



**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>Abkürzungen</b>	<b>4</b>
<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 . . . . .	9
1.2 Zieldefinition . . . . .	13
1.3 Abgrenzungen . . . . .	16
1.4 Abhängigkeiten . . . . .	18
1.5 Risikomanagement . . . . .	19
1.5.1 Riskcontrolling . . . . .	22
1.6 Vorgehensweise und Methoden . . . . .	25
1.7 Projektmanagement . . . . .	25
1.7.1 Projektcontrolling . . . . .	26
1.7.2 GANTT-Diagramm . . . . .	27
1.8 Status-Reports . . . . .	29
1.8.1 Initialer Statusbericht . . . . .	29
1.8.2 Zweiter Statusbericht . . . . .	30
1.9 Expertengespräche . . . . .	31
<b>2 Umsetzung</b>	<b>32</b>
2.1 Evaluation . . . . .	32
2.1.1 Exkurs Architektur . . . . .	32
2.1.2 Erheben und Gewichten der Anforderungen . . . . .	40
2.1.3 Testziele erarbeiten . . . . .	53
2.1.4 PostgreSQL Benchmarking . . . . .	53
2.1.5 Analyse gängiger PostgreSQL HA Cluster Lösungen . . . . .	57
2.1.6 Vorauswahl . . . . .	83
2.1.7 Installation verschiedener Lösungen . . . . .	84
2.1.8 Gegenüberstellung der Lösungen . . . . .	88
2.1.9 Entscheid . . . . .	88

2.2 Aufbau und Implementation Testsystem . . . . .	88
2.2.1 Bereitstellen der Grundinfrastruktur . . . . .	88
2.2.2 Installation und Konfiguration PostgreSQL HA Cluster . . . . .	88
2.2.3 Technical Review der Umgebung . . . . .	88
2.3 Testing . . . . .	88
2.3.1 Testing . . . . .	88
2.3.2 Protokollierung . . . . .	88
2.3.3 Review und Auswertung . . . . .	89
2.4 Troubleshooting und Lösungsfindung . . . . .	89
<b>3 Resultate</b>	<b>90</b>
3.1 Zielüberprüfung . . . . .	90
3.2 Schlussfolgerung . . . . .	90
3.3 Weiteres Vorgehen / offene Arbeiten . . . . .	90
3.4 Persönliches Fazit . . . . .	90
<b>Abbildungsverzeichnis</b>	<b>91</b>
<b>Tabellenverzeichnis</b>	<b>94</b>
<b>Listings</b>	<b>95</b>
<b>Literatur</b>	<b>96</b>
<b>Glossar</b>	<b>101</b>
<b>Anhang</b>	<b>i</b>
I Arbeitsrapport . . . . .	i
II Protokoll - Fachgespräche . . . . .	ii
III Kommentare / Anmerkungen . . . . .	iii
IV zotero.py . . . . .	iv
V zotero_bibtex_configuration.yaml . . . . .	x
VI zotero_biblatex_keystore.yaml . . . . .	x
VII riskmatrix.py . . . . .	xvii
VIII riskmatrix_plotter_conf.yaml . . . . .	xxi
IX riskmatrix_xy_axis_tuple_matrix.yaml . . . . .	xxv
X cost_benefit_diagram.py . . . . .	xxvi
XI cost_benefit_diagram_plotter_conf.yaml . . . . .	xxviii
XII pandas_dataframe_to_latex_table.py . . . . .	xxviii
XIII csv_to_latex_diplomarbeit.yaml . . . . .	xxxviii

## 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

# 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

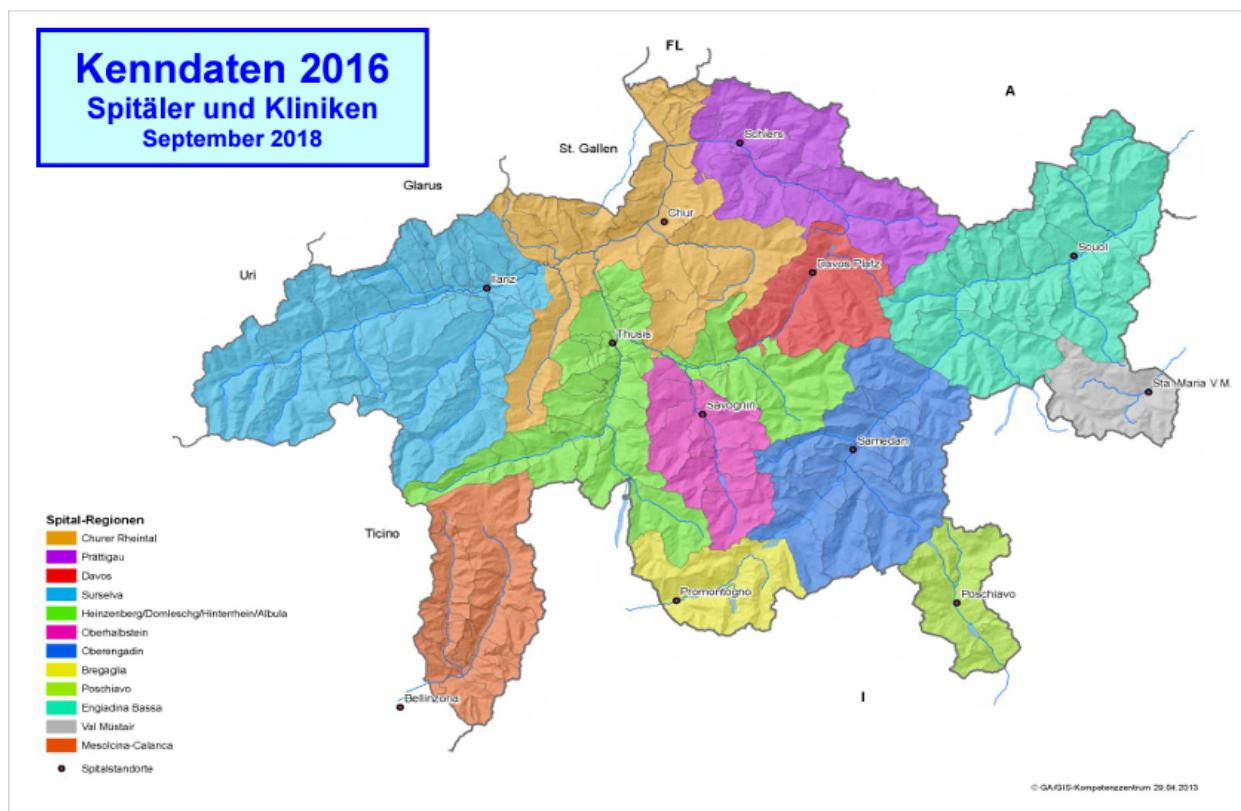


Abbildung 1.1: Spitalregionen Kanton Graubünden[38]

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[63]

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

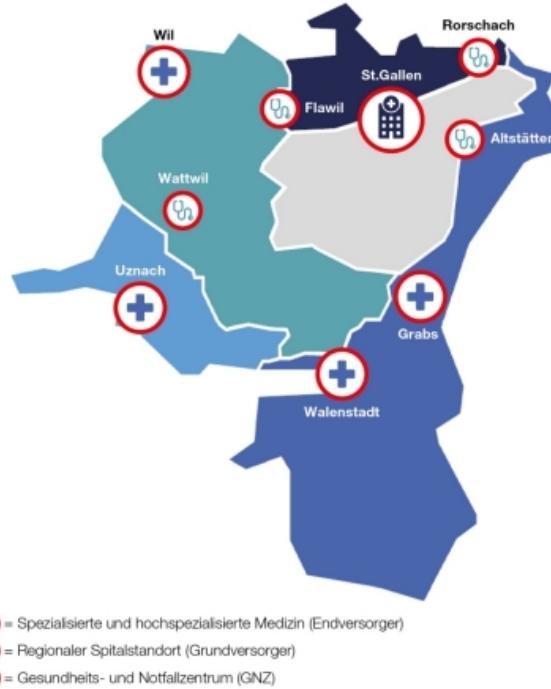


Abbildung 1.3: Spitalregionen / Spitalstrategie Kanton St. Gallen[32]

### 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

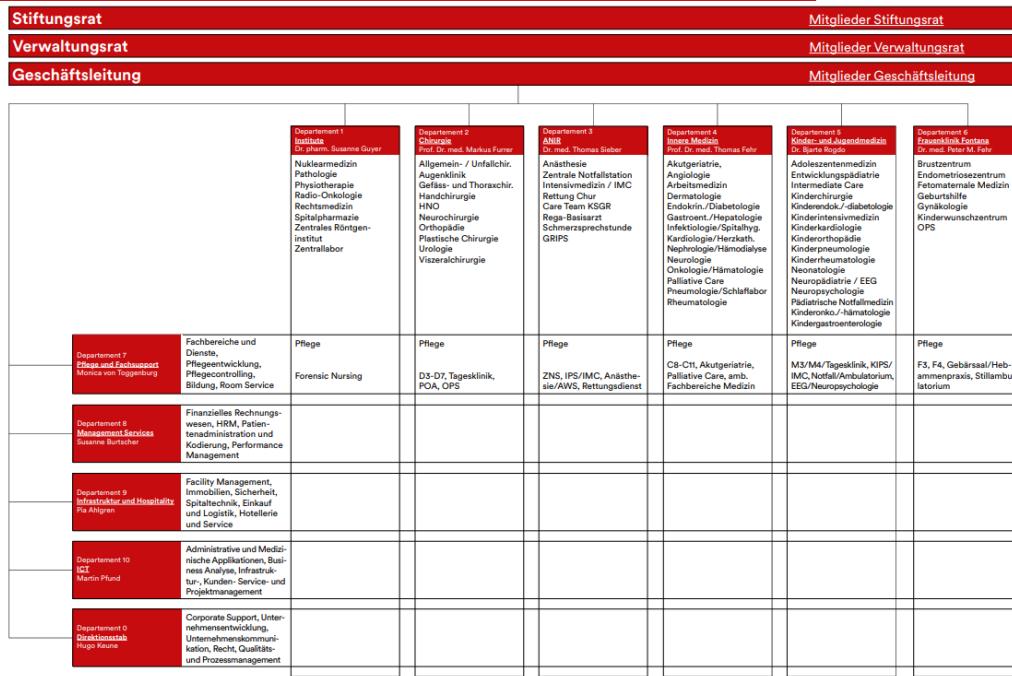
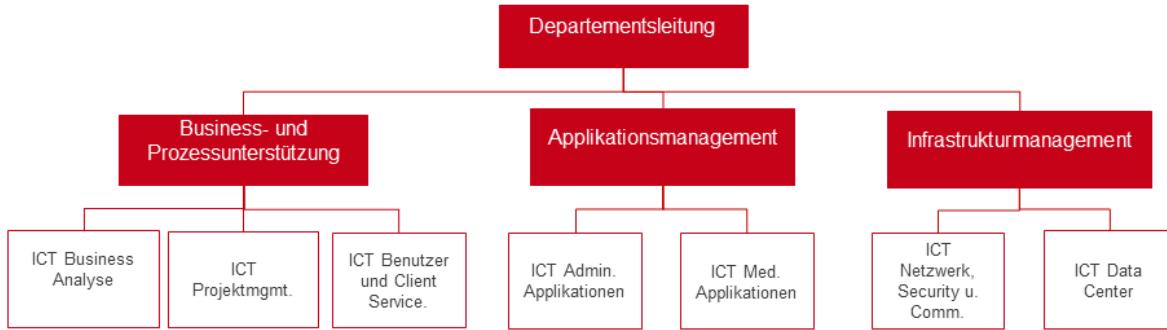


Abbildung 1.4: Organigramm Kantonsspital Graubünden

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

Kantonsspital  
Graubünden



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 Containern), 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.

