0	0	n/a	0	n/a	198.18.1.227	<code>sk51184</code>	3h7m	
<code>sg-healing-test-shard1-1</code>	●	6/6	Running	7	574	348	28	n/a	17	n/a	198.18.0.163	<code>sk51183</code>	3h7m
<code>sg-healing-test-shard1-2</code>	●	6/6	Running	0	38	421	1	n/a	20	n/a	198.18.2.253	<code>sk51185</code>	3h7m
<code>sg-healing-test-shard2-0</code>	●	6/6	Running	0	29	445	1	n/a	21	n/a	198.18.0.151	<code>sk51185</code>	3h7m
<code>sg-healing-test-shard2-1</code>	●	6/6	Running	7	251	349	12	n/a	17	n/a	198.18.0.186	<code>sk51183</code>	3h7m
<code>sg-healing-test-shard2-2</code>	●	6/6	Terminating	0	0	0	n/a	0	n/a	198.18.1.105	<code>sk51184</code>	3h7m	
<code>stackgres-operator-845db7c47-6tcqb</code>	●	1/1	Running	0	44	214	4	4	20	198.18.0.18	<code>sk51183</code>	3m56s	
<code>stackgres-operator-845db7c47-prkdk</code>	●	1/1	Terminating	1	0	0	0	0	0	198.18.1.165	<code>sk51184</code>	3h1m	
<code>stackgres-reapst-685c4688b5-pks79</code>	●	2/2	Running	0	44	401	2	2	19	198.18.2.243	<code>sk51185</code>	3h1m	

Abbildung XL: StackGres Testing - Pods Down

Der Patroni-Leader des Coordinators aber auch die der Shards wurden einem Failover ausgeführt:

## Diplomarbeit

```
gramic@cks4040: $ kubectl exec -it sg-healing-test-coord-0 -n sg-platform -c patroni -- patronictl list
+ Citus cluster: sg-healing-test --+-----+-----+-----+-----+
| Group | Member | Host | Role | State | TL | Lag in MB |
+-----+-----+-----+-----+-----+-----+-----+
| 0 | sg-healing-test-coord-0 | 198.18.1.60:7433 | Leader | running | 2 | 0 |
| 0 | sg-healing-test-coord-1 | 198.18.0.59:7433 | Sync Standby | streaming | 2 | 0 |
| 0 | sg-healing-test-coord-2 | 198.18.2.90:7433 | Sync Standby | streaming | 2 | 0 |
| 1 | sg-healing-test-shard0-0 | 198.18.0.106:7433 | Replica | streaming | 2 | 0 |
| 1 | sg-healing-test-shard0-1 | 198.18.1.213:7433 | Leader | running | 2 | 0 |
| 1 | sg-healing-test-shard0-2 | 198.18.2.3:7433 | Replica | streaming | 2 | 0 |
| 2 | sg-healing-test-shard1-0 | 198.18.1.227:7433 | Replica | streaming | 2 | 0 |
| 2 | sg-healing-test-shard1-1 | 198.18.0.163:7433 | Replica | streaming | 2 | 0 |
| 2 | sg-healing-test-shard1-2 | 198.18.2.253:7433 | Leader | running | 2 | 0 |
| 3 | sg-healing-test-shard2-0 | 198.18.2.31:7433 | Replica | streaming | 2 | 0 |
| 3 | sg-healing-test-shard2-1 | 198.18.0.186:7433 | Replica | streaming | 2 | 0 |
| 3 | sg-healing-test-shard2-2 | 198.18.1.105:7433 | Leader | running | 2 | 0 |
+-----+-----+-----+-----+-----+-----+-----+
gramic@cks4040: $ kubectl exec -it sg-healing-test-coord-0 -n sg-platform -c patroni -- patronictl list
Error from server: error dialing backend: proxy error from 127.0.0.1:9345 while dialing 10.0.20.104:10250, code 502: 502 Bad Gateway
gramic@cks4040: $ kubectl exec -it sg-healing-test-coord-1 -n sg-platform -c patroni -- patronictl list
+ Citus cluster: sg-healing-test --+-----+-----+-----+-----+
| Group | Member | Host | Role | State | TL | Lag in MB |
+-----+-----+-----+-----+-----+-----+-----+
| 0 | sg-healing-test-coord-0 | 198.18.1.60:7433 | Replica | running | 2 | 0 |
| 0 | sg-healing-test-coord-1 | 198.18.0.59:7433 | Sync Standby | streaming | 3 | 0 |
| 0 | sg-healing-test-coord-2 | 198.18.2.90:7433 | Leader | running | 3 | 0 |
| 1 | sg-healing-test-shard0-0 | 198.18.0.106:7433 | Replica | streaming | 3 | 0 |
| 1 | sg-healing-test-shard0-1 | 198.18.1.213:7433 | Replica | running | 2 | 0 |
| 1 | sg-healing-test-shard0-2 | 198.18.2.3:7433 | Leader | running | 3 | 0 |
| 2 | sg-healing-test-shard1-0 | 198.18.1.227:7433 | Replica | streaming | 2 | 0 |
| 2 | sg-healing-test-shard1-1 | 198.18.0.163:7433 | Replica | streaming | 2 | 0 |
| 2 | sg-healing-test-shard1-2 | 198.18.2.253:7433 | Leader | running | 2 | 0 |
| 3 | sg-healing-test-shard2-0 | 198.18.2.31:7433 | Leader | running | 3 | 0 |
| 3 | sg-healing-test-shard2-1 | 198.18.0.186:7433 | Replica | streaming | 3 | 0 |
| 3 | sg-healing-test-shard2-2 | 198.18.1.105:7433 | Replica | running | 2 | 0 |
+-----+-----+-----+-----+-----+-----+-----+
```

Abbildung XLI: StackGres Testing - Patroni Übersicht

Während dieser Zeit, ist die DB immer erreichbar:

## Diplomarbeit

The screenshot shows a PostgreSQL terminal window with several tabs at the top: 'u...self\_healing\_accounts 2', 'self\_healing\_test.hrm.self\_healing\_employees', and 'self...'. The main area displays a series of SQL queries (lines 184 through 190) followed by a table viewer.

SQL queries:

```
184 - After rebuild
185 ✓ select * from customer_serviceOfficers.self_healing_accounts;
186 ✓ select * from hrm.self_healing_employees;
187 ✓ select * from accountands.self_healing_accountand_protocol;
188 ✓ select * from generell.self_healing_intranet;
189 ✓ select * from generell.self_healing_intranet_users;
190
```

Table viewer:

	intranet_user_id	employees_id
1	100	100
2	200	200
3	300	300
4	400	400
5	500	500
6	600	600
7	700	400
8	800	500
9	900	600

Abbildung XLII: StackGres Testing - DB Zugriff

Allerdings werden längere Transaktionen geschlossen:

## Diplomarbeit

Citus cluster: sg-healing-test						
Group	Member	Host	Role	State	TL	Lag in MB
0	sg-healing-test-coord-0	198.18.1.106:7433	Sync Standby	streaming	3	0
0	sg-healing-test-coord-1	198.18.0.59:7433	Sync Standby	streaming	3	0
0	sg-healing-test-coord-2	198.18.2.90:7433	Leader	running	3	
1	sg-healing-test-shard0-0	198.18.0.106:7433	Replica	streaming	3	0
1	sg-healing-test-shard0-1	198.18.1.71:7433	Replica	streaming	3	0
1	sg-healing-test-shard0-2	198.18.2.3:7433	Leader	running	3	
2	sg-healing-test-shard1-0	198.18.1.155:7433	Replica	streaming	2	0
2	sg-healing-test-shard1-1	198.18.0.163:7433	Replica	streaming	2	0
2	sg-healing-test-shard1-2	198.18.2.253:7433	Leader	running	2	
3	sg-healing-test-shard2-0	198.18.2.31:7433	Leader	running	3	
3	sg-healing-test-shard2-1	198.18.0.186:7433	Replica	streaming	3	0
3	sg-healing-test-shard2-2	198.18.1.163:7433	Replica	streaming	3	0

Abbildung XLIII: StackGres Testing - Connection Timeout

## VII.III YugabyteDB

## VIII Exkurs Architekturen - Umsysteme und Prinzipien

### VIII.I Raft-Konsensus

### VIII.II local-path-provisioner

## IX Python Utils

### IX.I zotero.py

```

1 import json
2 import pybtex
3 import requests
4 import os
5 from pybtex.database import BibliographyData, Entry, Person
6 from dateutil.parser import parse
7 import math
8 import yaml
9
10 # Load the Configurations
11 def load_configuration(zotero_conf_filename):
12     zotero_bibtex_config = dict()
13     zotero_conf_dir = os.path.join(os.path.dirname(os.getcwd()), 'source', 'configuration')
14
15     yaml_path = os.path.join(zotero_conf_dir, zotero_conf_filename)
16
17     with open(yaml_path, "r") as file:
18         zotero_bibtex_config = yaml.load(file, Loader=yaml.FullLoader)
19

```

## Diplomarbeit

```

20     return zotero_bibtex_config
21 def downlaod_zotero_datas(URL, API_KEY):
22     zotero_result = list
23     response = requests.get(URL, headers={'Zotero-API-Key': API_KEY})
24     response = response.json()
25     zotero_raw = json.dumps(response, ensure_ascii=False) # json.loads(response)
26     zotero_result = json.loads(zotero_raw)
27     return zotero_result
28
29 # Get the bibtex Datas from Zotero
30 def get_data(zotero_bibtex_config):
31     result_limit = int(zotero_bibtex_config.get('result_limit'))
32     access_type = zotero_bibtex_config.get('access_type')
33     zotero_access_id = zotero_bibtex_config.get('zotero_access_id')
34     collection_id = zotero_bibtex_config.get('collection_id')
35     API_KEY = zotero_bibtex_config.get('api_key')
36     zotero_data = list()
37     URL = 'https://api.zotero.org/' + str(access_type) + '/' + str(
38         zotero_access_id) + '/collections/' + str(
39             collection_id) + '/items?limit=1?format=json?sort=dateAdded?direction=asc'
40
41     response = requests.get(URL, headers={'Zotero-API-Key': API_KEY})
42
43     header_dict = response.headers
44     total_elemets = int(header_dict.get('Total-Results'), 0)
45
46     if total_elemets < result_limit:
47         URL_ALL_ITEMS = 'https://api.zotero.org/' + str(access_type) + '/' + str(
48             zotero_access_id) + '/collections/' + str(collection_id) + '/items?
49             limit=' + str(
50                 result_limit) + '?format=json?sort=dateAdded?direction=asc'
51         zotero_result = downlaod_zotero_datas(URL_ALL_ITEMS, API_KEY)
52
53         zotero_data.extend(zotero_result)
54     else:
55         runs = int(math.ceil(total_elemets / result_limit))
56         index = 0
57         start_index = 0
58         while index < runs:
59             URL_Separated = 'https://api.zotero.org/' + str(access_type) + '/' +
60             str(
61                 zotero_access_id) + '/collections/' + str(collection_id) + '/items?
62             ?limit=' + str(
63                 result_limit) + '?format=json?sort=dateAdded?direction=asc' + '&
64             start=' + str(start_index)
65             zotero_result = downlaod_zotero_datas(URL_Separated, API_KEY)
66
67             zotero_data.extend(zotero_result)