## 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*[29] 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

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

Tabelle 1.2: Datenbankinventar

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
	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
<b>Gesamtergebnis</b>		7	105	80

Tabelle 1.3: Datenbankinventor - Nach Betriebssystemen aufgeschlüsselt

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:

Identifikation	ID	Schutzziel	Referenz BSI 200-3	Risiko	Beschreibung / Ursache	Auswirkung	Abschätzung		Behandlung		Zielwert	
							WS	SM	Massnahmen ergreifen?	WS	SM	
1 I	G0.22	Manipulation von Informationen			Durch veraltete Systeme die zudem unterschiedlich gut gehärtet und gesichert sind (z.B. durch Verschlüsselung des Verkehrs oder der Daten auf dem Storage), besteht das Risiko das Daten manipuliert werden.	Die Auswirkungen reichen von einer Fehlfunktion des Systems bis hin zum vollständigen Verlust der Integrität der Daten	2	4	Ja	1	2	Best-Practice bei Härtung der Systeme. Redundanzen einführen
2 A	G0.25	Ausfall von Geräten oder Systemen			Manche Datenbanken und deren Betriebssysteme sind sehr alt und sehr lange im Einsatz. Einige dieser Systeme sind schon so alt, das keine Hotfixes, Patches und Updates mehr erhältlich sind.	Sobald keine HA-Architektur aufgebaut wurde, ist die Verfügbarkeit ernsthaft gefährdet resp. die Applikation steht nicht mehr zur Verfügung.	4	4	Ja	2	2	Redundanzen einführen
3 C, I, A	G0.26	Fehlfunktion von Geräten oder Systemen			Hierdurch entsteht das Risiko, das Systeme Ausfallen. Manche Datenbanken und deren Betriebssysteme sind sehr alt und sehr lange im Einsatz. Einige dieser Systeme sind schon so alt, das keine Hotfixes, Patches und Updates mehr erhältlich sind.	Fehlfunktionen können innerhalb von Datenbanksystemen die Datenkonsistenz verletzen, Daten können verloren gehen oder ungewollt von Dritten und unberechtigten Personen eingesesehen werden. Systeme könnten nicht mehr oder nur noch eingeschränkt verfügbar werden.	2	4	Ja	2	2	Systeme zentralisieren Lifecycle etablieren
4 C, I, A	G0.27-1	Ressourcenmangel (personelle Ressourcen)			Allerdings versuchen Datenbanksysteme, die Auswirkungen so gering wie möglich zu halten. Aufgrund der sehr heterogenen Landschaft ist der Administrationsaufwand für die jetzigen Systeme sehr gross. Zu gross, als das für jede Datenbank und deren Betriebssystem die notwendige Zeit für eine bedarfsgerechte Administration erbracht werden kann.	Die Auswirkungen können vielfältig sein, abhängig davon welcher Aspekt unter dem Ressourcenmangel leidet.	3	3	Ja	2	3	Systeme zentralisieren
5 A	G0.27-2	Ressourcenmangel (technische Ressourcen)			Dadurch bleiben Fehler länger unentdeckt, Hotfixes, Patches, Updates und Upgrades können nicht oder nicht zur richtigen Zeit eingespielt werden.	Grundsätzlich wird aber sowohl die Vertraulichkeit, Integrität und Verfügbarkeit gefährdet.						
6 C, I, A	G0.31	Fehlerhafte Nutzung oder Administration von Geräten und Systemen			Bei einem akuten Problemfall ist nicht garantiert, dass die Leute erreichbar sind, die notwendig sind	Wenn die CPU- und Memory-Usage über einen gewissen Schwellwert geht, fängt das Betriebssystem an zu Priorisieren. Dies wird primär der Endanwender in form von Performance Einbussen bemerken. Im schlimmsten Fall steht eine Anwendung nicht mehr zur Verfügung.	2	2	Ja	1	2	Monitoring verschärfen
7 C, I, A	G0.32	Missbrauch von Berechtigungen			Kann auftreten wenn Ressourcenwachstum zu spät bemerkt wird. So kann die CPU Usage oder das Memory Usage schnell anwachsen. Auch der Storage eines Betriebssystems kann nicht mehr ausreichend für ein System werden.	Gefährlicher sind Storage Overflows, besonders wenn die Datenbank nicht mehr alle Informationen schreiben konnte, die sie für einen korrekten Neustart benötigte.						
8 A, I	G0.45	Datenverlust			Durch die Vielfalt an Datenbankversionen und Betriebssystemen und Plattformen worauf diese betrieben werden, besteht allen voran das Risiko einer Fehlerhaften Administration und Konfiguration.	Doch die folgen bleiben nichtsdestotrotz überschaubar. 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.	4	3	Ja	2	3	Systeme zentralisieren
					Obwohl das Microsoft Active Directory die Zentrale Benutzerverwaltung ist, sind die wenigsten Datenbanken an dieses angeschlossen. Hinzu kommt der umstand, das in der Vergangenheit jeder Softwarelieferant sein eigenes Benutzerkonzept mitgebracht hat, auch bei den Datenbankzugängen.	Daraus erschießt sich das auch bei diesem Risiko die Vertraulichkeit, Integrität und Verfügbarkeit gefährdet ist.						
					Multipiziert mit der Anzahl der unterschiedlichsten Datenbanken, Betriebssystemen und Applikationen entsteht das Risiko, das Berechtigungen Wissentlich oder Unwissentlich missbraucht werden. Verschiedene Datenbanken sind Standalone Cluster (Instanzen) welche über keinen Failover-Mechanismus verfügen.	Der Wissentliche oder Unwissentliche Missbrauch von Berechtigungen kann verheerende Auswirkungen haben. Unter anderem können Daten missbräuchlich abgezogen werden, Daten manipuliert oder das ganze System komplett zerstört werden.	2	4	Ja	2	2	Systeme zentralisieren Übergreifendes Berechtigungskonzept einführen Monitoring der Zugriffe
					Zudem wurden die meisten Datenbanken nur mittels Snapshots oder einem Filesystem Backup gesichert, nicht über eine eigentliche Sicherung mittels WAL. Gerade die fehlende WAL-Archivierung führt im Backupfall dazu, das alle Transaktionen die zwischen dem letzten Backup nicht mehr vorhanden sind.	Aus dem Risiko ergeben sich zwei Auswirkungen, die aber beide ein hohes Mass an Schaden verursachen können.						
					Hinzu kommt, das für die meisten Datenbanken hohe Sicherungsintervalle von einmal pro Stunde oder gar nur einmal am Tag gewählt wurde.	Erstens könnten Backups gar nicht mehr Wiederhergestellt werden, dies hätte dann einen Totalen Datenverlust zur Folge. Die zweite Ursache erwächst auf der fehlenden WAL-Archivierung, dadurch können zwar die Daten bis zu einem Zeitpunkt X Wiederhergestellt werden allerdings sind diese dann nicht zwingend Konsistent.	4	5	Ja	1	3	Systeme zentralisieren Einheitliches Backupkonzept Regelmäßige Restore-Tests

Tabelle 1.4: Risiko-Matrix aktuelle Situation PostgreSQL Datenbanken

Risiko Cockpit PostgreSQL Datenbanken KSGR

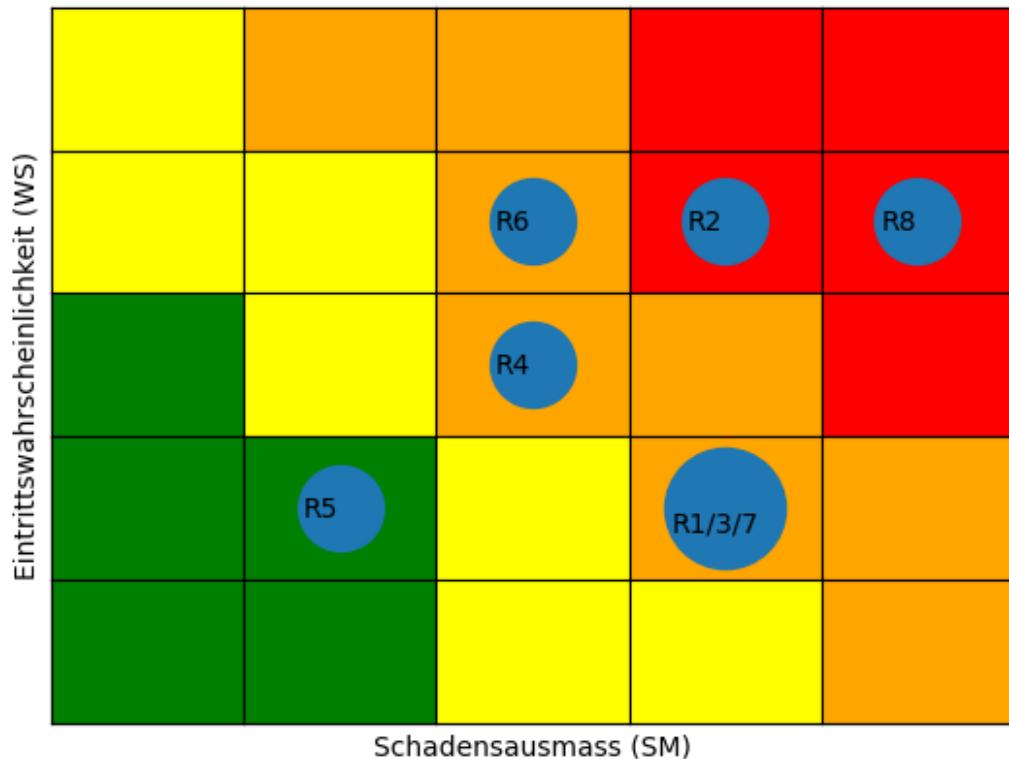


Abbildung 1.6: Risiken bestehende Lösung

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 gesenkt werden:

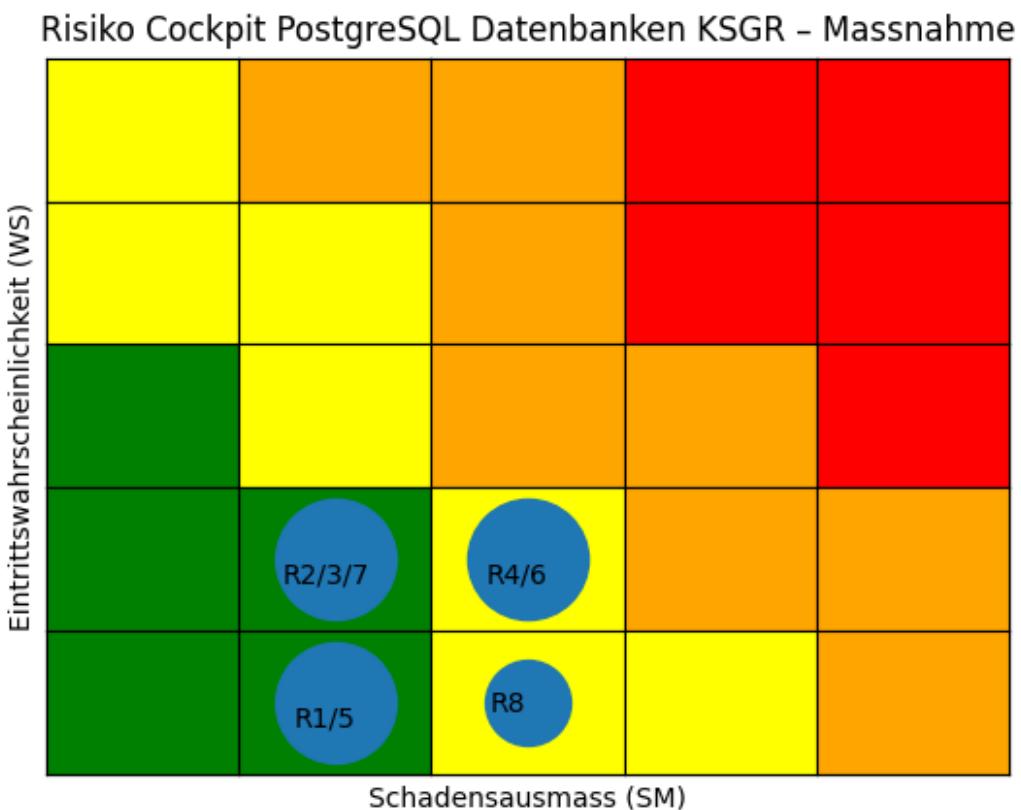


Abbildung 1.7: Risiken bestehende Lösung mit Massnahmen

## 1.2 Zieldefinition

Das administrieren einer PostgreSQL Datenbank umfasst i.d.R. [47, 53] 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[26]. 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[3]. 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[26]. 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[3].	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[77] folgende Abgrenzungen:

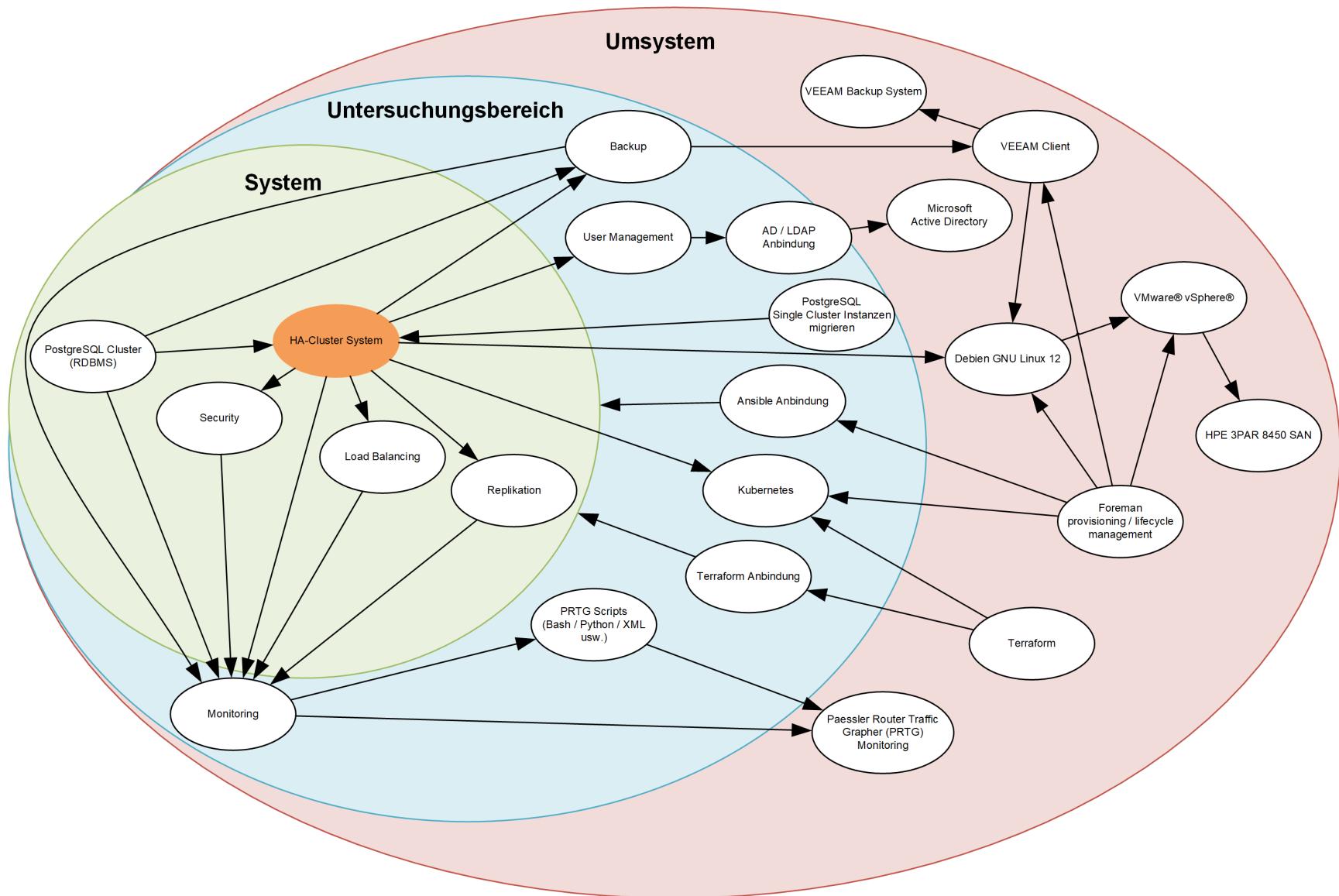


Abbildung 1.8: Systemabgrenzung

## 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.
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	Dies wird mehr Storage in Anspruch nehmen.
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	Projekt kann nicht zeitgemäß abgeschlossen werden
5	GitLab	Sicherung	Sicherung von Konfigurationen, Scripts usw.	GitLab ist implementiert und betriebsbereit.	GitLab steht nicht mehr zur Verfügung	Keine Versionierung und Teile Sicherungen mehr von Konfigurationsfiles, Scripts usw.
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.	Zertifikate können aus Zeitgründen nicht in der Evaluationsphase eingesetzt werden. Für die Testphase müssen Zertifikate manuell ausgestellt werden.
7	Veeam Kasten K10[4]	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

## 1.5 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. 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	4	1	Nein			
8	Foreman nicht betriebsbereit	Ms müssen von Hand provisioniert werden. Dies bedeutet einen massiven Mehraufwand und verzögert ggf. die Evaluationsphase und mit Sicherheit die Installationsphase	3	5	Ja	3	4	Massnahmen ergreifen um die manuelle Installation so effizient wie möglich zu gestalten.

Tabelle 1.10: Risiko-Matrix der Diplomarbeit

Daraus ergibt sich folgende Risikomatrix

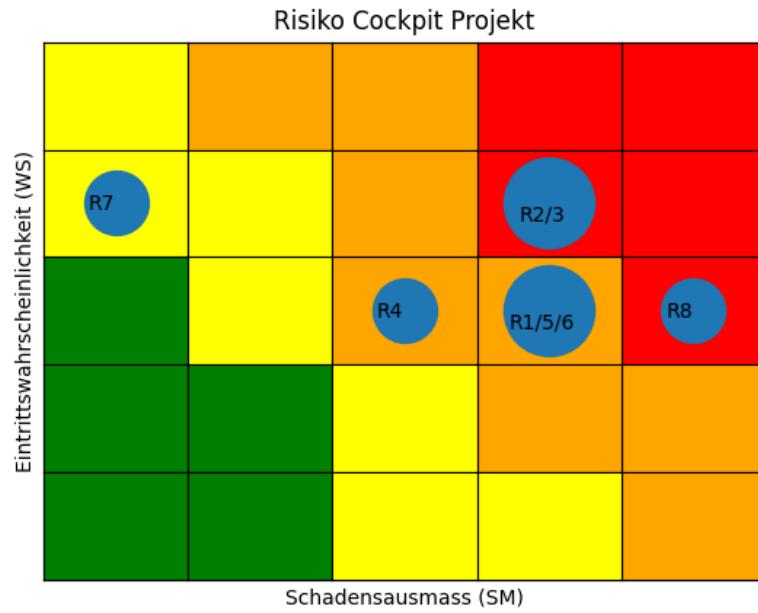


Abbildung 1.9: Projektrisiken

Mit den entsprechenden Massnahmen können die Risiken gesenkt werden:

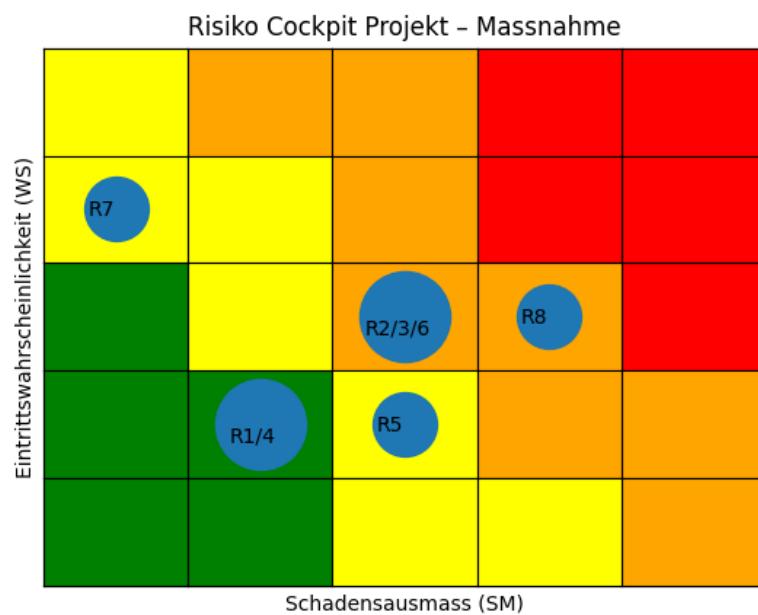


Abbildung 1.10: Projektrisiken mit Massnahmen

## 1.5.1 Riskcontrolling

### 1.5.1.1 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[4] nicht betriebsbereit	Abhängigkeit zum KSGR k8s Projekt.							