```

## Diplomarbeit

```
63
64         start_index += result_limit
65         index += 1
66
67     return zotero_data
68
69 # Convert String to Datetime
70 def convert_to_datetime(input_str, parserinfo=None):
71     return parse(input_str, parserinfo=parserinfo)
72
73 # Get Dates from Datetime
74 def get_dates(date, bibtex_item_type, bibtex_month_attributes):
75     dated_date = convert_to_datetime(date)
76     return_value = dict()
77     if bibtex_item_type in bibtex_month_attributes:
78         year = dated_date.year
79         month = dated_date.month
80         return_value = {'year': year, 'month': month}
81     else:
82         year = dated_date.year
83         return_value = {'year': year}
84
85     return return_value
86
87 # Split Creators into biblatex Creators
88 def split_creators(creators):
89     if creators != []:
90
91         creatorlist = ''
92         for index, creator in enumerate(creators):
93             type = creator.get('creatorType')
94             firstname = creator.get('firstName')
95             lastname = creator.get('lastName')
96             name = creator.get('name')
97             if type == 'author':
98
99                 if name and not (firstname or lastname):
100                     creatorlist = creatorlist + name
101                     if index != len(creators) - 1:
102                         creatorlist = creatorlist + ', and '
103                     else:
104                         creatorlist = creatorlist + lastname + ', ' + firstname
105                         if index != len(creators) - 1:
106                             creatorlist = creatorlist + ', and '
107             else:
108                 creatorlist = 'unknown author'
109
110     bib_entry = 'author=' + '"' + creatorlist + '"'
```

```
111
112     return bib_entry
113
114 #     Write the *.bib File
115 def write_bibliography(zotero_data, zotero_bibtex_config):
116     keystore_file = zotero_bibtex_config.get('keystore_file')
117     keystore_path = zotero_bibtex_config.get('keystore_filepath')
118     tex_dir = os.path.join(os.path.dirname(os.getcwd()), keystore_path)
119
120     yaml_path = os.path.join(tex_dir, keystore_file)
121
122     with open(yaml_path, "r") as file:
123         zotero_bibtex_keys = yaml.load(file, Loader=yaml.FullLoader)
124
125     zotero_bibtex_keys_specials = {
126         'thesis': {'phdthesis': ['dissertation', 'phd', 'doctorial', 'doctor', 'doktor', 'doktorarbeit'],
127                     'masterthesis': ['ma', 'master', 'masters']},
128     }
129     zotero_bibtex_attributes_special = {
130         'date': 'get_dates',
131         'creators': 'split_creators'
132     }
133     bibtex_month_attributes = ['booklet', 'mastersthesis', 'phdthesis', 'techreport']
134
#     Bibliography
135     # tex_dir = os.path.join(os.path.dirname(os.getcwd()), 'source')
136     bibtex_path = zotero_bibtex_config.get('bibtex_filepath')
137     tex_dir = os.path.join(os.path.dirname(os.getcwd()), bibtex_path)
138     # tex_dir = os.path.join(os.getcwd(), 'src', 'content')
139     # file_name = 'Datenbank_Projektauftrag_Michael_Graber.bib'
140     file_name = zotero_bibtex_config.get('bibtex_filename')
141
142     file_path = os.path.join(tex_dir, file_name)
143
144     # bib_datas = BibliographyData()
145     listKeys = list()
146     bib_data = ''
147     for zotero_items in zotero_data:
148         biblio_item = zotero_items.get('data')
149         itemkeys = biblio_item.keys()
150         listKeys.extend(biblio_item.keys())
151         zotero_item_key = biblio_item.get('key')
152         zotero_item_title = biblio_item.get('title')
153         zotero_item_nameofact = biblio_item.get('nameOfAct')
154         zotero_item_nameofcase = biblio_item.get('caseName')
155         zotero_item_subject = biblio_item.get('subject')
156         zotero_item_type = biblio_item.get('itemType')
```

```
157
158     # some item types have no titles
159     # set the special names instead of the title
160     if zotero_item_title:
161         bibtex_item_titel = zotero_item_title
162     else:
163         if zotero_item_type == 'statute':
164             biblio_item['title'] = zotero_item_nameofact
165             bibtex_item_titel = zotero_item_nameofact
166         elif zotero_item_type == 'case':
167             biblio_item['title'] = zotero_item_nameofcase
168             bibtex_item_titel = zotero_item_nameofcase
169         elif zotero_item_type == 'email':
170             biblio_item['title'] = zotero_item_subject
171             bibtex_item_titel = zotero_item_subject
172
173         if zotero_item_type == 'thesis':
174             master_list = zotero_bibtex_keys_specials.get(zotero_item_type).get('
175             masterthesis')
176             phd_list = zotero_bibtex_keys_specials.get(zotero_item_type).get('
177             phdthesis')
178
178         # First Master thesis
179         if any(item in bibtex_item_titel for item in master_list):
180             bibtex_item_key = 'masterthesis'
181         # Second PHD Thesis
182         elif any(item in bibtex_item_titel for item in phd_list):
183             bibtex_item_key = 'phdthesis'
184         else:
185             bibtex_item_key = 'masterthesis',
186         else:
187             if zotero_bibtex_keys.get(zotero_item_type).get('key'):
188                 bibtex_item_key = zotero_bibtex_keys.get(zotero_item_type).get('
189                 key')
190             else:
191                 bibtex_item_key = 'misc'
192
193         # get all Keys for the zotero item type
194         entryset = '\n'
195         entry = ''
196
197         zotero_item_attributes = zotero_bibtex_keys.get(zotero_item_type).get('
198         attributes').keys()
199         item_attributes = sorted(zotero_item_attributes, reverse=True)
200
201         for index, item_attribute in enumerate(item_attributes):
202             bibtex_item_attribute = zotero_bibtex_keys.get(zotero_item_type).get('
203             attributes').get(item_attribute)
```

```
200     zotero_item_value = biblio_item.get(item_attribute)
201     zotero_item_value_extra = '',
202     bibtex_item_attribute_extra = '',
203
204     # Special Cases
205     if bibtex_item_attribute == 'SPECIALCHECK' and zotero_item_value not
206     in ['', None]:
207         bibtex_special_attribute = zotero_bibtex_attributes_special.get(
208             item_attribute)
209
210         match bibtex_special_attribute:
211             case 'get_dates':
212                 zotero_item_value = get_dates(zotero_item_value,
213                     bibtex_item_key, bibtex_month_attributes)
214                 if zotero_item_value.get('month'):
215                     zotero_item_value_extra = zotero_item_value.get('month')
216
217             case 'month':
218                 bibtex_item_attribute_extra = 'month'
219
220             case 'year':
221                 bibtex_item_attribute = 'year'
222             case 'split_creators':
223                 authors = split_creators(zotero_item_value)
224                 entryset = entryset + authors
225
226             elif bibtex_item_attribute == 'howpublished':
227                 if zotero_item_value not in ['', None, []]:
228                     zotero_item_value = '\url{' + zotero_item_value + '}'
229
230
231             if bibtex_item_attribute not in ['', 'None', 'author', 'SPECIALCHECK']
232             and zotero_item_value not in ['', None, []]:
233                 if zotero_item_value_extra:
234
235                     if type(zotero_item_value_extra) == "string":
236                         entryset = entryset + str(bibtex_item_attribute_extra) + ,
237                         ='"' + str(zotero_item_value_extra) + '"',
238                     else:
239                         entryset = entryset + str(bibtex_item_attribute_extra) + ,
240                         =' + str(zotero_item_value_extra)
241
242                     if index != len(item_attributes) - 1:
243                         entryset = entryset + ',\n'
244                     else:
245                         entryset = entryset + '\n'
246
247                     if type(zotero_item_value) == str and not zotero_item_value.
248                     isnumeric():
249                         entryset = entryset + str(bibtex_item_attribute) + '=',
250                         + str(zotero_item_value) + '"'
```

```

239         else:
240             entryset = entryset + str(bibtex_item_attribute) + '=' + str(
241                 zotero_item_value)
242
243             if index != len(item_attributes) - 1:
244                 entryset = entryset + ',\n'
245             else:
246                 entryset = entryset + '\n'
247
248             # create the Entry
249             entry = '@' + bibtex_item_key + '{' + zotero_item_key + ',\n'
250             entry = entry + entryset + '}'
251             bib_data = bib_data + '\n' + entry
252
253             # parse String to pybtex.database Object
254             # bib_datas = pybtex.database.parse_string(bib_data, bib_format="bibtex",
255             # encoding='ISO-8859-1')
256             bib_datas = pybtex.database.parse_string(bib_data, bib_format="bibtex",
257             encoding='Iutf-8')
258             # Save pybtex.database to file
259             # BibliographyData.to_file(bib_datas, file_path, bib_format="bibtex", encoding
260             # ='ISO-8859-1')
261             BibliographyData.to_file(bib_datas, file_path, bib_format="bibtex", encoding=
262             'utf-8')
263
264
265             zotero_bibtex_config = load_configuration('zotero_bibtex_configuration.yaml')
266             zotero_data = get_data(zotero_bibtex_config)
267             write_bibliography(zotero_data, zotero_bibtex_config)

```

Listing 35: Python LaTex - zotero.py - Zotero BibLaTex Importer

## IX.II zotero\_bibtex\_configuration.yaml

```

1 result_limit: 100
2 access_type: "groups"
3 zotero_access_id: "5222465"
4 collection_id: "PC3BW6EP"
5 api_key: "6Xgb3XhGjQXwA8NuZgu3bw3s"
6 keystore_file: "zotero_biblatek_keystore.yaml"
7 keystore_filepath: "source/configuration"
8 bibtex_filepath: "source"
9 bibtex_filename: "Diplomarbeit_Michael_Graber.bib"

```

Listing 36: Python LaTex - zotero\_bibtex\_configuration.yaml - Konfigurationsdatei - Zotero BibLaTex Importer

## Diplomarbeit

### IX.III zotero\_biblatex\_keystore.yaml

```
1---
2artwork:
3  key: misc
4  attributes:
5    title: title
6    date: SPECIALCHECK
7    creators: SPECIALCHECK
8    url: howpublished
9    extra: note
10audioRecording:
11  key: misc
12  attributes:
13    title: title
14    date: SPECIALCHECK
15    creators: SPECIALCHECK
16bill:
17  key: misc
18  attributes:
19    title: title
20    date: SPECIALCHECK
21    creators: SPECIALCHECK
22    url: howpublished
23    extra: note
24blogPost:
25  key: misc
26  attributes:
27    title: title
28    date: SPECIALCHECK
29    creators: SPECIALCHECK
30    url: howpublished
31    extra: note
32book:
33  key: book
34  attributes:
35    title: title
36    date: SPECIALCHECK
37    creators: SPECIALCHECK
38    publisher: publisher
39    place: address
40bookSection:
41  key: inbook
42  attributes:
43    title: title
44    date: SPECIALCHECK
45    creators: SPECIALCHECK
46    pages: pages
```

## Diplomarbeit

```
47     publisher: publisher
48     place: address
49     bookTitle: booktitle
50 case:
51   key: misc
52   attributes:
53     title: title
54     date: SPECIALCHECK
55     creators: SPECIALCHECK
56     url: howpublished
57     extra: note
58 conferencePaper:
59   key: inproceedings
60   attributes:
61     title: title
62     date: SPECIALCHECK
63     creators: SPECIALCHECK
64     series: series
65     proceedingsTitle: booktitle
66     publisher: publisher
67     pages: pages
68     place: address
69 dictionaryEntry:
70   key: misc
71   attributes:
72     title: title
73     date: SPECIALCHECK
74     creators: SPECIALCHECK
75     url: howpublished
76     extra: note
77 document:
78   key: misc
79   attributes:
80     title: title
81     date: SPECIALCHECK
82     creators: SPECIALCHECK
83     url: howpublished
84     extra: note
85 email:
86   key: misc
87   attributes:
88     title: title
89     date: SPECIALCHECK
90     creators: SPECIALCHECK
91     url: howpublished
92     extra: note
93 encyclopediaArticle:
94   key: misc
```

## Diplomarbeit

```
95 attributes:  
96   title: title  
97   date: SPECIALCHECK  
98   creators: SPECIALCHECK  
99   url: howpublished  
100  extra: note  
101 film:  
102  key: misc  
103  attributes:  
104    title: title  
105    date: SPECIALCHECK  
106    creators: SPECIALCHECK  
107    url: howpublished  
108    extra: note  
109 forumPost:  
110  key: misc  
111  attributes:  
112    title: title  
113    date: SPECIALCHECK  
114    creators: SPECIALCHECK  
115    url: howpublished  
116    extra: note  
117 hearing:  
118  key: misc  
119  attributes:  
120    title: title  
121    date: SPECIALCHECK  
122    creators: SPECIALCHECK  
123    url: howpublished  
124    extra: note  
125 instantMessage:  
126  key: misc  
127  attributes:  
128    title: title  
129    date: SPECIALCHECK  
130    creators: SPECIALCHECK  
131    url: howpublished  
132    extra: note  
133 interview:  
134  key: misc  
135  attributes:  
136    title: title  
137    date: SPECIALCHECK  
138    creators: SPECIALCHECK  
139    url: howpublished  
140    extra: note  
141 journalArticle:  
142  key: article
```

## Diplomarbeit

```
143 attributes:  
144   title: title  
145   date: SPECIALCHECK  
146   creators: SPECIALCHECK  
147   volume: volume  
148   pages: pages  
149   seriesNumber: number  
150   seriesTitle: journal  
151   url: url  
152 letter:  
153   key: misc  
154   attributes:  
155     title: title  
156     date: SPECIALCHECK  
157     creators: SPECIALCHECK  
158     url: howpublished  
159     extra: note  
160 magazineArticle:  
161   key: article  
162   attributes:  
163     title: title  
164     date: SPECIALCHECK  
165     creators: SPECIALCHECK  
166     volume: volume  
167     pages: pages  
168     seriesNumber: number  
169     seriesTitle: journal  
170     url: url  
171 manuscript:  
172   key: unpublished  
173   attributes:  
174     title: title  
175     date: SPECIALCHECK  
176     creators: SPECIALCHECK  
177 map:  
178   key: misc  
179   attributes:  
180     title: title  
181     date: SPECIALCHECK  
182     creators: SPECIALCHECK  
183     url: howpublished  
184     extra: note  
185 newspaperArticle:  
186   key: article  
187   attributes:  
188     title: title  
189     date: SPECIALCHECK  
190     creators: SPECIALCHECK
```

## Diplomarbeit

```
191     volume: volume
192     pages: pages
193     seriesNumber: number
194     seriesTitle: journal
195     url: url
196 patent:
197     key: misc
198     attributes:
199         title: title
200         date: SPECIALCHECK
201         creators: SPECIALCHECK
202         url: howpublished
203         extra: note
204 podcast:
205     key: misc
206     attributes:
207         title: title
208         date: SPECIALCHECK
209         creators: SPECIALCHECK
210         url: howpublished
211         extra: note
212 presentation:
213     key: misc
214     attributes:
215         title: title
216         date: SPECIALCHECK
217         creators: SPECIALCHECK
218         url: howpublished
219         extra: note
220 radioBroadcast:
221     key: misc
222     attributes:
223         title: title
224         date: SPECIALCHECK
225         creators: SPECIALCHECK
226         url: howpublished
227         extra: note
228 report:
229     techreport: misc
230     attributes:
231         title: title
232         date: SPECIALCHECK
233         creators: SPECIALCHECK
234         url: howpublished
235         extra: note
236 software:
237     key: misc
238     attributes:
```

## Diplomarbeit

```
239     title: title
240     date: SPECIALCHECK
241     creators: SPECIALCHECK
242     url: howpublished
243     extra: note
244 computerProgram:
245     key: misc
246     attributes:
247         title: title
248         date: SPECIALCHECK
249         creators: SPECIALCHECK
250         url: howpublished
251         extra: note
252 statute:
253     key: misc
254     attributes:
255         title: title
256         date: SPECIALCHECK
257         creators: SPECIALCHECK
258         url: howpublished
259         extra: note
260 tvBroadcast:
261     key: misc
262     attributes:
263         title: title
264         date: SPECIALCHECK
265         creators: SPECIALCHECK
266         url: howpublished
267         extra: note
268 videoRecording:
269     key: misc
270     attributes:
271         title: title
272         date: SPECIALCHECK
273         creators: SPECIALCHECK
274         url: howpublished
275         extra: note
276 webpage:
277     key: misc
278     attributes:
279         title: title
280         date: SPECIALCHECK
281         creators: SPECIALCHECK
282         url: howpublished
283         extra: note
284 attachment:
285     key: misc
286     attributes:
```

## Diplomarbeit

```
287     title: title
288     date: SPECIALCHECK
289     creators: SPECIALCHECK
290     url: howpublished
291     extra: note
292
293 note:
294   key: misc
295   attributes:
296     title: title
297     date: SPECIALCHECK
298     creators: SPECIALCHECK
299     url: howpublished
300     extra: note
301
302 standard:
303   key: misc
304   attributes:
305     title: title
306     date: SPECIALCHECK
307     creators: SPECIALCHECK
308     url: howpublished
309     extra: note
310
311 preprint:
312   key: misc
313   attributes:
314     title: title
315     date: SPECIALCHECK
316     creators: SPECIALCHECK
317     url: howpublished
318     extra: note
319
320 dataset:
321   key: misc
322   attributes:
323     title: title
324     date: SPECIALCHECK
325     creators: SPECIALCHECK
326     url: howpublished
327     extra: note
328
329 thesis:
330   key: thesis
331   attributes:
332     title: title
333     date: SPECIALCHECK
334     creators: SPECIALCHECK
335     place: address
336     university: school
```