Tabelle 1.11: Neu Erkannte / Erfasste Risiken



### 1.5.1.2 Assessment 21.03.2024

ID	Risiko	Assessment-Datum	WS	SM	Status	Ergriffene 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[4] nicht betriebsbereit	21.03.2024	5	5	sehr hoch	noch keine		

Tabelle 1.12: Risiko-Assessment 21.03.2024

Risiko Cockpit Projekt - Assessment 21.03.2024

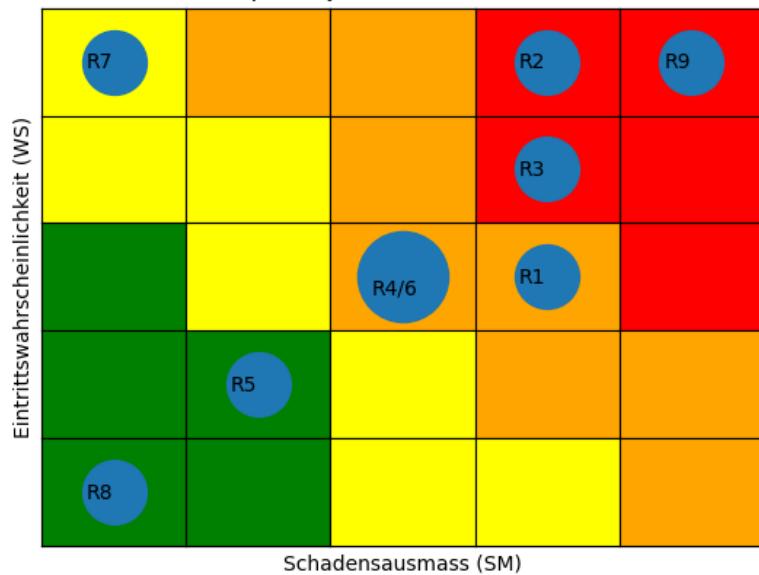


Abbildung 1.11: Riskikomatrix - Assessment 21.03.2024

## 1.6 Vorgehensweise und Methoden

## 1.7 Projektmanagement

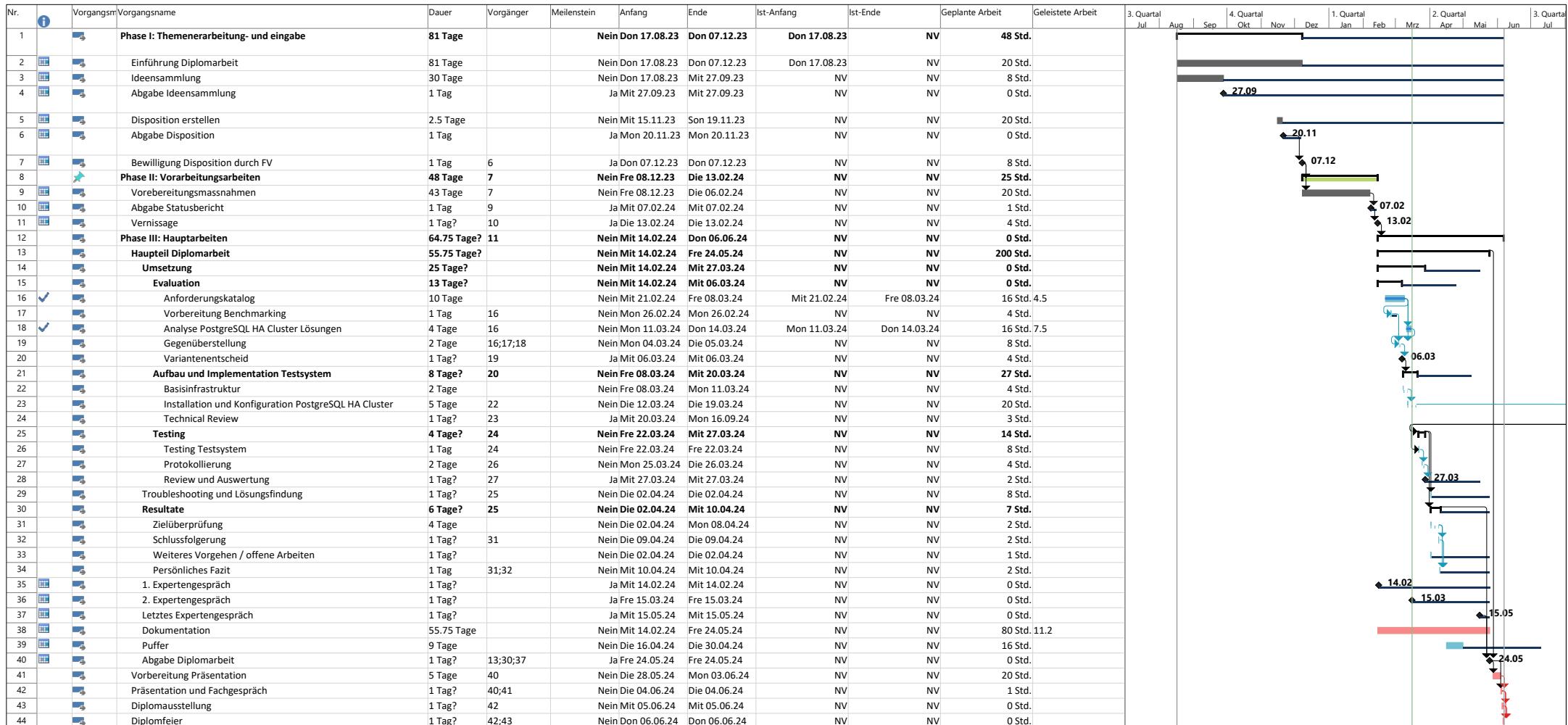
### 1.7.1 Projektcontrolling

Phase	Subphase	Dauer [h]	Geplante Dauer [h]	Verbleibende Zeit [h]
0 Dokumentation	-	22.2	80	57.8
1 Evaluation	Analyse PostgreSQL HA Cluster Lösungen	10.0	16	6.0
2 Evaluation	Anorderungskatalog	4.5	16	11.5
3 Evaluation	Vorbereitung Benchmarking	5.0	4	-1.0

Tabelle 1.13: Projektcontrolling

### 1.7.2

#### GANNT-Diagramm



Projekt: Diplomarbeit - PostgreSQL  
Datum: Sam 16.03.24



## 1.8 Status-Reports

### 1.8.1 Initialer Statusbericht

PostgreSQL HA Cluster - Konzeption und Implementation		ICT Projektstatusbericht 13.02.2024	
Projektbeschreibung	Evaluation und Implementation PostgreSQL HA Cluster	Priorität	PMA
ICT verantw. Person	Michael Graber	-	
Status	Ampel	Tendenz	Begründung
<b>Gesamtprojekt</b>			
Zeitplanung	■	↓	Projekt ist umfangreich und hat viele Teilsaspekte, die es zu planen und berücksichtigen gilt.
Ressourcen	▲	↓	Parallel läuft das Grossprojekt Erneuerung HP UX Plattform, wo die bestehende HP-UX Plattform durch eine Oracle Exadata Cloud@Customer Plattform abgelöst wird. Ab dem Zeitpunkt der Lieferung der Hardware werden die Oracle Datenbanken der Kernapplikation auf die neue Plattform migriert. Dies über das gesamte Jahr und auch während der Diplomarbeit sehr viele Ressourcen binden.
Kosten	●	↑	Kosten sind noch im Soll-Bereich
<b>Tätigkeiten vergangene Berichtsperiode</b>		<b>Tätigkeiten nächste Berichtsperiode</b>	
<ul style="list-style-type: none"> <li>- Dokumentenstruktur erstellt</li> <li>- Projektplanung erstellt</li> <li>- Versnissage vorbereitet</li> <li>- Statusbericht erstellt</li> </ul>		<ul style="list-style-type: none"> <li>- Anforderungskatalog erarbeiten</li> <li>- Vorbereitung Benchmarking</li> </ul>	
# erledigte Lieferobjekte (inkl. allfällige Links)		Status	Erfüllungsgrad
LO-001	Anforderungskatalog	in arbeit	<div style="width: 60%;">60%</div>
LO-002	Vorbereiten Benchmarking	in arbeit	<div style="width: 0%;">0%</div>
LO-003			
LO-004			
LO-005			
# Risiken	Auswirkungsgrad	Massnahmen	Verantw.
R-001	■	Organisation und Selbstmanagement	
R-002	■	Ressourcen reservieren	
R-003	■	Monitoring vorgängig ausbauen und Massnahmen definieren	
R-004	■	Werkzeuge im Vorfeld definieren und bereitstellen	
<b>Kostenübersicht</b>		<b>Abhängigkeiten zu anderen Projekten</b>	
Verfügbare Finanzen bis Ende Projekt: nachr. + 200k = 24'000 CHF		Erneuerung HP UX Plattform 6002201 KSGR Provisioning System (KPS) -->Foreman Umgebung	
		Ressourcen reservieren Massnahmen ergreifen um die manuelle Installation so effizient wie möglich zu gestalten	
<b>Bemerkungen / Informationen</b>		<b>Anträge</b>	
Eingereicht	Geprüft	<b>Bemerkungen/Auftrag PMO</b>	
PL:	PMO:		
Datum:	Datum:		
# erledigte Lieferobjekte (inkl. allfällige Links)			

Tabelle 1.14: Initialer Statusbericht

## 1.8.2

## Zweiter Statusbericht

PostgreSQL HA Cluster - Konzeption und Implementation		ICT Projektstatusbericht 18.03.2024	
Projektbeschreibung	Evaluation und Implementation PostgreSQL HA Cluster	Priorität	-
ICT verantw. Person	Michael Gruber	PMA	-
Status	Ampel	Tendenz	Begründung
Gesamtprojekt			In Verzug. Grossprojekt Erneuerung HP UX Plattform nimmt viel Zeit in Anspruch. Hinzu kommt, das die Analyse gängiger PostgreSQL HA Lösungen ebenfalls viel Zeit kostet. Dokumentationsaufwand unterschätzt.
Zeitplanung			In Verzug. Grossprojekt Erneuerung HP UX Plattform nimmt viel Zeit in Anspruch. Hinzu kommt, das die Analyse gängiger PostgreSQL HA Lösungen ebenfalls viel Zeit kostet. Dokumentationsaufwand unterschätzt.
Ressourcen			Parallel läuft das Grossprojekt Erneuerung HP UX Plattform, wo die bestehende HP UX Plattform durch eine Oracle Exadata Cloud@Customer Plattform abgelöst wird. Ab dem Zeitpunkt der Lieferung der Hardware werden die Oracle Datenbanken der Kernapplikation auf die neue Plattform migriert. Dies über das gesamte Jahr und auch während der Diplomarbeit sehr viele Ressourcen binden.
Kosten			Kosten sind noch im Soll-Bereich
Tätigkeiten vergangene Berichtsperiode		Tätigkeiten nächste Berichtsperiode	
<ul style="list-style-type: none"> <li>- Anforderungskatalog erstellt</li> <li>- Parallel dokumentiert</li> </ul>		<ul style="list-style-type: none"> <li>- Analyse der PostgreSQL HA Clusterlösungen abgeschlossen</li> <li>- Benchmarking abgeschlossen</li> <li>- Variantenentscheid getroffen</li> <li>- Basisystem für Testsystem aufgebaut</li> </ul>	
# nächste Lieferobjekte (inkl. allfällige Links)		Status	Erliebigungsgrad Soll Datum
LO-002 Vorbereiten Benchmarking		offen	
LO-003 Analyse PostgreSQL HA Cluster Lösungen		in Arbeit	
LO-004 Gegenüberstellung		offen	
LO-005 Variantenentscheid		offen	
LO-006 Aufbau Basisinfrastruktur Testsystem		offen	
# Risiken	Auswirkungsgrad	Massnahmen	Verantw.
R-001 Fehlende Ressourcen		Organisation und Selbstmanagement	
R-002 HP-UX Ablöseprojekt		Ressourcen reservieren	
R-003 Alte Infrastruktur kann ungeplant sämtliche Ressourcen binden		Monitoring vorgängig ausbauen und Massnahmen definieren	
R-004 Schwächen beim Selbstmanagement und in der Selbstorganisation		Werkzeuge im Vorfeld definieren und bereitstellen	
R-005 Scope Verlust während des Projekts		Ziele klar definieren	
R-006 Scope Creep		Ziele SMART definieren	
R-007 SIEM / Log Plattform nicht betriebsbereit		Massnahmen ergreifen um die manuelle Installation so effizient wie möglich zu gestalten	
R-008 Foreman nicht betriebsbereit		Massnahmen ergreifen um die manuelle Installation so effizient wie möglich zu gestalten	
Kostenübersicht		Abhängigkeiten zu anderen Projekten	Massnahmen
Verfügbare Finanzen bis Ende Projekt: $100 \text{ CHF h}^{-1} * 200 = 24\,000 \text{ CHF}$		Erneuerung HP UX Plattform 6000201 KSGR Provisioning System (KPS) -> Foreman Umgebung	Ressourcen reservieren Massnahmen ergreifen um die manuelle Installation so effizient wie möglich zu gestalten
Bemerkungen / Informationen		Anträge	
Eingereicht	Geprüft	Bemerkungen/Auftrag PMO	
PL:	PMO:		
Datum:	Datum:		
# erledigte Lieferobjekte (inkl. allfällige Links)			
LO-001 Anforderungskatalog			

Tabelle 1.15: Zweiter Statusbericht

## 1.9 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 1.16: Fachgespräche

Das Protokoll ist im Anhang zu finden.