Listing 37: Python LaTex - zotero\_biblatex\_keystore.yaml - x-y-Achse Konfigurationsdatei - Zotero BibLaTex Importer

## Diplomarbeit

### IX.IV riskmatrix.py

```

1 import os
2 import matplotlib.pyplot as plt
3 import yaml
4
5 # Load Configurations
6 def load_configuration(riskmatrix_conf_filename):
7     riskmatrix_config = dict()
8
9     riskmatrix_conf_dir = os.path.join(os.path.dirname(os.getcwd()), 'source', 'configuration')
10    yaml_path = os.path.join(riskmatrix_conf_dir, riskmatrix_conf_filename)
11
12    with open(yaml_path, "r") as file:
13        riskmatrix_config = yaml.load(file, Loader=yaml.FullLoader)
14
15    return riskmatrix_config
16
17 # Load x-y axis tuples
18 def load_xy_axis_tuples(riskmatrix_config):
19    startpath = riskmatrix_config.get('riskmatrix').get('startpath')
20    riskmatrix_xy_axis_tuples_dir = riskmatrix_config.get('riskmatrix').get('configfile_path')
21    riskmatrix_xy_axis_tuples_config = riskmatrix_config.get('riskmatrix').get('configfile_name')
22
23    if startpath == 'homedir':
24        directory = os.path.join(os.getcwd(), riskmatrix_xy_axis_tuples_dir)
25    else: # parentdir
26        directory = os.path.join(os.path.dirname(os.getcwd()),
27            riskmatrix_xy_axis_tuples_dir)
28
29    riskmatrix_xy_axis_tuples_path = os.path.join(directory,
30        riskmatrix_xy_axis_tuples_config)
31    riskmatrix_xy_axis_tuples = dict()
32    riskmatrix_xy_axis_tuples_aux = dict()
33
34    with open(riskmatrix_xy_axis_tuples_path, "r") as file:
35        riskmatrix_xy_axis_tuples_aux = yaml.load(file, Loader=yaml.FullLoader)
36
37    for string_key in riskmatrix_xy_axis_tuples_aux:
38        value = riskmatrix_xy_axis_tuples_aux.get(string_key)
39        int_key = eval(string_key)
40        riskmatrix_xy_axis_tuples.update({int_key:value})
41
42    return riskmatrix_xy_axis_tuples
43
44 # Load Data from csv

```

## Diplomarbeit

```

42 def get_data(data_path):
43
44     with open(data_path) as f:
45         csv_list = [[val.strip() for val in r.split(",")] for r in f.readlines()]
46
47     (_, *header), *data = csv_list
48     datas = {}
49     for row in data:
50         key, *values = row
51         datas[key] = {key: value for key, value in zip(header, values)}
52
53     return datas
54
55 # Generate Riskmatrix Image
56 #def riskmatrix(risk, conf, matrix):
57 def riskmatrix(conf, matrix):
58     risks = conf.get('risk_inventory')
59     for risk_conf in risks:
60         # get the risk config datas
61         startpath = conf.get('risks').get(risk_conf).get('startpath')
62         destination = conf.get('risks').get(risk_conf).get('destination_path')
63         imagename = conf.get('risks').get(risk_conf).get('imagename')
64         datafilename = conf.get('risks').get(risk_conf).get('datafile')
65         itemname = conf.get('risks').get(risk_conf).get('itemname')
66         x_axis_title = conf.get('risks').get(risk_conf).get('x-axis-title')
67         y_axis_title = conf.get('risks').get(risk_conf).get('y-axis-title')
68         title = conf.get('risks').get(risk_conf).get('title')
69         bubble_standard_size = conf.get('risks').get(risk_conf).get('bubble-
70             standard-size')
71
72         # Identify the index of the axes
73         green = conf.get('risks').get(risk_conf).get('settings').get('green-boxes',
74     )
75         yellow = conf.get('risks').get(risk_conf).get('settings').get('yellow-
76             boxes')
77         orange = conf.get('risks').get(risk_conf).get('settings').get('orange-
78             boxes')
79         red = conf.get('risks').get(risk_conf).get('settings').get('red-boxes')
80
81         if startpath == 'homedir':
82             directory = os.path.join(os.getcwd(), destination)
83         else: # parentdir
84             directory = os.path.join(os.path.dirname(os.getcwd()), destination)
85
86         data_path = os.path.join(directory, datafilename)
87         image_path = os.path.join(directory, imagename)
88
89         # get the Datas as dirct

```

## Diplomarbeit

```
86     datas = get_data(data_path)
87
88     fig = plt.figure()
89     plt.subplots_adjust(wspace=0, hspace=0)
90     plt.xticks([])
91     plt.yticks([])
92     plt.xlim(0, 5)
93     plt.ylim(0, 5)
94     plt.xlabel(x_axis_title)
95     plt.ylabel(y_axis_title)
96     plt.title(title)
97
98     # This example is for a 5 * 5 matrix
99     nrows = 5
100    ncols = 5
101    axes = [fig.add_subplot(nrows, ncols, r * ncols + c + 1) for r in range(0,
102                           nrows) for c in range(0, ncols)]
103
104    # remove the x and y ticks
105    for ax in axes:
106        ax.set_xticks([])
107        ax.set_yticks([])
108        ax.set_xlim(0, 5)
109        ax.set_ylim(0, 5)
110
111    # Add background colors
112    # This has been done manually for more fine-grained control
113    # Run the loop below to identify the indice of the axes
114    for _ in green:
115        axes[_].set_facecolor('green')
116
117    for _ in yellow:
118        axes[_].set_facecolor('yellow')
119
120    for _ in orange:
121        axes[_].set_facecolor('orange')
122
123    for _ in red:
124        axes[_].set_facecolor('red')
125
126    # run through datas and generate axis datas
127    dict_bubble_axis = dict()
128    bubble_axis = list()
129    for datasets in datas:
130        # get the datas
131        riskid = datas.get(datasets).get('risk-id')
132        x_axis = int(datas.get(datasets).get('x-axis'))
133        y_axis = int(datas.get(datasets).get('y-axis'))
```

```

133     axis_point = matrix.get((x_axis, y_axis))
134     x_axis_text = float(datas.get(datasets).get('x-axis-text'))
135     y_axis_text = float(datas.get(datasets).get('y-axis-text'))
136     x_axis_bubble = float(datas.get(datasets).get('x-axis-bubble'))
137     y_axis_bubble = float(datas.get(datasets).get('y-axis-bubble'))
138     bubble_axis.append(axis_point)
139
140     # merge risks if two or more risks share the same axispoint
141     if dict_bubble_axis.get(axis_point):
142         risktag = dict_bubble_axis.get(axis_point).get('risk')
143         risktag = risktag + '/' + riskid
144         x_axis_text = x_axis_text + 0.25
145         y_axis_text = y_axis_text - 0.5
146         bubble_size = bubble_standard_size * 2
147     else:
148         risktag = itemname + riskid
149         bubble_size = bubble_standard_size
150     dict_axis_value = dict()
151
152     dict_axis_value['risk'] = risktag
153     dict_axis_value['x-axis-text'] = x_axis_text
154     dict_axis_value['y-axis-text'] = y_axis_text
155     dict_axis_value['x-axis-bubble'] = x_axis_bubble
156     dict_axis_value['y-axis-bubble'] = y_axis_bubble
157     dict_axis_value['size'] = bubble_size
158     dict_bubble_axis[axis_point] = dict_axis_value
159
160     # cleanup the list, remove duplicated entries
161     bubble_axis = set(bubble_axis)
162
163     # plot the bubbles and texts in the bubbles
164     for axispoint in bubble_axis:
165         axes[axispoint].scatter(dict_bubble_axis[axispoint]['x-axis-bubble'],
166                               dict_bubble_axis[axispoint]['y-axis-bubble'],
167                               dict_bubble_axis[axispoint]['size'], alpha=1)
168         axes[axispoint].text(dict_bubble_axis[axispoint]['x-axis-text'],
169                               dict_bubble_axis[axispoint]['y-axis-text'], s=
170                               dict_bubble_axis[axispoint]['risk'],
171                               va='bottom', ha='center')
172
173     # save the plot as image
174     plt.savefig(image_path)
175
176 riskmatrix_config = load_configuration('riskmatrix_plotter_conf.yaml')
177 riskmatrix_xy_axis_tuples = load_xy_axis_tuples(riskmatrix_config)
178 riskmatrix(riskmatrix_config, riskmatrix_xy_axis_tuples)

```

Listing 38: Python LaTex - riskmatrix.py - Risikomatrizen

IX.V riskmatrix\_plotter\_conf.yaml

```
1 risk_inventory:
2   - "postgresql"
3   - "project"
4   - "Postgresql-massnahme"
5   - "Project-massnahme"
6 riskmatrix:
7   startpath: "parentdir"
8   configfile_path: "source/configuration"
9   configfile_name: "riskmatrix_xy_axis_tuple_matrix.yaml"
10 risks:
11   postgresql:
12     riskid: "postgresql"
13     startpath: "parentdir"
14     destination_path: "source/riskmatrix"
15     imagename: "riskmatrixproblem.png"
16     datafile_path: "source/tables"
17     datafile: "riskmatrixproblem.csv"
18     itemname: "R"
19     x-axis-title: "Schadensausmass (SM)"
20     y-axis-title: "Eintrittswahrscheinlichkeit (WS)"
21     title: "Risiko Cockpit PostgreSQL Datenbanken KSGR"
22     bubble-standard-size: 1000
23     settings:
24       green-boxes:
25         - 10
26         - 15
27         - 16
28         - 20
29         - 21
30       yellow-boxes:
31         - 0
32         - 5
33         - 6
34         - 11
35         - 17
36         - 22
37         - 23
38       orange-boxes:
39         - 1
40         - 2
41         - 7
42         - 12
43         - 13
44         - 18
45         - 19
46         - 24
```

## Diplomarbeit

```
47     red-boxes:  
48         - 3  
49         - 4  
50         - 8  
51         - 9  
52         - 14  
53     project:  
54         riskid: "project"  
55         startpath: "parentdir"  
56         destination_path: "source/riskmatrix"  
57         imagename: "riskmatrix-project.png"  
58         datafile_path: "source/tables"  
59         datafile: "riskmatrix-project.csv"  
60         itemname: "R"  
61         x-axis-title: "Schadensausmass (SM)"  
62         y-axis-title: "Eintrittswahrscheinlichkeit (WS)"  
63         title: "Risiko Cockpit Projekt"  
64         bubble-standard-size: 1000  
65     settings:  
66         green-boxes:  
67             - 10  
68             - 15  
69             - 16  
70             - 20  
71             - 21  
72         yellow-boxes:  
73             - 0  
74             - 5  
75             - 6  
76             - 11  
77             - 17  
78             - 22  
79             - 23  
80         orange-boxes:  
81             - 1  
82             - 2  
83             - 7  
84             - 12  
85             - 13  
86             - 18  
87             - 19  
88             - 24  
89         red-boxes:  
90             - 3  
91             - 4  
92             - 8  
93             - 9  
94             - 14
```

## Diplomarbeit

```
95 Postgresql-massnahme:
96   riskid: "Postgresql-massnahme"
97   startpath: "parentdir"
98   destination_path: "source/riskmatrix"
99   imagename: "Riskmatrixproblem-massnahmen.png"
100  datafile_path: "source/tables"
101  datafile: "riskmatrixproblem-massnahmen.csv"
102  itemname: "R"
103  x-axis-title: "Schadensausmass (SM)"
104  y-axis-title: "Eintrittswahrscheinlichkeit (WS)"
105  title: "Risiko Cockpit PostgreSQL Datenbanken KSGR - Massnahme"
106  bubble-standard-size: 1000
107  settings:
108    green-boxes:
109      - 10
110      - 15
111      - 16
112      - 20
113      - 21
114    yellow-boxes:
115      - 0
116      - 5
117      - 6
118      - 11
119      - 17
120      - 22
121      - 23
122    orange-boxes:
123      - 1
124      - 2
125      - 7
126      - 12
127      - 13
128      - 18
129      - 19
130      - 24
131    red-boxes:
132      - 3
133      - 4
134      - 8
135      - 9
136      - 14
137 Project-massnahme:
138   riskid: "Project-massnahme"
139   startpath: "parentdir"
140   destination_path: "source/riskmatrix"
141   imagename: "Riskmatrix-project-massnahmen.png"
142   datafile_path: "source/tables"
```

```

143     datafile: "riskmatrix-project-massnahmen.csv"
144     itemname: "R"
145     x-axis-title: "Schadensausmass (SM)"
146     y-axis-title: "Eintrittswahrscheinlichkeit (WS)"
147     title: "Risiko Cockpit Projekt - Massnahme"
148     bubble-standard-size: 1000
149     settings:
150         green-boxes:
151             - 10
152             - 15
153             - 16
154             - 20
155             - 21
156         yellow-boxes:
157             - 0
158             - 5
159             - 6
160             - 11
161             - 17
162             - 22
163             - 23
164         orange-boxes:
165             - 1
166             - 2
167             - 7
168             - 12
169             - 13
170             - 18
171             - 19
172             - 24
173         red-boxes:
174             - 3
175             - 4
176             - 8
177             - 9
178             - 14

```

Listing 39: Python LaTex - riskmatrix\_plotter\_conf.yaml - Konfigurationsdatei - Risikomatrizen

## IX.VI riskmatrix\_xy\_axis\_tuple\_matrix.yaml

```

1 #Matrix
2 #This Matrix translate the x/y axis from a given risk matrix csv to the axispoint.
3 #
4 #The key of each axispoint is an integer tupel (x, y)
5 #So, you can access the axis point this way:
6 #<axispoint> = matrix.get((<x_axis>, <y_axis>))

```

## Diplomarbeit

```

7 (1, 1): 20
8 (1, 2): 15
9 (1, 3): 10
10 (1, 4): 5
11 (1, 5): 0
12 (2, 1): 21
13 (2, 2): 16
14 (2, 3): 11
15 (2, 4): 6
16 (2, 5): 1
17 (3, 1): 22
18 (3, 2): 17
19 (3, 3): 12
20 (3, 4): 7
21 (3, 5): 2
22 (4, 1): 23
23 (4, 2): 18
24 (4, 3): 13
25 (4, 4): 8
26 (4, 5): 3
27 (5, 1): 24
28 (5, 2): 19
29 (5, 3): 14
30 (5, 4): 9
31 (5, 5): 4