## 2 Umsetzung

### 2.1 Evaluation

#### 2.1.1 Exkurs Architektur

##### 2.1.1.1 ACID

###### **Atomarität - Atomarität**

Besagt, dass jede Transaktion als separate Einheit behandelt wird.

Entweder die gesamte Transaktion wird ausgeführt und committed oder kein Teil von ihr.

###### **Consistency - Konsistenz**

Definiert, dass eine Transaktion einen gültigen Zustand erzeugt oder der alte Zustand wiederhergestellt wird (rollback).

###### **Isolation - Isolation**

Beschreibt, dass jede Transaktion voneinander isoliert ist und sich weder sehen noch gegenseitig beeinflussen können.

###### **Durability - Dauerhaftigkeit**

Sagt aus, dass jede Änderung die committed wurde, auch bei einem Systemausfall oder Defekt beständig sein muss.

Daten dürfen zudem nur mittels Transaktionen verändert werden und nicht von aussen.

#### 2.1.1.2 Sharding

##### 2.1.1.2.1 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 2.1: Sharding - Vertikale Partitionierung

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 2.2: Sharding - Horizontales Partitionierung

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

#### 2.1.1.2.2 Key Based Sharding

Hierbei wird das sharding anhand eines oder mehreren Keys ausgeführt.

#### 2.1.1.2.3 Range Based Sharding

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

#### 2.1.1.2.4 Directory Based Sharding

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

#### 2.1.1.2.5 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.

### 2.1.1.3 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

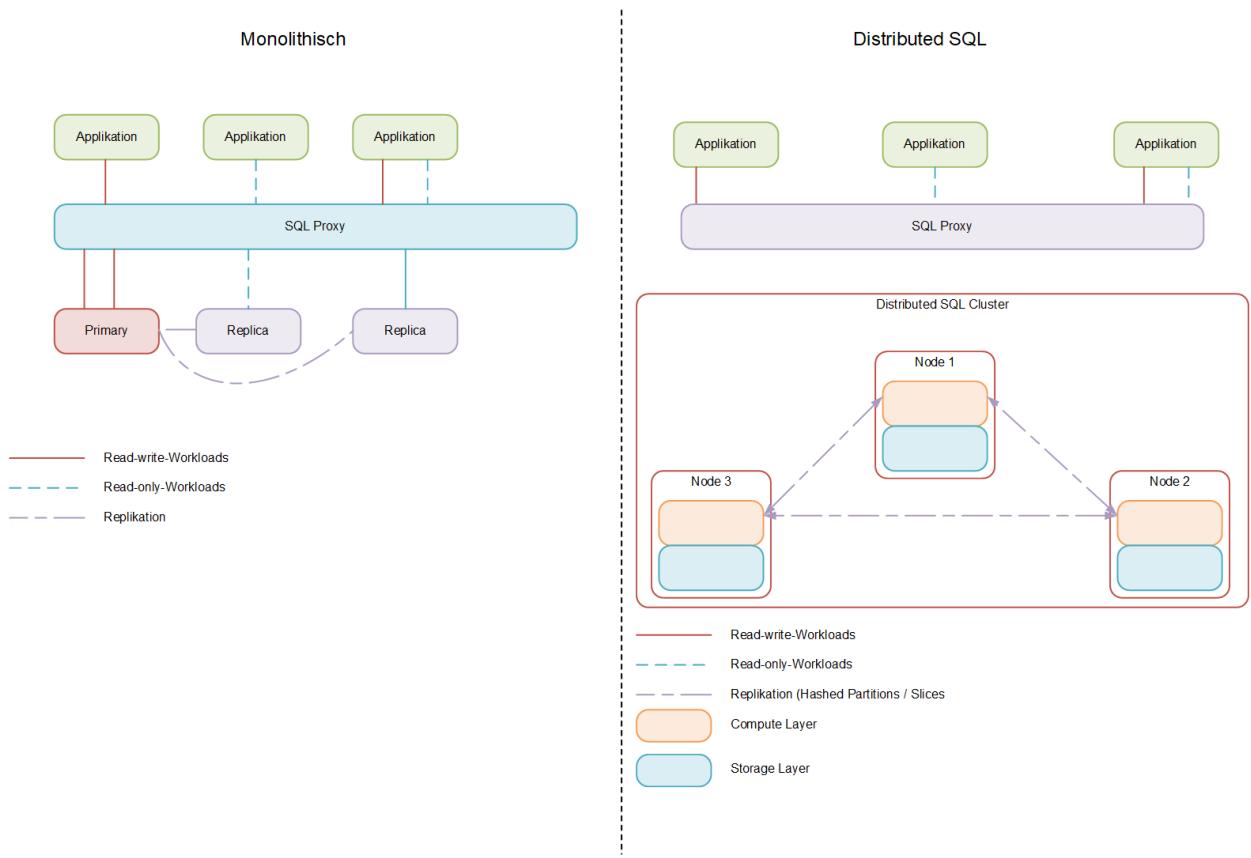


Abbildung 2.3: Monolithische vs. verteilte SQL Systeme

#### 2.1.1.4 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[51]. Dabei wird zwischen den folgenden beiden Replikationen unterschieden:

##### Synchrone Replikation

Wenn bei einer Synchronen 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.

#### 2.1.1.5 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 [49, 61, 44] 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 2.1: Quorum Beispiele

#### 2.1.1.6 CAP Theorem

Das CAP Theorem besagt, dass nur zwei der drei folgenden drei Merkmale von verteilten Systemen gewährleistet werden können[34].

##### **Konsistenz - Consistency**

Die Datenbank ist konsistent, alle Clients sehen gleichzeitig die gleichen Daten unabhängig davon auf 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 2.4: CAP-Theorem

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

#### 2.1.1.7 Skalierung

Datenbanken müssen skalierbar sein. Dabei wird unterschieden zwischen einer vertikalen Skalierung (scale-up) und horizontaler Skalierung (scale-out). Bei der vertikalen Skalierung 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[46]:

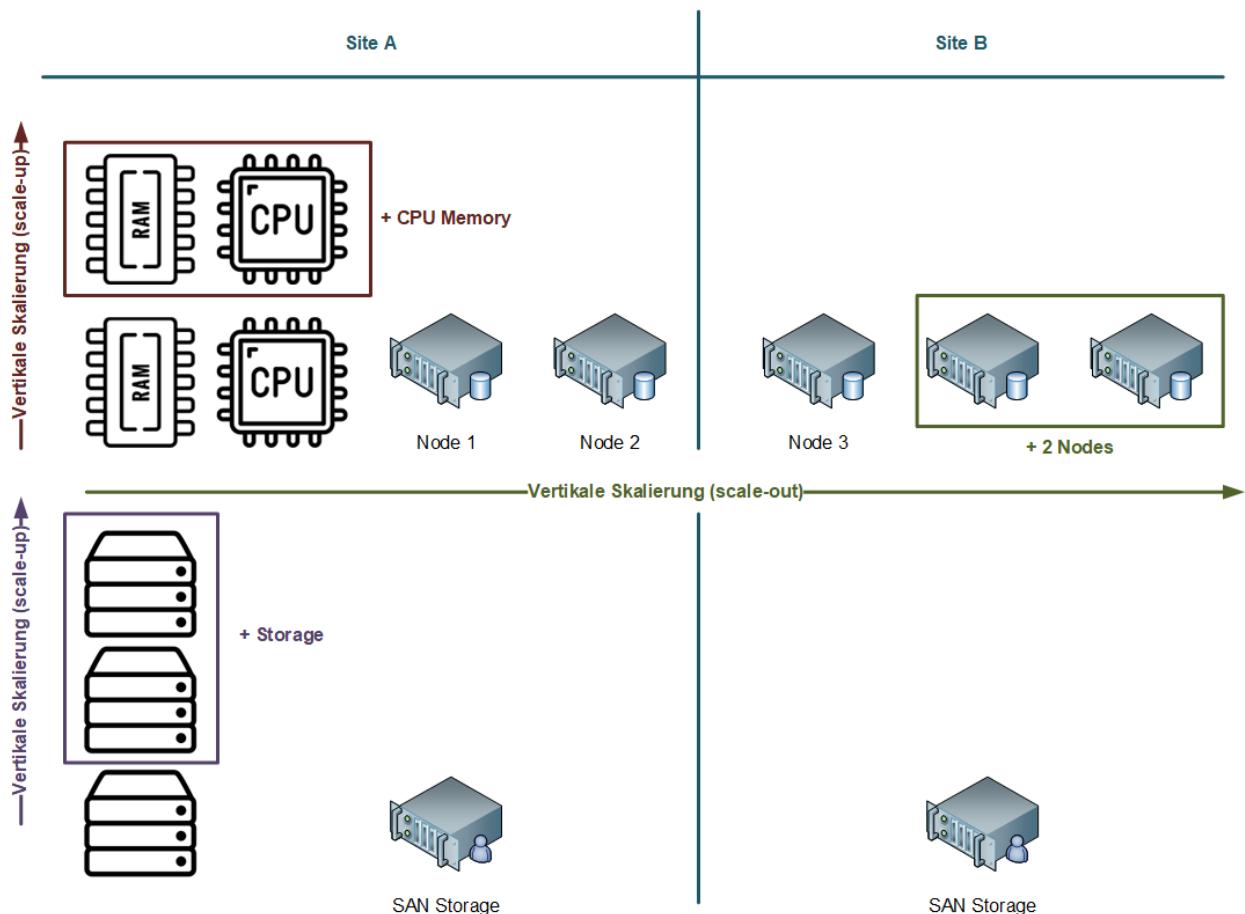


Abbildung 2.5: Datenbankskalierung

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

### 2.1.2 Erheben und Gewichten der Anforderungen

#### 2.1.2.1 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 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 Replicas 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
21	Management-API	Management-API vorhanden	Das Clustersystem muss Skripte oder eine API liefern, mit dem das System zu konfigurieren, verwalten oder überwachen zu können. Zudem müssen mit geringen Arbeitsaufwand	Beides	MUSS
22	Management-API	Authentifizierung & Autorisierung	damit Nodes hinzugefügt oder entfernt werden können	Beides	MUSS
23	Management-API	Aufwand	Es müssen gängige Standards für Authentifizierung und Autorisierung mitgebracht werden Der Aufwand,	Beides	MUSS
24	Backup	Backup mit PostgreSQL Standards	der benötigt wird um die DB zu verwalten, Nodes hinzuzufügen oder zu entfernen usw. muss gegeneinander verglichen werden.	Beides	MUSS
25	Backup	Restore mit PostgreSQL Standards	Backups müssen mittels PostgreSQL Standards angezogen werden	Beides	MUSS
26	Housekeeping - Log Rotation		Backups müssen mittels PostgreSQL Standards restored werden können	Beides	MUSS
27	Self Healing		Das Clustersystem muss die möglichkeit zur Log Rotation bieten	Beides	KANN
28	Monitoring - Node Failure		Das Clustersystem muss im Fehlerfall Nodes selber wiederherstellen können Läuft ein Node auf einen Fehler,	Beides	MUSS
29	Maintenance Quality		muss das Clustersystem dies erkennen und Melden resp. eine Schnittstelle liefern die abgefragt werden kann	Beides	MUSS
30	Performance	tps - Read-Only	Da die meisten PostgreSQL HA Lösungen Open-Source sind, muss sichergestellt werden,	Beides	MUSS
31	Performance	tps - Read-Writes	die gewählte Lösung auch aktiv gepflegt wird.	Beides	MUSS
32	Performance	Ø Latenz - Read-Only	Als Basis dienen hier Informationen wie z.B. GitHub Insights.	Beides	MUSS
33	Performance	Ø Latenz - Read-Write	Die Transaktionsrate (transactions per second / tps) für DQL Transaktionen	Beides	MUSS
			Die Transaktionsrate (transactions per second / tps) für DML Transaktionen	Beides	MUSS
			Die Latenzzeit bei DQL Transaktionen	Beides	MUSS
			Die Latenzzeit bei DML Transaktionen	Beides	MUSS

Tabelle 2.2: Anforderungskatalog

## 2.1.2.2 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 2.3: Stakeholder

## 2.1.2.3 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:

Gewicht	Nennungen	Rang	Nr.	Ziele	1	Systemvielfalt	2	Synergien	3	Failover	4	Switchover	5	Restore	6	Replikation	7	Sharding	8	Quorum	9	Connection	10	Management-API	11	Backup	12	Housekeeping - Log Rotation	13	Self Healing	14	Monitoring - Node Failure	15	Maintenance Quality
13	15	1	1	Systemvielfalt																														
12	14	2	2	Synergien	1																													
11	13	3	3	Failover	1	2																												
10	12	4	4	Switchover	1	2	3																											
9	11	5	5	Restore	1	2	3	4																										
8	10	6	6	Replikation	1	2	3	4	5																									
3	3	13	7	Sharding	1	2	3	4	5	6																								
7	8	7	8	Quorum	1	2	3	4	5	6	8																							
6	7	8	9	Connection	1	2	3	4	5	6	9	8																						
3	4	12	10	Management-API	1	2	3	4	5	6	10	8	9																					
6	7	8	11	Backup	1	2	3	4	5	6	11	8	9	11																				
1	1	14	12	Housekeeping - Log Rotation	1	2	3	4	5	6	7	8	9	10	11																			
1	1	14	13	Self Healing	1	2	3	4	5	6	7	8	9	10	11	13																		
5	6	11	14	Monitoring - Node Failure	1	2	3	4	5	6	14	8	9	14	11	14	14																	
1	1	14	15	Maintenance Quality	1	2	3	4	5	6	7	8	9	10	11	12	15	14																
6	7	8	16	Performance	1	2	3	4	5	6	16	16	16	16	11	16	16	14	16															
<b>100</b>	<b>120</b>																																	

#### Legende

Eingabefelder

Zellbezüge

berechnete Felder

Abbildung 2.6: Präferenzmatrix

Die Gewichtung der Failover-Anforderungen setzt sich wie folgt zusammen:

Gewicht	Nennungen	Rang	Nr.	Ziele		
5	3	1	1	Automatismus		
4	2	2	2	Connection-Stabilität	1	
2	1	3	3	Geschwindigkeit	1	2
<b>11</b>	<b>6</b>					

**Legende**

Eingabefelder

Zellbezüge

berechnete Felder

Abbildung 2.7: Präferenzmatrix - Failover

Beim Switchover wurde die Gewichtung wie folgt aufgeteilt:

Gewicht	Nennungen	Rang	Nr.	Ziele			Skript / API	1	2
5	3	1	1	Skript / API					
3	2	2	2	Connection - Stabilität			1		
2	1	3	3	Geschwindigkeit			1	2	
<b>10</b>	<b>6</b>								

**Legende**

Eingabefelder

Zellbezüge

berechnete Felder

Abbildung 2.8: Präferenzmatrix - Switchover

Die Gewichtung und Aufteilung der Restore-Anforderungen sieht wie folgt aus:

Gewicht	Nennungen	Rang	Nr.	Ziele		1	2	3
					Skript / API	Datensicherheit	Connection - Stabilität	
5	3	1	1	Skript / API				
2	1	2	2	Datensicherheit	1			
2	1	2	3	Connection - Stabilität	1	2		
2	1	2	4	Geschwindigkeit	1	4	3	
9	6							

**Legende**

Eingabefelder

Zellbezüge

berechnete Felder

Abbildung 2.9: Präferenzmatrix - Restore

Die Replikationsanforderungen resp. deren Gewichtung ist wie folgt aufgebaut:

Gewicht	Nennungen	Rang	Nr.	Ziele		
4	3	1	1	Synchrone Replikation		
3	2	2	2	Failover / Switchover Garantie	1	
1	1	3	3	Throughput	1	2
<b>8</b>	<b>6</b>					

**Legende**

Eingabefelder

Zellbezüge

berechnete Felder

Abbildung 2.10: Präferenzmatrix - Replikation

Das Sharding setzt sich aus folgenden Teilen zusammen:

Gewicht	Nennungen	Rang	Nr.	Ziele	
2	2	1	1	Datenkonsistenz- und Integrität	
1	1	2	2	Schutz vor Datenverlust	1
3	3				

**Legende** Eingabefelder Zellbezüge berechnete Felder

Abbildung 2.11: Präferenzmatrix - Sharding

Die Quorum-Anforderung ist folgendermassen zusammengesetzt:

Gewicht	Nennungen	Rang	Nr.	Ziele	Quorum-System vorhanden
4	2	1	1	Quorum -System vorhanden	
2	1	2	2	Robustheit	1
7	3				

**Legende** Eingabefelder Zellbezüge berechnete Felder

Abbildung 2.12: Präferenzmatrix - Quorum

Bei der Management-API gibt es mehrere Sub-Anforderungen:

Gewicht	Nennungen	Rang	Nr.	Ziele	Management-API vorhanden	Authentifizierung & Autorisierung
1	2	1	1	Management-API vorhanden		
1	2	1	2	Authentifizierung & Autorisierung	1	
1	2	1	3	Aufwand	3	2
<b>3</b>	<b>6</b>					

**Legende**

Eingabefelder

Zellbezüge

berechnete Felder

Abbildung 2.13: Präferenzmatrix - Management-API

Anforderungen zum Backup wurden nachfolgend aufgeteilt und gewichtet:

Gewicht	Nennungen	Rang	Nr.	Ziele	
4	2	1	1	Backup mit PostgreSQL Standards	
2	1	2	2	Restore mit PostgreSQL Standards	1
6	3				

**Legende**

-  Eingabefelder
-  Zellbezüge
-  berechnete Felder

Abbildung 2.14: Präferenzmatrix - Backup

Performance-Benchmarking lässt sich in nachfolgende Teile unterteilen:

Gewicht	Nennungen	Rang	Nr.	Ziele	1	2	3
2	4	1	1	tps - Read-Only			
2	3	2	2	tps - Read-Writes	1		
1	2	3	3	Ø Latenz - Read-Only	1	2	
1	1	4	4	Ø Latenz - Read-Write	1	2	3
<b>6</b>	<b>10</b>						

### Legende

Eingabefelder

Zellbezüge

berechnete Felder

Abbildung 2.15: Präferenzmatrix - Performance

2.1.3 Testziele erarbeiten

2.1.4 PostgreSQL Benchmarking

2.1.4.1 pgBench - Basis-Benchmarking

PostgreSQL bietet ein Benchmarking-Tool,[42, 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 [20, 17, 76, 1] sind dies folgende Anforderungen:

### **pgGather**

Mit pgGather [48] 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 [72] usw., müssen ebenfalls bemessen werden.

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

#### **2.1.4.2 Replication Throughput Benchmarking**

Etwas Komplexer wird es beim Benchmark des Throughput. Den nebst den DB-Latenzen usw. kommt nun noch die Netzwerklatenz usw. hinzu [60].

#### **2.1.4.3 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:

# Diplomarbeit



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 2.16: Benchmark Settings - Zabbix - Systeminformationen

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:

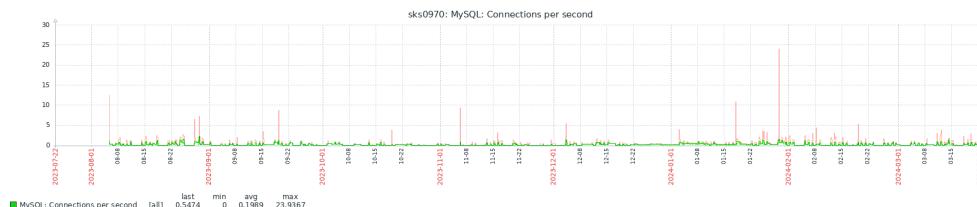


Abbildung 2.17: Benchmark Settings - Zabbix - Connections per Seconds

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

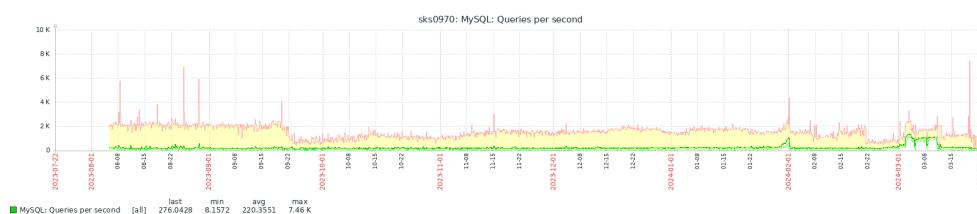


Abbildung 2.18: Benchmark Settings - Zabbix - Queries per Seconds

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

# Diplomarbeit

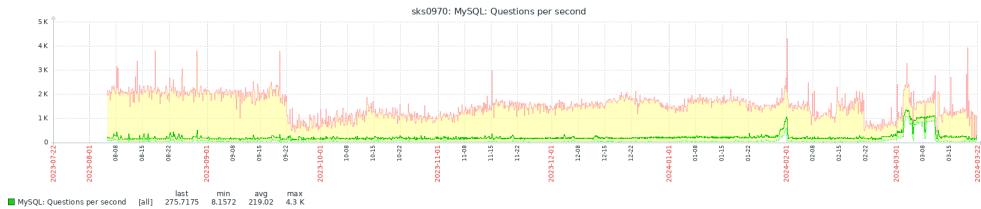


Abbildung 2.19: Benchmark Settings - Zabbix - Client Queries per Seconds

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

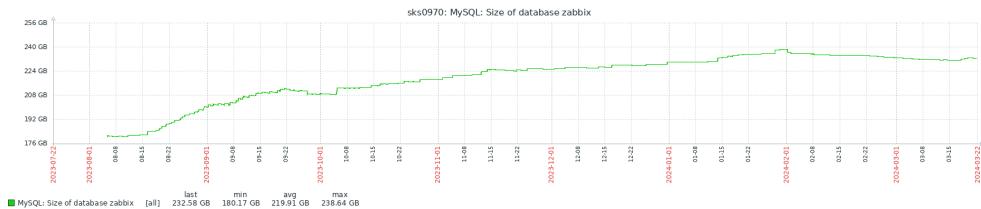


Abbildung 2.20: Benchmark Settings - Zabbix - DB Size

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 2.4: 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 2.5: Benchmark Settings - DQL Transaktionen

## 2.1.5 Analyse gängiger PostgreSQL HA Cluster Lösungen

### 2.1.5.1 PostgreSQL Replikation

PostgreSQL bietet von Haus aus Möglichkeiten, um Replikationen durchzuführen. Dabei ist nicht jede gleich gut für jedes Szenario geeignet[41].