```

Listing 40: Python LaTex - riskmatrix\_xy\_axis\_tuple\_matrix.yaml - Konfigurationsdatei - Risikomatrizen - X-Y-Achsen Tuples

## IX.VII cost\_benefit\_diagram.py

```

1 import os
2 import matplotlib.pyplot as plt
3 import yaml
4
5 # Get the Configuration
6 def load_configuration():
7     cost_benefit_config = dict()
8     cbd_conf_filename = 'scatter_plotter_conf.yaml'
9     cbd_conf_dir = os.path.join(os.path.dirname(os.getcwd()), 'source', 'configuration')
10    yaml_path = os.path.join(cbd_conf_dir, cbd_conf_filename)
11
12    with open(yaml_path, "r") as file:
13        cost_benefit_config = yaml.load(file, Loader=yaml.FullLoader)
14
15    return cost_benefit_config

```

```

16 # Get the Datas
17 def get_data(cost_benefit_config):
18     # Config Variables
19     startpath = cost_benefit_config.get('startpath')
20     destination = cost_benefit_config.get('desitination_path')
21     datafilename = cost_benefit_config.get('datafile')
22
23     if startpath == 'homendir':
24         directory = os.path.join(os.getcwd(), destination)
25     else: # parentdir
26         directory = os.path.join(os.path.dirname(os.getcwd()), destination)
27
28     # get the Datas as dirct
29     data_path = os.path.join(directory, datafilename)
30
31     # load datas from csv into dict
32     with open(data_path) as f:
33         csv_list = [[val.strip() for val in r.split(",")] for r in f.readlines()]
34
35     (_, *header), *data = csv_list
36     datas = {}
37     for row in data:
38         key, *values = row
39         datas[key] = {key: value for key, value in zip(header, values)}
40
41     cost_benefit_data = []
42     for key, value in datas.items():
43         variant_name = value['variant_name']
44         x_axis = int(value['x-axis'])
45         y_axis = int(value['y-axis'])
46         cost_benefit_data[variant_name] = (x_axis, y_axis)
47
48     return cost_benefit_data
49
50 # Plot the Datas
51 def cost_benefit_diagram(cost_benefit_config, cost_benefit_data):
52     # Config Variables
53     startpath = cost_benefit_config.get('startpath')
54     destination = cost_benefit_config.get('desitination_path')
55     imagename = cost_benefit_config.get('imagename')
56
57     if startpath == 'homendir':
58         directory = os.path.join(os.getcwd(), destination)
59     else: # parentdir
60         directory = os.path.join(os.path.dirname(os.getcwd()), destination)
61
62     # get the Datas as dirct
63     data_path = os.path.join(directory, imagename)

```

## Diplomarbeit

```

64
65     # Extract the Datas
66     labels, values = zip(*cost_benefit_data.items())
67     x, y = zip(*values)
68
69     # Create Scatter-Diagram
70     plt.scatter(x, y, color=cost_benefit_config.get('scatter-point-color'))
71
72     # X-Lines
73     plt.axhline(y=cost_benefit_config.get('y-axis-line-pos'), color=
74     cost_benefit_config.get('y-axis-line-color'), linestyle=cost_benefit_config.
75     get('y-axis-line-type'), label=cost_benefit_config.get('y-axis-line-label'))
76
77     # Y-Lines
78     plt.axvline(x=cost_benefit_config.get('x-axis-line-pos'), color=
79     cost_benefit_config.get('x-axis-line-color'), linestyle=cost_benefit_config.
80     get('x-axis-line-type'), label=cost_benefit_config.get('x-axis-line-label'))
81
82     # Add Labels
83     plt.xlabel(cost_benefit_config.get('x-axis-title'))
84     plt.ylabel(cost_benefit_config.get('y-axis-title'))
85     plt.title(cost_benefit_config.get('title'))
86
87     # Labling Data Points
88     for label, x_point, y_point in zip(labels, x, y):
89         plt.text(x_point, y_point, label)
90
91     # Show Legends
92     plt.legend()
93
94     # Show Grid
95     plt.grid(True)
96
97     # Save Diagram as PNG
98     plt.savefig(data_path)
99
100
101 cost_benefit_config = load_configuration()
102 cost_benefit_data = get_data(cost_benefit_config)
103 cost_benefit_diagram(cost_benefit_config, cost_benefit_data)

```

Listing 41: Python LaTex - cost\_benefit\_diagram.py - Kosten-Nutzen-Diagramm

## IX.VIII cost\_benefit\_diagram\_plotter\_conf.yaml

```

1 startpath: "parentdir"
2 destination_path: "source/cost_benefit_diagram"
3 datafile: "cost_benefit_diagram.csv"

```

## Diplomarbeit

```

4 imagename: "cost_benefit_diagram.png"
5 scatter-point-color: "blue"
6 x-axis-title: "Punkte"
7 x-axis-line-pos: 80
8 x-axis-line-label: "Kosten-Minimum"
9 x-axis-line-type: "--"
10 x-axis-line-color: "red"
11 y-axis-title: "Kosten"
12 y-axis-line-pos: 80
13 y-axis-line-label: "Punkte-Minimum"
14 y-axis-line-type: "--"
15 y-axis-line-color: "green"
16 title: "Kosten-Nutzen-Diagramm Beispiel"

```

Listing 42: Python LaTex - cost\_benefit\_diagram\_plotter\_conf.yaml - Konfigurationsdatei - Kosten-Nutzen-Diagramm

## IX.IX pandas\_dataframe\_to\_latex\_table.py

```

1 import os
2 import pandas as pd
3 import yaml
4 from pathlib import Path
5 import chardet
6
7 import csv
8
9 # Get the Configuration
10 def load_configuration(plt_conf_filename):
11     panda_latex_tables_config = dict()
12     plt_conf_dir = os.path.join(os.path.dirname(os.getcwd()), 'source', 'configuration')
13     yaml_path = os.path.join(plt_conf_dir, plt_conf_filename)
14
15     with open(yaml_path, "r") as file:
16         panda_latex_tables_config = yaml.load(file, Loader=yaml.FullLoader)
17
18     return panda_latex_tables_config
19
20
21 def get_data(startpath, destination, tablefilename, datafile_path, datafile,
22             alternative_csv_load, separator, decimal):
23     # Config Variables
24     if startpath == 'homedir':
25         directory = os.path.join(os.getcwd(), datafile_path)
26     else: # parentdir
        directory = os.path.join(os.path.dirname(os.getcwd()), datafile_path)

```

```
27
28     # get the Datas as direct
29     data_path = os.path.join(directory, datafile)
30
31     # load datas from csv into dict
32     detected = chardet.detect(Path(data_path).read_bytes())
33     encoding = detected.get("encoding")
34
35     # if alternative_csv_load:
36     #     with open(data_path, 'r', encoding=encoding) as file:
37     #         reader = csv.reader(file)
38     #         data = list(reader)
39     #
40     #         # panda_table_data = pd.DataFrame(data, columns=data[0])
41     #         # panda_table_data = pd.read_csv(data_path, sep=separator, decimal=
42     #         decimal, encoding=encoding, lineterminator='\n', engine='python')
43     #         panda_table_data = pd.read_csv(data_path, sep=separator, decimal=decimal
44     , encoding=encoding, lineterminator='\n')
45     #         df_dtype = {
46     #             "Nr.": int,
47     #             "Anforderung": str,
48     #             "Beschreibung": str,
49     #             "System": str,
50     #             "Muss / Kann": str
51     #         }
52     #         # panda_table_data = pd.read_csv(data_path, sep=",", decimal=".",
53     encoding=encoding, lineterminator='\n', dtype=df_dtype)
54     #         # panda_table_data = pd.read_csv(data_path, sep=",", decimal=".",
55     encoding=encoding)
56     # else:
57     #     panda_table_data = pd.read_csv(data_path, sep=separator, decimal=decimal
58     , encoding=encoding)
59     # panda_table_data = pd.read_csv(data_path, sep=",", decimal=".",
60     encoding=encoding, low_memory=False, engine='python')
61     # panda_table_data = pd.read_csv(data_path, sep=",", decimal=".",
62     encoding=encoding, engine='python', dtype='unicode')
63     # readed = open(data_path, 'r', encoding=encoding)
64     # panda_table_data = pd.read_csv(open(data_path, 'r', encoding=encoding), sep
65     =",", decimal=".",
66     encoding=encoding)
67     # panda_table_data = pd.read_csv(data_path, sep=",", decimal=".",
68     encoding = "ISO-8859-1")
69     # panda_table_data = pd.read_csv(data_path, sep=",", decimal=".",
70     encoding=encoding, chunksize=10)
71
72     # for chunk in pd.read_csv(data_path, sep=",", decimal=".",
73     encoding=encoding, chunksize=5):
74     #     print(chunk)
75     # panda_table_data = pd.DataFrame()
```

```

64     # temp = pd.read_csv(data_path, iterator=True, sep=",", decimal=".", encoding=
65     # encoding, chunksize=1000)
66
67     df_dtype = {
68         "Nr.": int,
69         "Anforderung": str,
70         "Beschreibung": str,
71         "System": str,
72         "Muss / Kann": str
73     }
74
75     # panda_table_data = pd.read_csv(data_path, sep=",", decimal= ".", encoding=
76     # encoding, engine='python', dtype=df_dtype)
77     # panda_table_data = pd.read_csv(data_path, sep=",", decimal= ".", encoding=
78     # encoding, dtype=df_dtype)
79
80     # import dask.dataframe as dd
81     # df = dd.read_csv(data_path, sep=",", decimal= ".", encoding=encoding)
82     # panda_table_data = df
83     print(encoding)
84
85     panda_table_data = pd.read_csv(data_path, sep=separator, decimal=decimal,
86     encoding=encoding)
87
88     # return data
89
90     return panda_table_data
91
92
93
94
95
96
97
98
99
100

```

**def create\_latex\_tables(panda\_latex\_tables\_config):**

**plt\_tables = panda\_latex\_tables\_config.get('tables\_inventory')**

**for table\_item in plt\_tables:**

**# id and filesystem informations**

**table\_id = panda\_latex\_tables\_config.get('tables').get(table\_item).get('id')**

**,**

**isbigfile = panda\_latex\_tables\_config.get('tables').get(table\_item).get('**

**isbigfile')**

**has\_longtexts = panda\_latex\_tables\_config.get('tables').get(table\_item).**

**get('has\_longtexts')**

**if isbigfile or has\_longtexts:**

**alternative\_cvs\_load = True**

**else:**

**alternative\_cvs\_load = False**

**startpath = panda\_latex\_tables\_config.get('tables').get(table\_item).get('**

**startpath')**

**destination = panda\_latex\_tables\_config.get('tables').get(table\_item).get('**

**destination\_path')**

**tablefilename = panda\_latex\_tables\_config.get('tables').get(table\_item).**

**get('tablefilename')**

**datafile\_path = panda\_latex\_tables\_config.get('tables').get(table\_item).**

**get('datafile\_path')**

```

101     datafile = panda_latex_tables_config.get('tables').get(table_item).get(
102         'datafile')
103         if startpath == 'homedir':
104             directory = os.path.join(os.getcwd(), destination)
105         else: # parentdir
106             directory = os.path.join(os.path.dirname(os.getcwd()), destination)
107         tablefile = os.path.join(directory, tablefilename)
108         separator = panda_latex_tables_config.get('tables').get(table_item).get(
109             'separator')
110         decimal = panda_latex_tables_config.get('tables').get(table_item).get(
111             'decimal')
112
113         # column operations
114         column_operations = panda_latex_tables_config.get('tables').get(table_item).
115             get('column_operations').get('datas')
116
117         # group by / aggregation
118         groupby_values = panda_latex_tables_config.get('tables').get(table_item).
119             get('group_by')
120         group_by_function = panda_latex_tables_config.get('tables').get(table_item).
121             get('group_by_function')
122         # selected_rows = panda_latex_tables_config.get('tables').get(table_item).
123             get('selected_rows')
124         agg_funtion = panda_latex_tables_config.get('tables').get(table_item).get(
125             'agg_funtion')
126         agg_columns = panda_latex_tables_config.get('tables').get(table_item).get(
127             'agg_columns')
128         # dropping and renaming columns
129         drop_columns = panda_latex_tables_config.get('tables').get(table_item).get(
130             'drop_columns')
131         rename_columns = panda_latex_tables_config.get('tables').get(table_item).
132             get('rename_columns')
133
134         # table filtering and sorting
135         where_clausel = panda_latex_tables_config.get('tables').get(table_item).
136             get('where_clausel')
137         order_by = panda_latex_tables_config.get('tables').get(table_item).get(
138             'sorting').get('order_by')
139         sort_acending = panda_latex_tables_config.get('tables').get(table_item).
140             get('sorting').get('sort_acending')
141         sort_inplace = panda_latex_tables_config.get('tables').get(table_item).get(
142             'sorting').get('sort_inplace')
143
144         # pivot settings
145         pivot = panda_latex_tables_config.get('tables').get(table_item).get('pivot
146             ')
147         pivot_column = panda_latex_tables_config.get('tables').get(table_item).get(
148             'pivot_columns')

```

```
132     pivot_value = panda_latex_tables_config.get('tables').get(table_item).get(
133         'pivot_values')
134
134     # pivot_table settings
135     pivot_table = panda_latex_tables_config.get('tables').get(table_item).get(
136         'pivot_table')
137     pivot_table_column = panda_latex_tables_config.get('tables').get(
138         table_item).get('pivot_table').get(
139             'pivot_columns')
140     pivot_table_value = panda_latex_tables_config.get('tables').get(table_item).
141         get('pivot_table').get(
142             'pivot_values')
143     pivot_table_agg_function = panda_latex_tables_config.get('tables').get(
144         table_item).get('pivot_table').get(
145             'pivot_agg_func')
146     pivot_table_indizes = panda_latex_tables_config.get('tables').get(
147         table_item).get('pivot_table').get(
148             'pivot_index').get('pivot_indizes')
149     pivot_table_indizes_visible = panda_latex_tables_config.get('tables').get(
150         table_item).get('pivot_table').get(
151             'pivot_index').get('pivot_indizes_visible')
152     pivot_table_rename_indizes = panda_latex_tables_config.get('tables').get(
153         table_item).get('pivot_table').get(
154             'pivot_index').get('pivot_rename_indizes')
155
156     # margins (subtotals)
157     margin = panda_latex_tables_config.get('tables').get(table_item).get(
158         'margins').get('margin')
159     margin_name = panda_latex_tables_config.get('tables').get(table_item).get(
160         'margins').get('margin_name')
161
161     # table settings
162     table_caption = panda_latex_tables_config.get('tables').get(table_item).
163         get('caption')
164     table_label = panda_latex_tables_config.get('tables').get(table_item).get(
165         'label')
166     table_style = panda_latex_tables_config.get('tables').get(table_item).get(
167         'table_styles')
168     sparse_columns = panda_latex_tables_config.get('tables').get(table_item).
169         get('table_styles').get(
170             'sparse_columns')
171     table_caption_position = panda_latex_tables_config.get('tables').get(
172         table_item).get('table_styles').get(
173             'props').get('caption-side')
174     table_position = panda_latex_tables_config.get('tables').get(table_item).
175         get('table_styles').get('props').get(
176             'position')
177     longtable = panda_latex_tables_config.get('tables').get(table_item).get(
```

```
164     table_styles').get('props').get(
165         'longtable')
166     linebreak_columns = panda_latex_tables_config.get('tables').get(table_item)
167     .get('table_styles').get('props').get(
168         'linebreak_columns')
169     resize_textwidth = panda_latex_tables_config.get('tables').get(table_item)
170     .get('table_styles').get('props').get(
171         'resize_textwidth')
172
173     # get the pandas (panda data)
174     panda_table_data = get_data(startpath, destination, tablefilename,
175     datafile_path, datafile, alternative_csv_load, separator, decimal)
176
177     # filter by where clause
178     if where_clause:
179         panda_table_data = panda_table_data.query(where_clause)
180
181     # Drop unused columns
182     if drop_columns:
183         panda_table_data = panda_table_data.drop(columns=drop_columns)
184
185     # set aggregation functions
186     # if groupby_values and not agg_funtion and not pivot_column and not
187     # pivot_table_column:
188     if groupby_values and not (pivot_column or (pivot_table_column or
189     pivot_table_value or pivot_table_index)):
190         match group_by_function:
191             case 'max':
192                 panda_table_data = panda_table_data.groupby(groupby_values,
193                 as_index=False).max()
194             case 'min':
195                 panda_table_data = panda_table_data.groupby(groupby_values,
196                 as_index=False).min()
197             case 'head':
198                 panda_table_data = panda_table_data.groupby(groupby_values,
199                 as_index=False).head()
200             case 'sum':
201                 panda_table_data = panda_table_data.groupby(groupby_values,
202                 as_index=False).sum()
203             case 'mean':
204                 panda_table_data = panda_table_data.groupby(groupby_values,
205                 as_index=False).mean()
206             else:
207                 panda_table_data = panda_table_data
208
209     # pivot if pivot is selected
210     if pivot_table_column or pivot_table_value or pivot_table_index:
211         if type(pivot_table_agg_function) is list:
```

```

201         agg_tuple = tuple(pivot_table_agg_function)
202         panda_table_data = pd.pivot_table(panda_table_data, index=
203                                         pivot_table_index,
204                                         columns=pivot_table_column,
205                                         values=pivot_table_value,
206                                         aggfunc=agg_tuple, margins=
207                                         margin, margins_name=margin_name)
208
209         elif type(pivot_table_agg_function) is dict:
210             panda_table_data = pd.pivot_table(panda_table_data, index=
211                                         pivot_table_index,
212                                         columns=pivot_table_column,
213                                         values=pivot_table_value,
214                                         aggfunc=pivot_table_agg_function,
215                                         margins=margin, margins_name=
216                                         margin_name)
217
218         else:
219             panda_table_data = pd.pivot_table(panda_table_data, index=
220                                         pivot_table_index,
221                                         columns=pivot_table_column,
222                                         values=pivot_table_value,
223                                         aggfunc=pivot_table_agg_function
224                                         , margins=margin,
225                                         margins_name=margin_name)
226
227
228         # set column operations
229         if column_operations:
230             for column_ops in column_operations:
231                 operation_function = panda_latex_tables_config.get('tables').get(
232                     table_item).get('column_operations').get('operations').get(column_ops).get(
233                         'operation_function')
234
235                 operation_columns = panda_latex_tables_config.get('tables').get(
236                     table_item).get('column_operations').get('operations').get(column_ops).get(
237                         'columns')
238
239                 operation_axis = panda_latex_tables_config.get('tables').get(
240                     table_item).get('column_operations').get('operations').get(column_ops).get(
241                         'axis_number')
242
243                 match operation_function:
244                     case 'max':
245                         panda_table_data[column_ops] = panda_table_data[
246                             operation_columns].max()
247                     case 'min':
248                         panda_table_data[column_ops] = panda_table_data[
249                             operation_columns].min()
250                     case 'head':
251                         panda_table_data[column_ops] = panda_table_data[
252                             operation_columns].head()
253                     case 'sum':
254                         panda_table_data[column_ops] = panda_table_data[

```

```
operation_columns].sum(axis=operation_axis)
231             case 'mean':
232                 panda_table_data[column_ops] = panda_table_data[
233                     operation_columns].mean()
234             case 'diff':
235                 panda_table_data[column_ops] = panda_table_data[
236                     operation_columns[1]] - panda_table_data[operation_columns[0]]
237
238             # order by
239             if order_by:
240                 panda_table_data.sort_values(by=order_by, inplace=sort_inplace,
241                     ascending=sort_acending)
242
243             # rename columns
244             if rename_columns:
245                 panda_table_data = panda_table_data.rename(columns=rename_columns)
246
247             # rename indices
248             if pivot_table_rename_indizes:
249                 panda_table_data = panda_table_data.rename_axis(index=
250                     pivot_table_rename_indizes)
251
252             # frame carriage return columns in subtable
253             if linebreak_columns:
254                 for lbr_column in linebreak_columns:
255                     panda_table_data[lbr_column] = "\begin{tabular}[c]{@{}l@{}}" +
256                     panda_table_data[lbr_column].astype(str) + "\end{tabular}"
257
258             # convert python panda to latex table
259             latex_table = panda_table_data.to_latex(header=True, bold_rows=False,
260                     longtable=longtable,
261                                         sparsify=sparse_columns, label=
262                     table_label, caption=table_caption,
263                                         position=table_position, na_rep='',
264                     , index=pivot_table_indizes_visible)
265
266             # textwidth resize
267             if resize_textwidth:
268                 with open(tablefile, 'w') as wrlt:
269                     wrlt.write(latex_table)
270
271
272                 with open(tablefile) as file:
273                     lines = file.readlines()
274
275
276             # replace table with resize
277             resize_line_nr = 0
278             resize_line = ""
279             if longtable:
```

```

270         table_type = '\\begin{longtable}'  
271     else:  
272         table_type = '\\begin{table}'  
273  
274     for number, line in enumerate(lines, 1):  
275         # for number, line in latex_table.splitlines():  
276         # for number, line in latex_table.readlines():  
277         # for number, line in latex_table.splitlines('\\n'):   
278         # for number, line in lines.split('\\n'):   
279  
280             # Condition true if the key exists in the line  
281             # If true then display the line number  
282             if table_type in line:  
283                 # print(f'{key} is at line {number}')  
284                 resize_line_nr = number  
285                 resize_line = line  
286  
287             line_table_resize = resize_line + "\\n" + "\\resizebox{\\columnwidth  
288 }{!}{%"  
289             latex_table = latex_table.replace(resize_line, line_table_resize)  
290  
291             # replace table end with bracket  
292             resize_line_nr = 0  
293             resize_line = ""  
294             if longtable:  
295                 table_type = '\\end{longtable}'  
296             else:  
297                 table_type = '\\end{table}'  
298  
299             for number, line in enumerate(lines, 1):  
300  
301                 # Condition true if the key exists in the line  
302                 # If true then display the line number  
303                 if table_type in line:  
304                     # print(f'{key} is at line {number}')  
305                     resize_line_nr = number  
306                     resize_line = line  
307  
308             line_table_resize = "}" + "\\n" + resize_line  
309             latex_table = latex_table.replace(resize_line, line_table_resize)  
310  
311             # caption below is not supported yet (pandas 2.2)  
312             # replace caption and replace table end with the caption line and table  
313             end  
314             if table_caption_position == 'below':  
315                 caption_label = "\\caption{" + table_caption + "}" "\\label{" +  
316                 table_label + "}" "\\\\"  
317                 caption_label_nbr = "\\caption{" + table_caption + "}" "\\label{" +

```

```

315     table_label + "}"
316     caption_only = "\\caption{" + table_caption + "}" \\
317     caption_only_nbr = "\\caption{" + table_caption + "}"
318     label_only = "\\label{" + table_label + "}" \\
319     label_only_nbr = "\\label{" + table_label + "}"
320     latex_table = latex_table.replace(caption_label, '')
321     latex_table = latex_table.replace(caption_only, '')
322     latex_table = latex_table.replace(label_only, '')
323     latex_table = latex_table.replace(caption_label_nbr, '')
324     latex_table = latex_table.replace(caption_only_nbr, '')
325     latex_table = latex_table.replace(label_only_nbr, '')

326     if longtable:
327         table_string = '\\end{longtable}'
328         new_caption = caption_label_nbr + "\n" + table_string
329         latex_table = latex_table.replace(table_string, new_caption)
330     else:
331         table_string = '\\end{table}'
332         new_caption = caption_label_nbr + "\n" + table_string
333         latex_table = latex_table.replace(table_string, new_caption)

334

335     # write latex table to filesystem
336     with open(tablefile, 'w') as wrlt:
337         wrlt.write(latex_table)

338

339

340

341 # run the methods / functions
342 panda_latex_tables_config = load_configuration('csv_to_latex_diplomarbeit.yaml')
343 create_latex_tables(panda_latex_tables_config)

```

Listing 43: Python LaTex - pandas\_dataframe\_to\_latex\_table.py CSV - LaTex Tabelle

## IX.X csv\_to\_latex\_diplomarbeit.yaml

```

1 tables_inventory:
2   - "db_inventory"
3   - "db_inventory_per_rdbms"
4   - "db_inventory_per_os"
5   - "anforderungskatalog"
6   - "arbeitsrapport"
7   - "projektcontrolling"
8   - "evaluation_inventory"
9   - "dependencis"
10  - "predecision_out"
11  - "predecision_in"
12  - "project_comments"

```

## Diplomarbeit

```
13 - "evaluation_distributed_sql"
14 - "expert_discussions_overview"
15 - "expert_discussions_full_list"
16 - "stakeholder"
17 tables:
18 db_inventory:
19   id: "db_inventory"
20   isbigfile:
21   has_longtexts: False
22   separator: ","
23   decimal: "."
24   caption: "Datenbankinventar - Roh"
25   label: "db_inventory"
26   startpath: "parentdir"
27   destination_path: "content/latex_tables"
28   datafile_path: "source/tables"
29   datafile: "inventory.csv"
30   tablefilename: "db_inventory.tex"
31   decimal_format:
32   group_by:
33   group_by_function:
34   agg_funtion:
35   agg_columns:
36   drop_columns:
37   - "comment"
38   - "eol"
39   - "eol_since"
40   - "releasedate"
41 column_operations:
42   datas:
43     operations:
44       dauer_summe:
45         operation_function:
46         axis_number:
47         columns:
48 pivot:
49   pivot_columns:
50   pivot_values:
51 pivot_table:
52   pivot_index:
53   pivot_indizes_visible:
54   pivot_rename_indizes:
55 pivot_columns:
56 pivot_values:
57 pivot_agg_func:
58 rename_columns:
59   server: "Server - Hostname"
60   os: "OS"
```

## Diplomarbeit

```
61     rdbms: "RDBMS"
62     instance: "Instanz"
63     databases: "Datenbanken"
64     appliance: "Appliance"
65     comment: "Kommentar"
66     version: "Version"
67     releasedate: "Version - Releasedatum"
68     eol: "EoL"
69     age: "Version - Alter"
70     eol_since: "EoL seit"
71     where_clause:
72     sorting:
73         order_by:
74             - "server"
75             - "rdbms"
76         sort_acending: True
77         sort_inplace: True
78     margins:
79         margin: False
80         margin_name:
81     table_styles:
82         selector: "caption"
83         props:
84             caption-side: "below"
85             position: "H"
86             sparse_columns: True
87             longtable: True
88             resize_textwidth: False
89             linebreak_columns:
90             table_header: True
91     db_inventory_per_rdbms:
92         id: "db_inventory_per_rdbms"
93         isbigfile:
94         has_longtexts: False
95         separator: ","
96         decimal: "."
97         caption: "Datenbankinventar"
98         label: "db_inventory_per_rdbms"
99         startpath: "parentdir"
100        destination_path: "content/latex_tables"
101        datafile_path: "source/tables"
102        datafile: "inventory.csv"
103        tablefilename: "db_inventory_per_rdbms.tex"
104        decimal_format:
105        group_by:
106            - "rdbms"
107        group_by_function: "sum"
108        agg_funtion:
```

## Diplomarbeit

```
109 agg_columns:
110   - "rdbms"
111 drop_columns:
112   - "server"
113   - "os"
114   - "version"
115   - "releasedate"
116   - "eol"
117   - "age"
118   - "eol_since"
119   - "comment"
120 column_operations:
121   datas:
122     operations:
123       dauer_summe:
124         operation_function:
125         axis_number:
126         columns:
127   pivot:
128     pivot_columns:
129     pivot_values:
130   pivot_table:
131     pivot_index:
132     pivot_indizes_visible:
133     pivot_rename_indizes:
134   pivot_columns:
135   pivot_values:
136   pivot_agg_func:
137 rename_columns:
138   rdbms: "RDBMS"
139   instance : "Instanz"
140   databases : "Datenbanken"
141   appliance: "Appliance"
142 where_clausel:
143 sorting:
144   order_by:
145     - "rdbms"
146   sort_acending: True
147   sort_inplace: True
148 margins:
149   margin: True
150   margin_name: "Gesamtergebnis"
151 table_styles:
152   selector: "caption"
153   props:
154     caption-side: "below"
155     position: "H"
156     sparse_columns: True
```

## Diplomarbeit

```
157     longtable: False
158     resize_textwidth: False
159     linebreak_columns:
160     table_header: True
161 db_inventory_per_os:
162   id: "db_inventory_per_os"
163   isbigfile:
164   has_longtexts: False
165   separator: ","
166   decimal: "."
167   caption: "Datenbankinventor - Nach Betriebssystemen üaufgeschlisselt"
168   label: "db_inventory_per_os"
169   startpath: "parentdir"
170   destination_path: "content/latex_tables"
171   datafile_path: "source/tables"
172   datafile: "inventory.csv"
173   tablefilename: "db_inventory_per_os.tex"
174   decimal_format:
175   group_by:
176     - "rdbms"
177     - "os"
178   group_by_function: "sum"
179   agg_funtion:
180   agg_columns:
181     - "os"
182   drop_columns:
183     - "server"
184     - "version"
185     - "releasedate"
186     - "eol"
187     - "age"
188     - "eol_since"
189     - "comment"
190 #     - "appliance"
191   column_operations:
192     datas:
193       operations:
194         dauer_summe:
195           operation_function:
196           axis_number:
197           columns:
198         pivot:
199           pivot_columns:
200           pivot_values:
201           pivot_table:
202             pivot_index:
203             pivot_indizes:
204               - "os"
```

## Diplomarbeit

```
205     - "rdbms"
206     pivot_indexes_visible: True
207     pivot_rename_indexes:
208       os: "OS"
209       rdbms: "RDBMS"
210     pivot_columns:
211     pivot_values:
212     pivot_agg_func:
213       instance: "sum"
214       databases: "sum"
215       appliance: "sum"
216     transpose: True
217     rename_columns:
218       rdbms: "RDBMS"
219       instance : "Instanz"
220       databases : "Datenbanken"
221       os : "OS"
222       appliance: "Appliance"
223     where_clauses:
224     sorting:
225       order_by:
226         sort_ascending: False
227         sort_inplace: True
228     margins:
229       margin: True
230       margin_name: "Gesamtergebnis"
231     table_styles:
232       selector: "caption"
233     props:
234       caption-side: "below"
235       position: "H"
236     sparse_columns: True
237     longtable: True
238     resize_textwidth: False
239     linebreak_columns:
240     table_header: True
241   anforderungskatalog:
242     id: "anforderungskatalog"
243     isbigfile:
244     has_longtexts: True
245     separator: ";"
246     decimal: "."
247     caption: "Anforderungskatalog"
248     label: "anforderungskatalog"
249     startpath: "parentdir"
250     destination_path: "content/latex_tables"
251     datafile_path: "source/tables"
252     datafile: "anforderungskatalog.CSV"
```

## Diplomarbeit

```
253     tablefilename: "anforderungskatalog.tex"
254     decimal_format:
255     group_by:
256     group_by_function:
257     agg_funtion:
258     agg_columns:
259     drop_columns:
260     column_operations:
261     datas:
262     operations:
263     dauer_summe:
264     operation_function:
265     axis_number:
266     columns:
267     pivot:
268     pivot_columns:
269     pivot_values:
270     pivot_table:
271     pivot_index:
272     pivot_indizes_visible: False
273     pivot_rename_indizes:
274     pivot_columns:
275     pivot_values:
276     pivot_agg_func:
277     rename_columns:
278     where_clausel:
279     sorting:
280     order_by:
281     - "Nr."
282     sort_acending: True
283     sort_inplace: True
284     margins:
285     margin: False
286     margin_name:
287     table_styles:
288     selector: "caption"
289     props:
290     caption-side: "below"
291     position: "H"
292     sparse_columns: False
293     longtable: False
294     resize_textwidth: True
295     linebreak_columns:
296     - "Beschreibung"
297     table_header: True
298     arbeitsrapport:
299     id: "arbeitsrapport"
300     isbigfile:
```

## Diplomarbeit

```
301 has_longtexts: False
302 separator: ";"
303 decimal: "."
304 caption: "Arbeitsrapport"
305 label: "arbeitsrapport"
306 startpath: "parentdir"
307 destination_path: "content/latex_tables"
308 datafile_path: "source/tables"
309 datafile: "arbeitsrapport.CSV"
310 tablefilename: "arbeitsrapport.tex"
311 decimal_format: "{:0.1f}"
312 group_by:
313 group_by_function:
314 agg_funtion:
315 agg_columns:
316 drop_columns:
317 - "Hide"
318 - "Geplante Dauer [h]"
319 - "dauer_summe"
320 column_operations:
321 datas:
322 operations:
323 dauer_summe:
324 operation_function:
325 axis_number:
326 columns:
327 pivot:
328 pivot_columns:
329 pivot_values:
330 pivot_table:
331 pivot_index:
332 pivot_indexes_visible: False
333 pivot_rename_indexes:
334 pivot_columns:
335 pivot_values:
336 pivot_agg_func:
337 rename_columns:
338 where_clause1: "Hide == 0"
339 sorting:
340 order_by:
341 - "Datum"
342 - "Von"
343 sort_acending: False
344 sort_inplace: False
345 margins:
346 margin: False
347 margin_name:
348 table_styles:
```

## Diplomarbeit

```
349 selector: "caption"
350 props:
351   caption-side: "below"
352   position: "H"
353   sparse_columns: True
354   longtable: False
355   resize_textwidth: True
356   linebreak_columns:
357     - "ÄTtigkeit"
358     - "Bemerkung"
359     - "Schwierigkeit"
360     - "öLsungen"
361   table_header: True
362 projektcontrolling:
363   id: "projektcontrolling"
364   isbigfile:
365   has_longtexts: False
366   separator: ";"
367   decimal: "."
368   caption: "Projektcontrolling"
369   label: "projektcontrolling"
370   startpath: "parentdir"
371   destination_path: "content/latex_tables"
372   datafile_path: "source/tables"
373   datafile: "arbeitsrapport.CSV"
374   tablefilename: "projektcontrolling.tex"
375   decimal_format: "{:0.1f}"
376   group_by:
377     - "Phase"
378     - "Subphase"
379   group_by_function: "sum"
380   agg_funtion:
381   agg_columns:
382     - "Dauer [h]"
383     - "Geplante Dauer [h]"
384     - "dauer_summe"
385   drop_columns:
386     - "Datum"
387     - "Von"
388     - "Bis"
389     - "Hide"
390     - "ÄTtigkeit"
391     - "Bemerkung"
392     - "Schwierigkeit"
393     - "öLsungen"
394   column_operations:
395     datas:
396       - "dauer_summe"
```

## Diplomarbeit

```
397 operations:
398     dauer_summe:
399         operation_function: "diff"
400         axis_number: 1
401         columns:
402             - "Dauer [h]"
403             - "Geplante Dauer [h]"
404 pivot:
405     pivot_columns:
406     pivot_values:
407 pivot_table:
408     pivot_index:
409     pivot_indexes_visible:
410     pivot_rename_indexes:
411     pivot_columns:
412     pivot_values:
413     pivot_agg_func:
414 rename_columns:
415     dauer_summe: "Verbleibende Zeit [h]"
416 where_clausel:
417 sorting:
418     order_by:
419         - "Phase"
420         - "Subphase"
421     sort_acending: True
422     sort_inplace: True
423 margins:
424     margin: True
425     margin_name: "Total"
426 table_styles:
427     selector: "caption"
428     props:
429         caption-side: "below"
430         position: "H"
431         sparse_columns: True
432         longtable: False
433         resize_textwidth: True
434         linebreak_columns:
435         table_header: True
436 evaluation_inventory:
437     id: "evaluation_inventory"
438     isbigfile:
439     has_longtexts: False
440     separator: ";"
441     decimal: "."
442     caption: "Evaluationssysteme"
443     label: "evaluation_inventory"
444     startpath: "parentdir"
```

## Diplomarbeit

```
445 destination_path: "content/latex_tables"
446 datafile_path: "source/tables"
447 datafile: "evaluation_platform_serverlist.csv"
448 tablefilename: "evaluation_inventory.tex"
449 decimal_format:
450 group_by:
451 group_by_function:
452 agg_funtion:
453 agg_colums:
454 drop_columns:
455 column_operations:
456     datas:
457     operations:
458         dauer_summe:
459             operation_function:
460                 axis_number:
461                 columns:
462 pivot:
463     pivot_columns:
464     pivot_values:
465 pivot_table:
466     pivot_index:
467         pivot_indizes_visible: False
468         pivot_rename_indizes:
469 pivot_columns:
470 pivot_values:
471 pivot_agg_func:
472 rename_columns:
473 where_clause:
474 sorting:
475     order_by:
476         - "Server"
477         - "Typ"
478 sort_acending: True
479 sort_inplace: True
480 margins:
481     margin: False
482     margin_name:
483 table_styles:
484     selector: "caption"
485 props:
486     caption-side: "below"
487     position: "H"
488 sparse_columns: True
489 longtable: True
490 resize_textwidth: False
491 linebreak_columns:
492     table_header: True
```

## Diplomarbeit

```
493 dependencis:  
494     id: "dependencis"  
495     isbigfile:  
496     has_longtexts: False  
497     separator: ";"  
498     decimal: "."  
499     caption: "ÄAbhngigkeiten"  
500     label: "dependencis"  
501     startpath: "parentdir"  
502     destination_path: "content/latex_tables"  
503     datafile_path: "source/tables"  
504     datafile: "dependencis.csv"  
505     tablefilename: "dependencis.tex"  
506     decimal_format:  
507     group_by:  
508     group_by_function:  
509     agg_funtion:  
510     agg_columns:  
511     drop_columns:  
512     column_operations:  
513     datas:  
514     operations:  
515         dauer_summe:  
516             operation_function:  
517             axis_number:  
518             columns:  
519         pivot:  
520             pivot_columns:  
521             pivot_values:  
522             pivot_table:  
523             pivot_index:  
524             pivot_indizes_visible: False  
525             pivot_rename_indizes:  
526             pivot_columns:  
527             pivot_values:  
528             pivot_agg_func:  
529             rename_columns:  
530             where_clausel:  
531             sorting:  
532                 order_by:  
533                     - "Nr."  
534                 sort_acending: True  
535                 sort_inplace: True  
536             margins:  
537                 margin: False  
538                 margin_name:  
539             table_styles:  
540                 selector: "caption"
```

## Diplomarbeit

```
541 props:
542   caption_side: "below"
543   position: "H"
544   sparse_columns: True
545   longtable: False
546   resize_textwidth: True
547   linebreak_columns:
548     - "äAbhngigkeit"
549     - "Beschreibung"
550     - "Status"
551     - "Risiko"
552     - "Impact"
553   table_header: True
554 predecision_out:
555   id: "predecision_out"
556   isbigfile:
557   has_longtexts: False
558   separator: ";"
559   decimal: "."
560   caption: "Vorauswahl - Ausgeschieden"
561   label: "predecision_out"
562   startpath: "parentdir"
563   destination_path: "content/latex_tables"
564   datafile_path: "source/tables"
565   datafile: "pre-decision.csv"
566   tablefilename: "pre-decision-out.tex"
567   decimal_format:
568   group_by:
569   group_by_function:
570   agg_funtion:
571   agg_columns:
572   drop_columns:
573     - "hide_state"
574   column_operations:
575     datas:
576     operations:
577       dauer_summe:
578         operation_function:
579         axis_number:
580         columns:
581   pivot:
582     pivot_columns:
583     pivot_values:
584   pivot_table:
585     pivot_index:
586     pivot_indexes_visible: False
587     pivot_rename_indexes:
588     pivot_columns:
```

## Diplomarbeit

```
589     pivot_values:
590     pivot_agg_func:
591     rename_columns:
592     where_clause: "hide_state == 1"
593     sorting:
594         order_by:
595             - "Nr."
596         sort_acending: True
597         sort_inplace: True
598     margins:
599         margin: False
600         margin_name:
601     table_styles:
602         selector: "caption"
603     props:
604         caption-side: "below"
605         position: "H"
606         sparse_columns: True
607         longtable: False
608         resize_textwidth: True
609         linebreak_columns:
610             - "üBegrndung"
611         table_header: True
612     predecision_in:
613         id: "predecision_in"
614         isbigfile:
615         has_longtexts: False
616         separator: ";"
617         decimal: "."
618         caption: "Vorauswahl - Evaluation"
619         label: "predecision_in"
620         startpath: "parentdir"
621         destination_path: "content/latex_tables"
622         datafile_path: "source/tables"
623         datafile: "pre-decision.csv"
624         tablefilename: "pre-decision-in.tex"
625         decimal_format:
626         group_by:
627         group_by_function:
628         agg_funtion:
629         agg_columns:
630         drop_columns:
631             - "hide_state"
632         column_operations:
633             datas:
634             operations:
635                 dauer_summe:
636                 operation_function:
```

## Diplomarbeit

```
637     axis_number:
638     columns:
639     pivot:
640         pivot_columns:
641         pivot_values:
642     pivot_table:
643         pivot_index:
644             pivot_indexes_visible: False
645             pivot_rename_indexes:
646         pivot_columns:
647         pivot_values:
648         pivot_agg_func:
649     rename_columns:
650     where_clause: "hide_state == 2"
651     sorting:
652         order_by:
653             - "Nr."
654         sort_ascending: True
655         sort_inplace: True
656     margins:
657         margin: False
658         margin_name:
659     table_styles:
660         selector: "caption"
661     props:
662         caption_side: "below"
663         position: "H"
664         sparse_columns: True
665         longtable: False
666         resize_textwidth: True
667         linebreak_columns:
668             - "ÜBegrndung"
669         table_header: True
670     project_comments:
671         id: "project_comments"
672         isbigfile:
673         has_longtexts: False
674         separator: ";"
675         decimal: "."
676         caption: "Kommentare - Anmerkung"
677         label: "project_comments"
678         startpath: "parentdir"
679         destination_path: "content/latex_tables"
680         datafile_path: "source/tables"
681         datafile: "pre-fazit.csv"
682         tablefilename: "pre-fazit.tex"
683         decimal_format:
684         group_by:
```

## Diplomarbeit

```
685 group_by_function:  
686 agg_funtion:  
687 agg_columns:  
688 drop_columns:  
689 column_operations:  
690 datas:  
691 operations:  
692 dauer_summe:  
693 operation_function:  
694 axis_number:  
695 columns:  
696 pivot:  
697 pivot_columns:  
698 pivot_values:  
699 pivot_table:  
700 pivot_index:  
701 pivot_indizes_visible: False  
702 pivot_rename_indizes:  
703 pivot_columns:  
704 pivot_values:  
705 pivot_agg_func:  
706 rename_columns:  
707 where_clausel:  
708 sorting:  
709 order_by:  
710 - "Woche"  
711 sort_acending: True  
712 sort_inplace: True  
713 margins:  
714 margin: False  
715 margin_name:  
716 table_styles:  
717 selector: "caption"  
718 props:  
719 caption-side: "below"  
720 position: "H"  
721 sparse_columns: True  
722 longtable: False  
723 resize_textwidth: True  
724 linebreak_columns:  
725 - "Beschreibung / Event / Problem"  
726 table_header: True  
727 evaluation_distributed_sql:  
728 id: "evaluation_distributed_sql"  
729 isbigfile:  
730 has_longtexts: False  
731 separator: ";"  
732 decimal: "."
```

## Diplomarbeit

```
733 caption: "Evaluationssystem - Distributed SQL / Sharding"
734 label: "evaluation_distributed_sql"
735 startpath: "parentdir"
736 destination_path: "content/latex_tables"
737 datafile_path: "source/tables"
738 datafile: "evaluation_platform_distributed_sql.csv"
739 tablefilename: "evaluation_platform_distributed_sql.tex"
740 decimal_format:
741 group_by:
742 group_by_function:
743 agg_funtion:
744 agg_columns:
745 drop_columns:
746 column_operations:
747 datas:
748 operations:
749 dauer_summe:
750     operation_function:
751     axis_number:
752     columns:
753 pivot:
754     pivot_columns:
755     pivot_values:
756 pivot_table:
757     pivot_index:
758     pivot_indizes_visible: False
759     pivot_rename_indizes:
760     pivot_columns:
761     pivot_values:
762     pivot_agg_func:
763 rename_columns:
764 where_clausel:
765 sorting:
766     order_by:
767     sort_acending: False
768     sort_inplace: False
769 margins:
770     margin: False
771     margin_name:
772 table_styles:
773     selector: "caption"
774     props:
775         caption-side: "below"
776         position: "H"
777         sparse_columns: True
778         longtable: False
779         resize_textwidth: False
780         linebreak_columns:
```

## Diplomarbeit

```
781     table_header: False
782 expert_discussions_overview:
783     id: "expert_discussions_overview"
784     isbigfile:
785     has_longtexts: False
786     separator: ";"
787     decimal: "."
788     caption: "äFachgespräche"
789     label: "expert_discussions_overview"
790     startpath: "parentdir"
791     destination_path: "content/latex_tables"
792     datafile_path: "source/tables"
793     datafile: "expert_discussions.csv"
794     tablefilename: "expert_discussions_overview.tex"
795     decimal_format:
796     group_by:
797     group_by_function:
798     agg_funtion:
799     agg_columns:
800     drop_columns:
801         - "Fragen"
802         - "Antworten"
803         - "Sonstige Themen"
804     column_operations:
805     datas:
806     operations:
807         dauer_summe:
808             operation_function:
809             axis_number:
810             columns:
811         pivot:
812             pivot_columns:
813             pivot_values:
814             pivot_table:
815             pivot_index:
816                 pivot_indizes_visible: False
817                 pivot_rename_indizes:
818             pivot_columns:
819             pivot_values:
820             pivot_agg_func:
821             rename_columns:
822             where_clausel:
823             sorting:
824             order_by:
825                 - "äFachgespräch"
826             sort_acending: True
827             sort_inplace: True
828             margins:
```

## Diplomarbeit

```
829     margin: False
830     margin_name:
831     table_styles:
832         selector: "caption"
833         props:
834             caption-side: "below"
835             position: "H"
836             sparse_columns: True
837             longtable: False
838             resize_textwidth: True
839             linebreak_columns:
840                 - "Studenten"
841                 - "Bemerkungen"
842             table_header: True
843     expert_discussions_full_list:
844         id: "expert_discussions_full_list"
845         isbigfile:
846         has_longtexts: False
847         separator: ";"
848         decimal: "."
849         caption: "äFachgespräche - Protokoll"
850         label: "expert_discussions_full_list"
851         startpath: "parentdir"
852         destination_path: "content/latex_tables"
853         datafile_path: "source/tables"
854         datafile: "expert_discussions.csv"
855         tablefilename: "expert_discussions_full_list.tex"
856         decimal_format:
857         group_by:
858         group_by_function:
859         agg_funtion:
860         agg_colums:
861         drop_columns:
862         column_operations:
863         datas:
864         operations:
865             dauer_summe:
866             operation_function:
867             axis_number:
868             columns:
869         pivot:
870             pivot_columns:
871             pivot_values:
872         pivot_table:
873             pivot_index:
874                 pivot_indizes_visible: False
875                 pivot_rename_indizes:
876                 pivot_columns:
```

## Diplomarbeit

```
877     pivot_values:  
878     pivot_agg_func:  
879     rename_columns:  
880     where_clause:  
881     sorting:  
882         order_by:  
883             - "äFachgespräch"  
884         sort_acending: True  
885         sort_inplace: True  
886     margins:  
887         margin: False  
888         margin_name:  
889     table_styles:  
890         selector: "caption"  
891     props:  
892         caption-side: "below"  
893         position: "H"  
894         sparse_columns: True  
895         longtable: False  
896         resize_textwidth: True  
897         linebreak_columns:  
898             - "Studenten"  
899             - "Fragen"  
900             - "Antworten"  
901             - "Sonstige Themen"  
902             - "Bemerkungen"  
903         table_header: True  
904     stakeholder:  
905         id: "stakeholder"  
906         isbigfile:  
907         has_longtexts: False  
908         separator: ";"  
909         decimal: "."  
910         caption: "Stakeholder"  
911         label: "stakeholder"  
912         startpath: "parentdir"  
913         destination_path: "content/latex_tables"  
914         datafile_path: "source/tables"  
915         datafile: "stakeholder.csv"  
916         tablefilename: "stakeholder.tex"  
917         decimal_format:  
918         group_by:  
919         group_by_function:  
920         agg_funtion:  
921         agg_columns:  
922         drop_columns:  
923         column_operations:  
924             datas:
```

## Diplomarbeit

```

925     operations:
926     dauer_summe:
927     operation_function:
928     axis_number:
929     columns:
930
931     pivot:
932     pivot_columns:
933     pivot_values:
934     pivot_table:
935     pivot_index:
936     pivot_indexes_visible: False
937     pivot_rename_indexes:
938     pivot_columns:
939     pivot_values:
940     pivot_agg_func:
941     rename_columns:
942     where_clause:
943     sorting:
944     order_by:
945     sort_ascending: True
946     sort_inplace: True
947     margins:
948     margin: False
949     margin_name:
950     table_styles:
951     selector: "caption"
952     props:
953     caption_side: "below"
954     position: "H"
955     sparse_columns: True
956     longtable: False
957     resize_textwidth: True
958     linebreak_columns:
959     table_header: True

```

Listing 44: Python LaTex - csv\_to\_latex\_diplomarbeit.yaml - Konfigurationsdatei - CSV - LaTex-Tabelle

## IX.XI pandas\_data\_chart\_plotter.py

```

1 import os
2 from pathlib import Path
3 import chardet
4 import pandas as pd
5 import yaml
6
7

```

```

8 def load_configuration(panda_diagram_plotter_conf_filename):
9     panda_diagram_plotter_config = dict()
10
11     riskmatrix_conf_dir = os.path.join(os.path.dirname(os.getcwd()), 'source', 'configuration')
12     yaml_path = os.path.join(riskmatrix_conf_dir,
13     panda_diagram_plotter_conf_filename)
14
15     with open(yaml_path, "r") as file:
16         panda_diagram_plotter_config = yaml.load(file, Loader=yaml.FullLoader)
17
18     return panda_diagram_plotter_config
19
20
21 def get_data(startpath, destination, tablefilename, datafile_path, datafile,
22 separator, decimal):
23     # Config Variables
24     if startpath == 'homedir':
25         directory = os.path.join(os.getcwd(), datafile_path)
26     else: # parentdir
27         directory = os.path.join(os.path.dirname(os.getcwd()), datafile_path)
28
29     # get the Datas as direct
30     data_path = os.path.join(directory, datafile)
31
32     # load datas from csv into dict
33     detected = chardet.detect(Path(data_path).read_bytes())
34     encoding = detected.get("encoding")
35
36     print(datafile, ';;', encoding)
37     panda_table_data = pd.read_csv(data_path, sep=separator, decimal=decimal,
38     encoding=encoding)
39
40     # return data
41     return panda_table_data
42
43 def create_panda_diagram_plotter(panda_diagram_plotter_config):
44     pdp_tables = panda_diagram_plotter_config.get('diagram_inventory')
45     for table_item in pdp_tables:
46         print(table_item)
47         startpath = panda_diagram_plotter_config.get('panda_diagram_plotter').get(
48         table_item).get('startpath')
49         destination = panda_diagram_plotter_config.get('panda_diagram_plotter').
50         get(table_item).get('destination_path')
51         imagename = panda_diagram_plotter_config.get('panda_diagram_plotter').get(
52         table_item).get('imagename')
53         datafile_path = panda_diagram_plotter_config.get('panda_diagram_plotter').
54         get(table_item).get('datafile_path')
55         datafile = panda_diagram_plotter_config.get('panda_diagram_plotter').get(
56         table_item).get('datafile')
57         if startpath == 'homedir':

```

```

47         directory = os.path.join(os.getcwd(), destination)
48     else: # parentdir
49         directory = os.path.join(os.path.dirname(os.getcwd()), destination)
50     image_path = os.path.join(directory, imagename)
51     separator = panda_diagram_plotter_config.get('panda_diagram_plotter').get(
52         table_item).get('separator')
53     decimal = panda_diagram_plotter_config.get('panda_diagram_plotter').get(
54         table_item).get('decimal')
55
56     # column operations
57     column_operations = panda_diagram_plotter_config.get(
58         'panda_diagram_plotter').get(table_item).get('column_operations').get('datas')
59
60     # group by / aggregation
61     groupby_values = panda_diagram_plotter_config.get('panda_diagram_plotter').
62     get(table_item).get('group_by')
63     group_by_function = panda_diagram_plotter_config.get(
64         'panda_diagram_plotter').get(table_item).get('group_by_function')
65
66     agg_funtion = panda_diagram_plotter_config.get('panda_diagram_plotter').
67     get(table_item).get('agg_funtion')
68     agg_colums = panda_diagram_plotter_config.get('panda_diagram_plotter').get(
69         table_item).get('agg_colums')
70     # dropping and renaming columns
71     drop_columns = panda_diagram_plotter_config.get('panda_diagram_plotter').
72     get(table_item).get('drop_columns')
73     rename_columns = panda_diagram_plotter_config.get('panda_diagram_plotter').
74     get(table_item).get('rename_columns')
75
76     # table filtering and sorting
77     where_clausel = panda_diagram_plotter_config.get('panda_diagram_plotter').
78     get(table_item).get('where_clausel')
79     order_by = panda_diagram_plotter_config.get('panda_diagram_plotter').get(
80         table_item).get('sorting').get('order_by')
81     sort_acending = panda_diagram_plotter_config.get('panda_diagram_plotter').
82     get(table_item).get('sorting').get('sort_acending')
83     sort_inplace = panda_diagram_plotter_config.get('panda_diagram_plotter').
84     get(table_item).get('sorting').get('sort_inplace')
85
86     # pivot settings
87     pivot = panda_diagram_plotter_config.get('panda_diagram_plotter').get(
88         table_item).get('pivot')
89     pivot_column = panda_diagram_plotter_config.get('panda_diagram_plotter').
90     get(table_item).get('pivot_columns')
91     pivot_value = panda_diagram_plotter_config.get('panda_diagram_plotter').
92     get(table_item).get('pivot_values')
93
94     # pivot_table settings

```

```
79     pivot_table = panda_diagram_plotter_config.get('panda_diagram_plotter').
80         get(table_item).get('pivot_table')
81     pivot_table_column = panda_diagram_plotter_config.get(
82         'panda_diagram_plotter').get(table_item).get('pivot_table').get(
83             'pivot_columns')
84     pivot_table_value = panda_diagram_plotter_config.get(
85         'panda_diagram_plotter').get(table_item).get('pivot_table').get(
86             'pivot_values')
87     pivot_table_agg_function = panda_diagram_plotter_config.get(
88         'panda_diagram_plotter').get(table_item).get('pivot_table').get(
89             'pivot_agg_func')
90     pivot_table_indexes = panda_diagram_plotter_config.get(
91         'panda_diagram_plotter').get(table_item).get('pivot_table').get(
92             'pivot_index').get('pivot_indexes')
93     pivot_table_indexes_visible = panda_diagram_plotter_config.get(
94         'panda_diagram_plotter').get(table_item).get('pivot_table').get(
95             'pivot_index').get('pivot_indexes_visible')
96     pivot_table_rename_indexes = panda_diagram_plotter_config.get(
97         'panda_diagram_plotter').get(table_item).get('pivot_table').get(
98             'pivot_index').get('pivot_rename_indexes')

99     # margins (subtotals)
100    margin = panda_diagram_plotter_config.get('panda_diagram_plotter').get(
101        table_item).get('margins').get('margin')
102    margin_name = panda_diagram_plotter_config.get('panda_diagram_plotter').
103        get(table_item).get('margins').get('margin_name')

104    # chart settings
105    chart = panda_diagram_plotter_config.get('panda_diagram_plotter').get(
106        table_item).get('chart-kind')
107    title = panda_diagram_plotter_config.get('panda_diagram_plotter').get(
108        table_item).get('title')
109    x_axis_title = panda_diagram_plotter_config.get('panda_diagram_plotter').
110        get(table_item).get('x-axis-title')
111    y_axis_title = panda_diagram_plotter_config.get('panda_diagram_plotter').
112        get(table_item).get('y-axis-title')
113    x_axis_columns = panda_diagram_plotter_config.get('panda_diagram_plotter').
114        get(table_item).get('x-axis-columns')
115    y_axis_columns = panda_diagram_plotter_config.get('panda_diagram_plotter').
116        get(table_item).get('y-axis-columns')
117    index = panda_diagram_plotter_config.get('panda_diagram_plotter').get(
118        table_item).get('chart-index')

119    # chart styles
120    grid = panda_diagram_plotter_config.get('panda_diagram_plotter').get(
121        table_item).get('chart-designs').get('grid')
122    legend = panda_diagram_plotter_config.get('panda_diagram_plotter').get(
123        table_item).get('chart-designs').get('legend')
```

## Diplomarbeit

```

109     rot = panda_diagram_plotter_config.get('panda_diagram_plotter').get(
110         table_item).get('chart-designs').get('rot')
111     fontsize = panda_diagram_plotter_config.get('panda_diagram_plotter').get(
112         table_item).get('chart-designs').get('fontsize')
113     figsize = panda_diagram_plotter_config.get('panda_diagram_plotter').get(
114         table_item).get('chart-designs').get('figsize')
115     stacked = panda_diagram_plotter_config.get('panda_diagram_plotter').get(
116         table_item).get('chart-designs').get('stacked')
117     secondary_y = panda_diagram_plotter_config.get('panda_diagram_plotter').get(
118         table_item).get('chart-designs').get('secondary_y')
119     stylelist = panda_diagram_plotter_config.get('panda_diagram_plotter').get(
120         table_item).get('chart-designs').get('stylelist')
121     subplots = panda_diagram_plotter_config.get('panda_diagram_plotter').get(
122         table_item).get('chart-designs').get('subplots')
123     autopct = panda_diagram_plotter_config.get('panda_diagram_plotter').get(
124         table_item).get('chart-designs').get('autopct')
125     loc = panda_diagram_plotter_config.get('panda_diagram_plotter').get(
126         table_item).get('chart-designs').get('loc')
127     bbox_to_anchor = panda_diagram_plotter_config.get('panda_diagram_plotter').get(
128         table_item).get('chart-designs').get('bbox_to_anchor')

129     # get the pandas (panda data)
130     panda_table_data = get_data(startpath, destination, imagename,
131         datafile_path, datafile, separator, decimal)

132     # filter by where clause
133     if where_clause:
134         panda_table_data = panda_table_data.query(where_clause)

135     # Drop unused columns
136     if drop_columns:
137         panda_table_data = panda_table_data.drop(columns=drop_columns)

138     # set aggregation functions
139     # if groupby_values and not agg_funtion and not pivot_column and not
140     # pivot_table_column:
141         if groupby_values and not (pivot_column or (pivot_table_column or
142             pivot_table_value or pivot_table_indizes)):
143             match group_by_function:
144                 case 'max':
145                     panda_table_data = panda_table_data.groupby(groupby_values,
146                         as_index=False).max()
147                 case 'min':
148                     panda_table_data = panda_table_data.groupby(groupby_values,
149                         as_index=False).min()
150                 case 'head':
151                     panda_table_data = panda_table_data.groupby(groupby_values,
152                         as_index=False).head()

```

```

141             case 'sum':
142                 panda_table_data = panda_table_data.groupby(groupby_values,
143                     as_index=False).sum()
144             case 'mean':
145                 panda_table_data = panda_table_data.groupby(groupby_values,
146                     as_index=False).mean()
147         else:
148             panda_table_data = panda_table_data
149
150         # pivot if pivot is selected
151         if pivot_table_column or pivot_table_value or pivot_table_indizes:
152             if type(pivot_table_agg_function) is list:
153                 agg_tuple = tuple(pivot_table_agg_function)
154                 panda_table_data = pd.pivot_table(panda_table_data, index=
155                     pivot_table_indizes,
156                     columns=pivot_table_column,
157                     values=pivot_table_value,
158                     aggfunc=agg_tuple, margins=
159                     margin, margins_name=margin_name)
160             elif type(pivot_table_agg_function) is dict:
161                 panda_table_data = pd.pivot_table(panda_table_data, index=
162                     pivot_table_indizes,
163                     columns=pivot_table_column,
164                     values=pivot_table_value,
165                     aggfunc=pivot_table_agg_function,
166                     margins=margin, margins_name=
167                     margin_name)
168             else:
169                 panda_table_data = pd.pivot_table(panda_table_data, index=
170                     pivot_table_indizes,
171                     columns=pivot_table_column,
172                     values=pivot_table_value,
173                     aggfunc=pivot_table_agg_function
174                     , margins=margin,
175                     margins_name=margin_name)
176
177         # set column operations
178         if column_operations:
179             for column_ops in column_operations:
180                 operation_function = panda_diagram_plotter_config.get(
181                     'panda_diagram_plotter').get(table_item).get('column_operations').get(
182                     'operations').get(column_ops).get('operation_function')
183                 operation_columns = panda_diagram_plotter_config.get(
184                     'panda_diagram_plotter').get(table_item).get('column_operations').get(
185                     'operations').get(column_ops).get('columns')
186                 operation_axis = panda_diagram_plotter_config.get(
187                     'panda_diagram_plotter').get(table_item).get('column_operations').get(
188                     'operations').get(column_ops).get('axis_number')

```

```
172         match operation_function:
173             case 'max':
174                 panda_table_data[column_ops] = panda_table_data[
175                     operation_columns].max()
176             case 'min':
177                 panda_table_data[column_ops] = panda_table_data[
178                     operation_columns].min()
179             case 'head':
180                 panda_table_data[column_ops] = panda_table_data[
181                     operation_columns].head()
182             case 'sum':
183                 panda_table_data[column_ops] = panda_table_data[
184                     operation_columns].sum(axis=operation_axis)
185             case 'mean':
186                 panda_table_data[column_ops] = panda_table_data[
187                     operation_columns].mean()
188             case 'diff':
189                 panda_table_data[column_ops] = panda_table_data[
190                     operation_columns[1]] - panda_table_data[operation_columns[0]]
191
192
193     # order by
194     if order_by:
195         panda_table_data.sort_values(by=order_by, inplace=sort_inplace,
196                                     ascending=sort_acending)
197
198     # rename columns
199     if rename_columns:
200         panda_table_data = panda_table_data.rename(columns=rename_columns)
201
202     # set indices
203     if index:
204         index_values = panda_table_data.get(index)
205         #panda_table_data.set_index(index_values)
206         panda_table_data = panda_table_data.set_index(index)
207
208     # rename indices
209     if pivot_table_rename_indizes:
210         panda_table_data = panda_table_data.rename_axis(index=
211             pivot_table_rename_indizes)
212
213     # Plotter
214     # Plotter Process starts here!
215     if autopct:
216         panda_chart_plot = panda_table_data.plot(kind=chart, title=title, y=
217             y_axis_columns, x=x_axis_columns, xlabel=x_axis_title,
218                                         ylabel=y_axis_title, grid=grid,
219                                         stacked=stacked, legend=legend,
```

## Diplomarbeit

```

210                                     secondary_y=secondary_y, subplots
211                                     =subplots, rot=rot, fontsize=fontsize,
212                                     figsize=figsize, autopct=autopct)
213
214     else:
215         panda_chart_plot = panda_table_data.plot(kind=chart, title=title, y=
216             y_axis_columns, x=x_axis_columns, xlabel=x_axis_title,
217                                         ylabel=y_axis_title, grid=grid,
218                                         stacked=stacked, legend=legend,
219                                         secondary_y=secondary_y, subplots
220                                         =subplots, rot=rot, fontsize=fontsize,
221                                         figsize=figsize)
222
223
224     match chart:
225
226         case 'pie':
227             panda_chart_plot[0].legend(loc=loc, bbox_to_anchor=bbox_to_anchor)
228             plt = panda_chart_plot[0].get_figure()
229             plt.savefig(image_path, bbox_inches='tight')
230
231         case _:
232             plt = panda_chart_plot.get_figure()
233             plt.savefig(image_path, bbox_inches='tight')
234
235
236
237
238     return "blade runner"
239
240 panda_diagram_plotter_config = load_configuration('pandas_data_chart_plotter.conf.
241                                                 yaml')
242
243 create_panda_diagram_plotter(panda_diagram_plotter_config)

```

Listing 45: Python LaTex - pandas\_data\_chart\_plotter.py CSV - Diagramm

## IX.XII pandas\_data\_chart\_plotter\_conf.yaml

```

1 diagram_inventory:
2   - "tps_mixed"
3   - "db_inventory_per_rdbms"
4   - "db_inventory_per_os"
5 panda_diagram_plotter:
6   tps_mixed:
7     id: "tps_mixed"
8     startpath: "parentdir"
9     destination_path: "source/pandas_data_chart_plotter"
10    imagename: "tps_mixed.png"
11    datafile_path: "source/pandas_data_chart_plotter"
12    datafile: "tps_evaluation.csv"
13    separator: ","
14    decimal: "."
15    x-axis-columns: "Varianten"
16    y-axis-columns:

```

## Diplomarbeit

```
17      - "2. Iteration"
18      - "3. Iteration"
19      - "4. Iteration"
20  x-axis-title: "Varianten"
21  y-axis-title: "Transaktionen pro Sekunde (tps) Bsp."
22  title: "Transaktionen pro Sekunden - mixed"
23  chart-index:
24  chart-kind: "bar"
25  chart-designs:
26    subplots: False
27    grid: True
28    legend: True
29    rot:
30    fontsize:
31    stacked: False
32    secondary_y: False
33    stylelist:
34    figsize:
35    autopct:
36    loc:
37    bbox_to_anchor:
38  group_by:
39  group_by_function:
40  agg_function:
41  agg_columns:
42  drop_columns:
43    - "tps_1_iteration"
44    - "tps_typ"
45  column_operations:
46    datas:
47  pivot:
48    pivot_columns:
49    pivot_values:
50  pivot_table:
51    pivot_index:
52    pivot_indexes_visible: False
53    pivot_rename_indexes:
54    pivot_columns:
55    pivot_values:
56    pivot_agg_func:
57  rename_columns:
58    variante: "Varianten"
59    tps_2_iteration: "2. Iteration"
60    tps_3_iteration: "3. Iteration"
61    tps_4_iteration: "4. Iteration"
62    where_clause1: "tps_typ == 'mixed'"
63  sorting:
64    order_by:
```

## Diplomarbeit

```
65     sort_acending: True
66     sort_inplace: True
67     margins:
68         margin: False
69         margin_name:
70     db_inventory_per_rdbms:
71         id: "db_inventory_per_rdbms"
72         startpath: "parentdir"
73         destination_path: "source/pandas_data_chart_plotter"
74         imagename: "db_inventory_per_rdbms.png"
75         datafile_path: "source/tables"
76         datafile: "inventory.csv"
77         separator: ","
78         decimal: "."
79         x-axis-columns: "RDBMS"
80         y-axis-columns:
81         x-axis-title:
82         y-axis-title:
83         title: "Datenbankinventor - Pro RDBMS"
84         chart-index: "RDBMS"
85         chart-kind: "pie"
86         chart-designs:
87             subplots: True
88             grid: False
89             legend: True
90             rot:
91             fontsize:
92             stacked: False
93             secondary_y: False
94             stylelist:
95             figsize: !!python/tuple [25,10]
96             autopct: '%1.0f%'
97             loc: "best"
98             bbox_to_anchor:
99             group_by:
100             - "rdbms"
101             group_by_function: "sum"
102             agg_funtion:
103             agg_columns:
104             - "rdbms"
105             drop_columns:
106             - "server"
107             - "os"
108             - "version"
109             - "releasedate"
110             - "eol"
111             - "age"
112             - "eol_since"
```

## Diplomarbeit

```
113     - "comment"
114     - "appliance"
115     column_operations:
116         datas:
117     pivot:
118         pivot_columns:
119         pivot_values:
120     pivot_table:
121         pivot_index:
122             pivot_indexes_visible: False
123             pivot_rename_indexes:
124         pivot_columns:
125         pivot_values:
126         pivot_agg_func:
127     rename_columns:
128     rdbms: "RDBMS"
129     instance : "Instanz"
130     databases : "Datenbanken"
131     appliance: "Appliance"
132     where_clauses:
133     sorting:
134         order_by:
135             - "rdbms"
136         sort_acending: True
137         sort_inplace: True
138     margins:
139         margin: False
140         margin_name:
141     db_inventory_per_os:
142         id: "db_inventory_per_os"
143         separator: ","
144         decimal: "."
145         startpath: "parentdir"
146         destination_path: "source/pandas_data_chart_plotter"
147         datafile_path: "source/tables"
148         datafile: "inventory.csv"
149         imagename: "db_inventory_per_os.png"
150         decimal_format:
151         x-axis-columns: "RDBMS"
152         y-axis-columns:
153         x-axis-title:
154         y-axis-title:
155         title: "Datenbankinventor - Pro OS"
156         chart-index:
157         chart-kind: "pie"
158         chart-designs:
159             subplots: True
160             grid: False
```

## Diplomarbeit

```
161     legend: False
162     rot:
163     fontsize:
164     stacked: False
165     secondary_y: False
166     stylelist:
167     figsize: !!python/tuple [25,10]
168     autopct: '%1.0f%%'
169     loc: "upper center"
170     bbox_to_anchor: !!python/tuple [0,0]
171 group_by:
172   - "rdbms"
173   - "os"
174 group_by_function: "sum"
175 agg_funtion:
176 agg_columns:
177   - "os"
178 drop_columns:
179   - "server"
180   - "version"
181   - "releasedate"
182   - "eol"
183   - "age"
184   - "eol_since"
185   - "comment"
186 #   - "appliance"
187 column_operations:
188   datas:
189   operations:
190     dauer_summe:
191       operation_function:
192       axis_number:
193       columns:
194     pivot:
195       pivot_columns:
196       pivot_values:
197     pivot_table:
198       pivot_index:
199       pivot_indizes:
200         - "os"
201         - "rdbms"
202       pivot_indizes_visible: True
203       pivot_rename_indizes:
204         os: "OS"
205         rdbms: "RDBMS"
206       pivot_columns:
207       pivot_values:
208       pivot_agg_func:
```

```
209     instance: "sum"
210     databases: "sum"
211     appliance: "sum"
212     transpose: True
213     rename_columns:
214     rdbms: "RDBMS"
215     instance : "Instanz"
216     databases : "Datenbanken"
217     os : "OS"
218     appliance: "Appliance"
219     where_clausel:
220     sorting:
221     order_by:
222     sort_acending: False
223     sort_inplace: True
224     margins:
225     margin: False
226     margin_name:
227     table_styles:
228     selector: "caption"
229     props:
230     caption-side: "below"
231     position: "H"
232     sparse_columns: True
233     longtable: True
234     resize_textwidth: False
235     linebreak_columns:
236     table_header: True
```

Listing 46: Python LaTex - pandas\_data\_chart\_plotter\_conf.yaml - Konfigurationsdatei - CSV - Diagramme