#### Shared Disk Failover

#### File System (Block Device) Replication

#### Write-Ahead Log Shipping

#### Logical Replication

#### Trigger-Based Primary-Standby Replication

#### Data Partitioning

#### Multiple-Server Parallel Query Execution

## 2.1.5.2 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[62]. 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.

## 2.1.5.3 pgpool-II

pgpool-II ist eine Middleware die zwischen einem PostgreSQL Cluster und einem PostgreSQL Client gesetzt wird. pgpool-II bietet folgende Funktionen[58, 39]:

### **High Availability**

pgpool-II bietet einen automatic Failover genannten Service an, den Watchdog. Dieser schwenkt auf einen Standby-Server und entfernt den Defekten Server. Um false positive Events und Split-brains zu verhindern setzt pgpool-II auf einen eigens entwickelten Quorum-Algorithmus.

### **Connection Pooling**

Bestehende Connections werden wiederverwendet um die Anzahl gleichzeitig offener Connections zu reduzieren. Der Pool wird dabei anhand von Username, Database, Protocol und weiteren Verbindungsparametern zugeordnet.

### **Replikation**

Nebst dem Standard PostgreSQL bietet pgpool-II sein eigenes Replikationssystem an.

### **Load Balancing**

Ähnlich wie Oracle Active Data Guard [25] bietet auch pgpool-II die Möglichkeit, SELECT-Queries und Backup-Jobs auf die Secondary-Nodes umzuleiten um den Primary Node zu entlasten.

### **Limiting Exceeding Connections**

Die Anzahl an concurrent Connections, also gleichzeitiger Verbindungen, ist bei PostgreSQL begrenzt (Systemparameter wird dabei vom DBA gesetzt). pgpool-II speichert alle Connections, die über dem Limit sind, in einer Queue und somit nicht sofort fehlerhaft abgelehnt.

### **Watchdog**

Der Watchdog koordiniert mehrere pgpool-II Nodes und verhindert ein Split-brain.

### In Memory Query Caching

pgpool-II speichert SELECT-Queries in einem Cache und verwendet die ResultSets wieder, wenn eine identische Abfrage eingeht.

### Online Recovery

pgpool-II bietet die Möglichkeit, einen Online Recovery resp. eine Online Synchronisation eines Nodes durchzuführen, auch kann ein neuer Standby-Node synchronisiert werden. Dafür muss der Node aber im Detached Mode stehen, unabhängig ob der Detach manuell oder von pgpool-II ausgeführt wurde.

#### 2.1.5.4 pg\_auto\_failover

##### 2.1.5.4.1 Core-Features

##### 2.1.5.4.2 Replikation

##### 2.1.5.4.3 Proxy

pg\_auto\_failover benötigt einen HAProxy, um Load Balancing usw. [13]

##### 2.1.5.4.4 API / Skripte

##### 2.1.5.4.5 Architektur

Die Dokumentation von pg\_auto\_failover [5] zeigt auf, wie der Failover funktioniert:

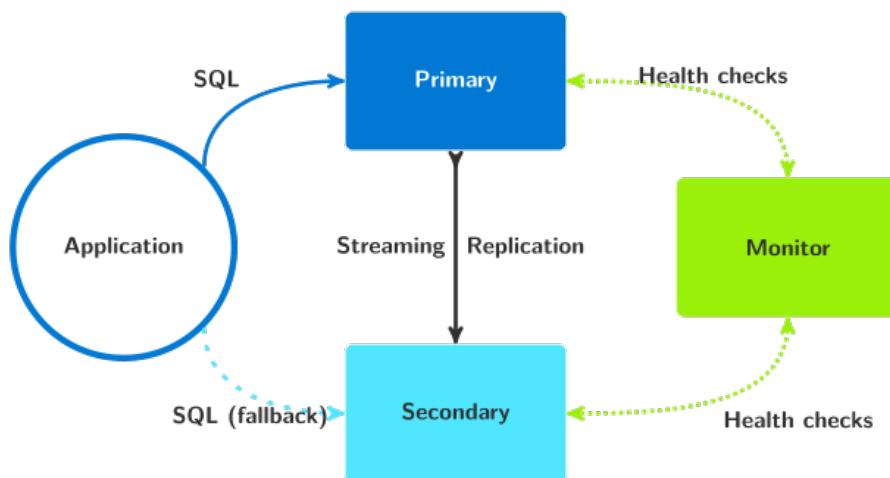


Abbildung 2.21: pg\_auto\_failover-Architektur - Single Standby

Aber auch Multi-Nodes können eingebunden werden[15]:

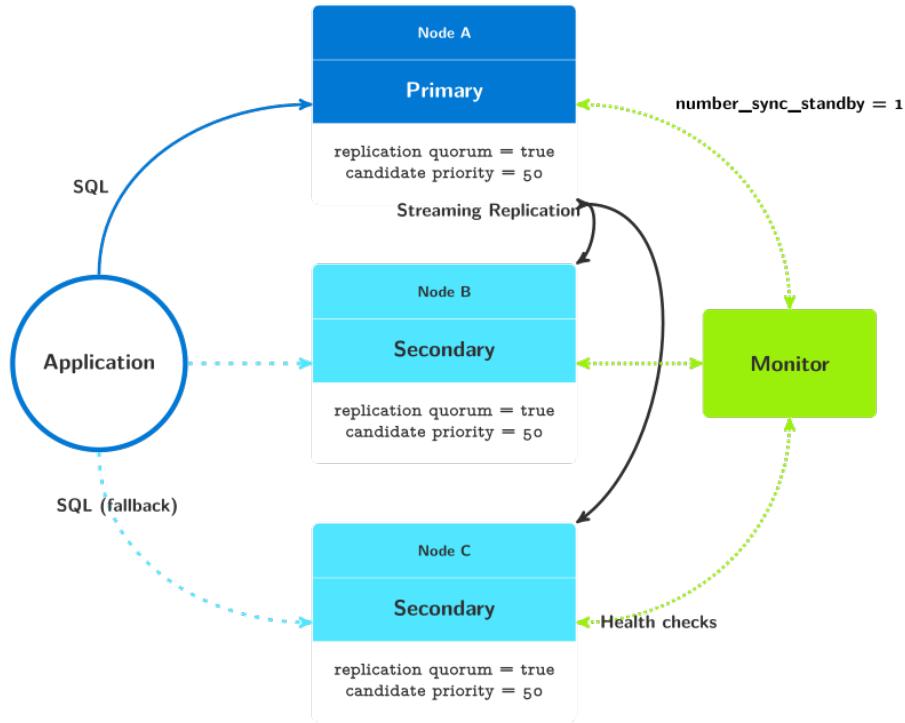


Abbildung 2.22: pg\_auto\_failover-Architektur - Multi-Node Standby

pg\_auto\_failover kann Citus einbinden[7]. Allerdings bleibt die Architektur im Kern immer Monolithisch.

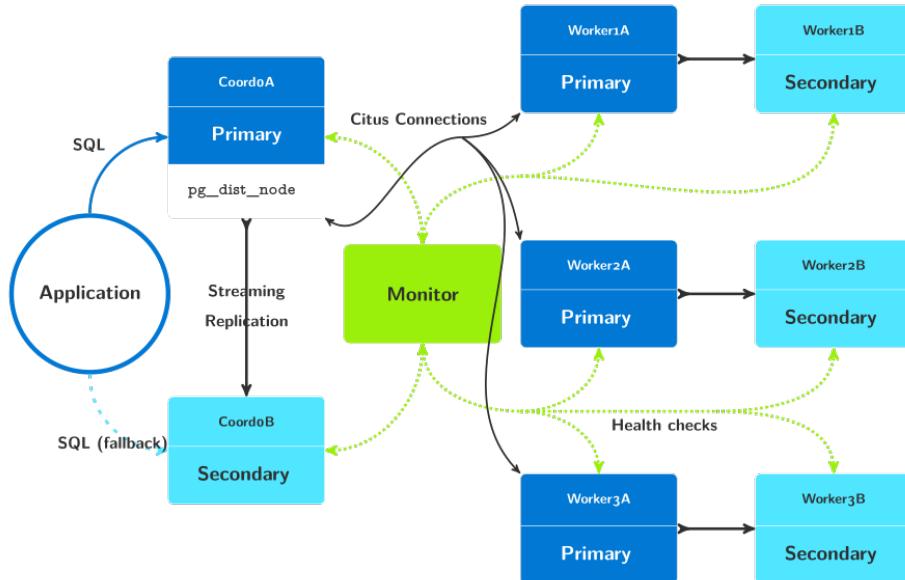


Abbildung 2.23: pg\_auto\_failover-Architektur - Citus

#### 2.1.5.4.6 Synergien und Mehrwert

#### 2.1.5.5 Patroni

Patroni ist eine von Zalando entwickelte

##### 2.1.5.5.1 Core-Features

##### 2.1.5.5.2 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.

##### 2.1.5.5.3 Proxy

Patroni benötigt einen HAProxy, um Load Balancing usw. [13]

##### 2.1.5.5.4 Pooling

##### 2.1.5.5.5 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.

##### 2.1.5.5.6 etcd

Patroni benötigt etcd als key-value-store

##### 2.1.5.5.7 Architektur

Das Architektur-Schaubild sieht folgendermassen aus:

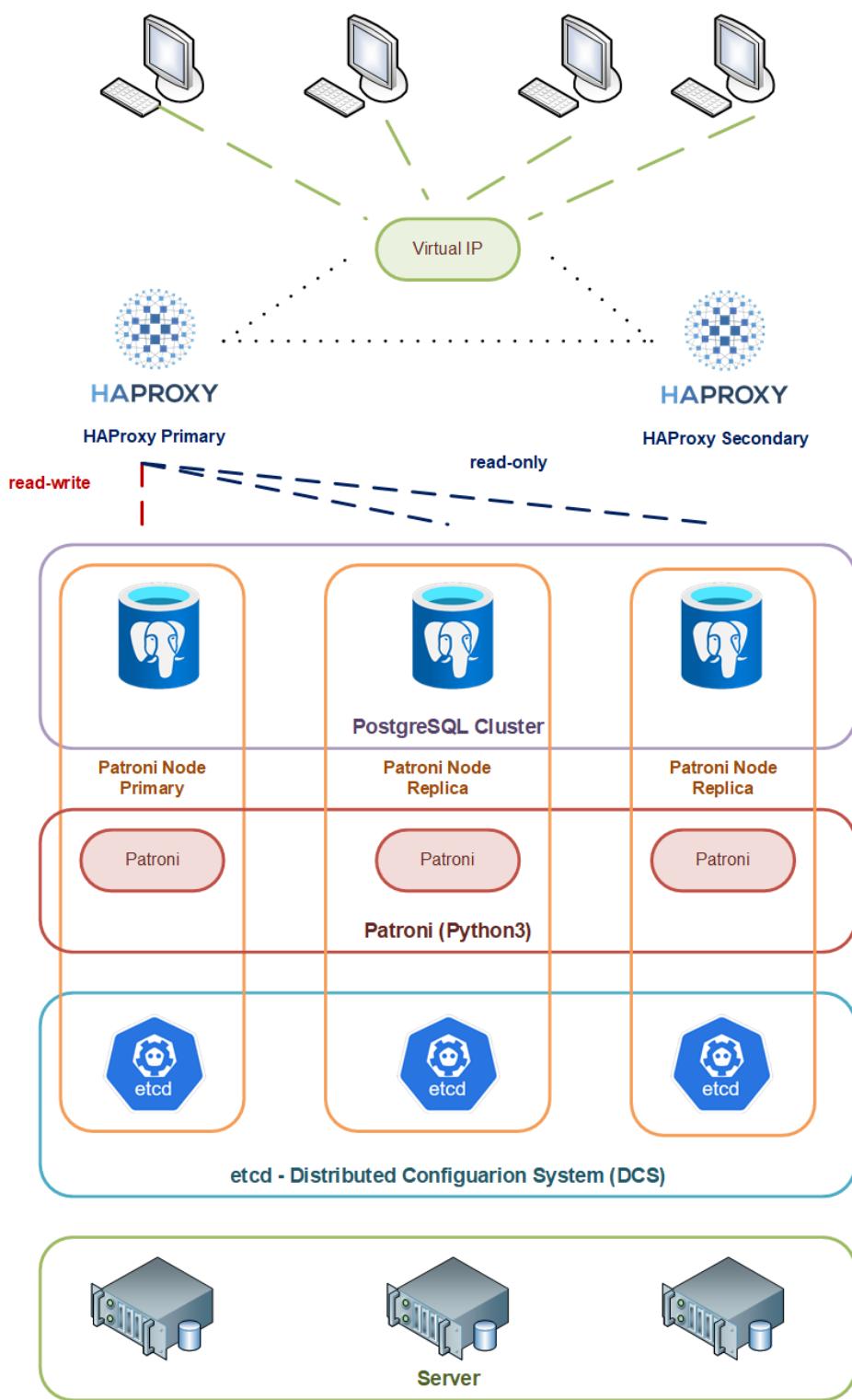


Abbildung 2.24: Patroni-Architektur

## 2.1.5.5.8 Maintenance

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

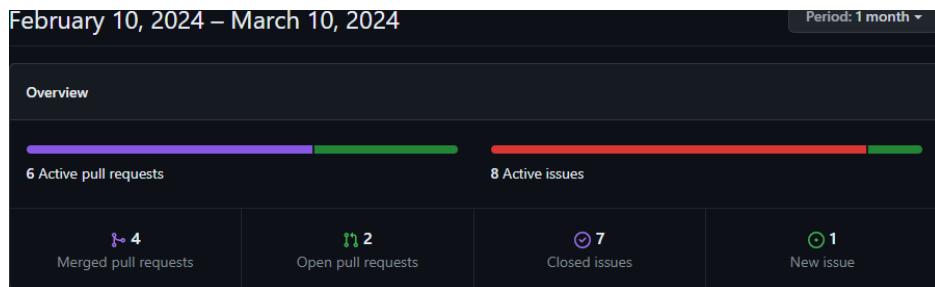


Abbildung 2.25: Patroni - Pulse

Code wird Regelmässig hinzugefügt und entfernt:

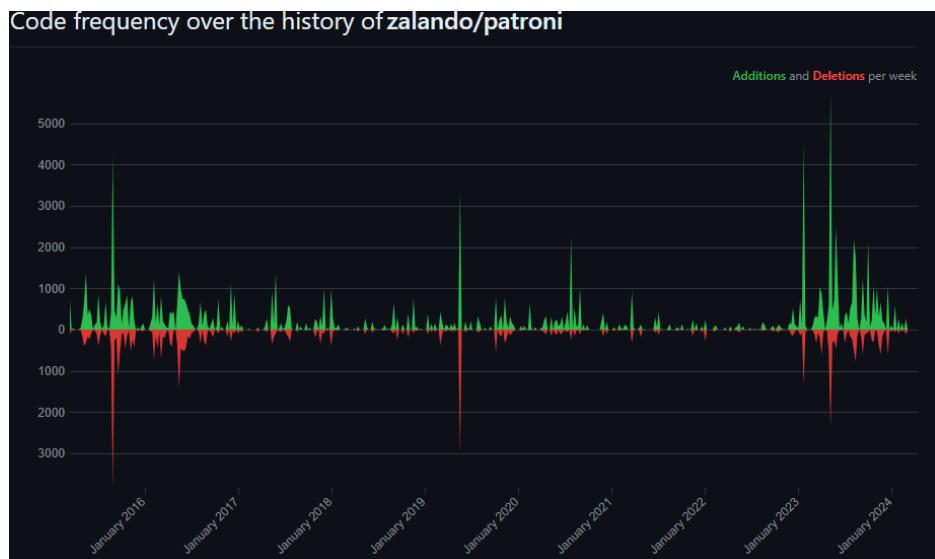


Abbildung 2.26: Patroni - Code Frequency

Das Projekt hält auch die gängigen Standards auf Github ein:

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

[Set up a security policy](#) [Propose](#)

Abbildung 2.27: Patroni - Community Standards

Die Contributors commiten, löschen und erweitern Patroni Regelmässig:

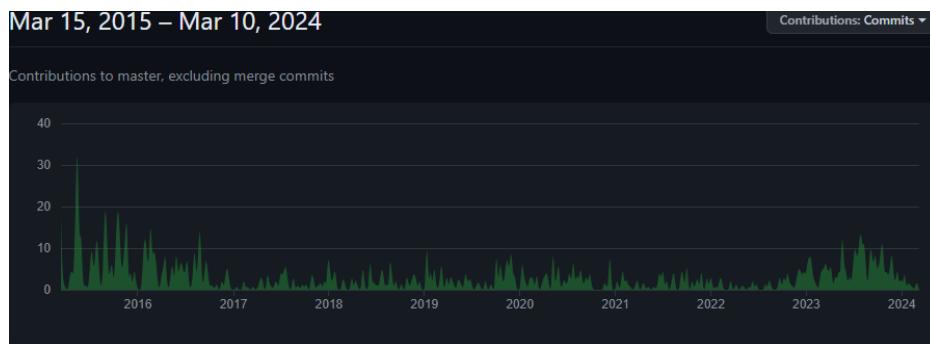


Abbildung 2.28: Patroni - Contributors Commits

# Diplomarbeit



Abbildung 2.29: Patroni - Contributors Deletations



Abbildung 2.30: Patroni - Contributors Additions

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

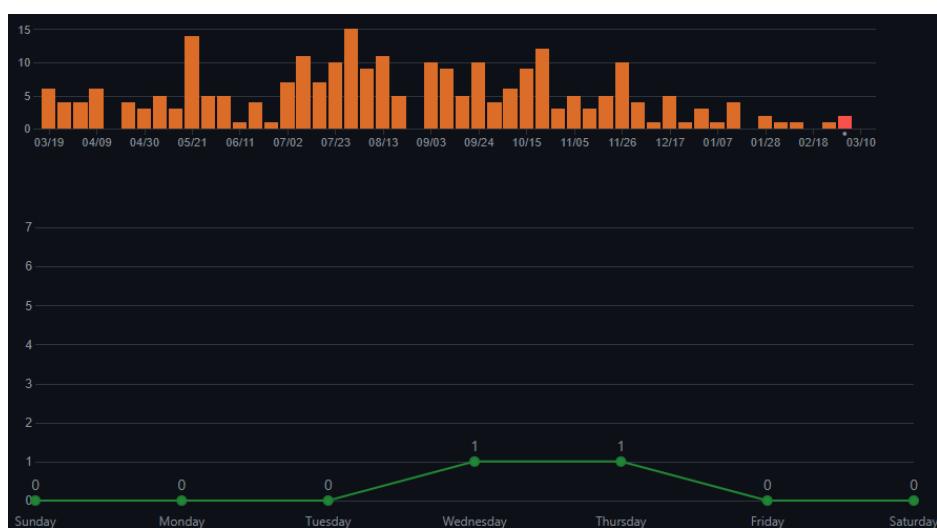


Abbildung 2.31: Patroni - Commit Activity

Nebst Zalando selbst, hat auch EnterpriseDB[27] ein grösseres Repository eingebunden. Dies weil EnterpriseDB stark auf Patroni setzt.



Abbildung 2.32: Patroni - Network Graph

## 2.1.5.5.9 Synergien und Mehrwert

## 2.1.5.6 CloudNativePG

CloudNativePG ist eine Containerlösung für PostgreSQL auf Kubernetes.

## 2.1.5.6.1 Core-Features

Die wichtigsten Features von CloudNativePG sind[8]:

## k8s API integration

## Autoamtischer Failover

## Self-Healing von Nodes resp. Replikas

Skalierbarkeit (Vertikal, Horizontal bedingt)

## Volumne Backup

## Object Backup

Rolling Postgres

## Pooling mit PgBouncer

Offline und Online Imp

#### 2.1.5.6.2 Replikation

CloudNativePG bietet die üblichen PostgreSQL Replikationen an.

#### 2.1.5.6.3 Proxy

CloudNativePG benötigt keinen zusätzlichen Proxy.

#### 2.1.5.6.4 Pooling

CloudNativePG unterstützt pgBouncer als Pooler.

#### 2.1.5.6.5 API / Skripte

CloudNativePG bietet eine API zum Monitoren und Verwalten von Backups, Clustern und dem System selbst[2].

#### 2.1.5.6.6 Architektur

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

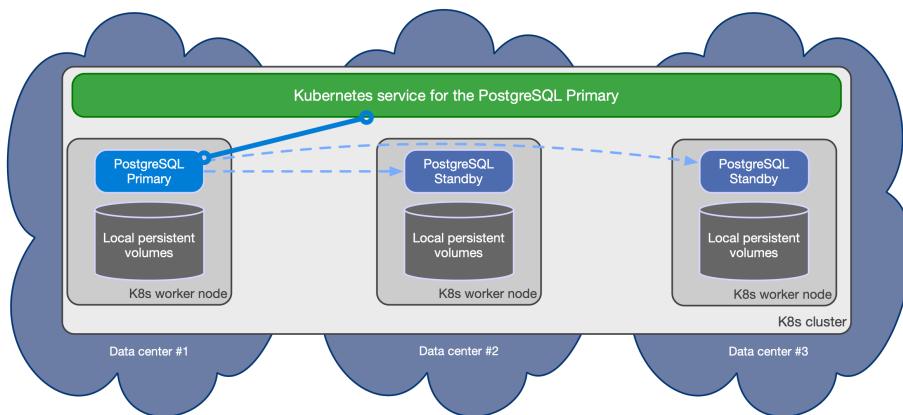


Abbildung 2.33: CloudNativePG - Kubernetes - PostgreSQL

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

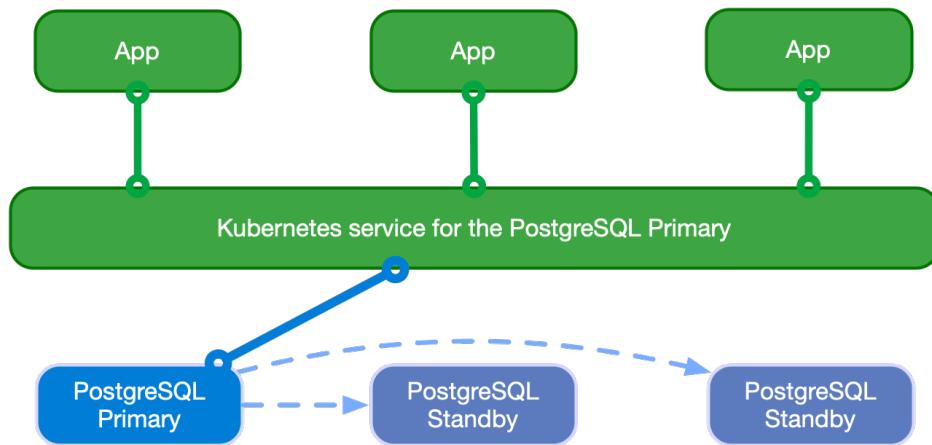


Abbildung 2.34: CloudNativePG - Kubernetes - Read-write workloads

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

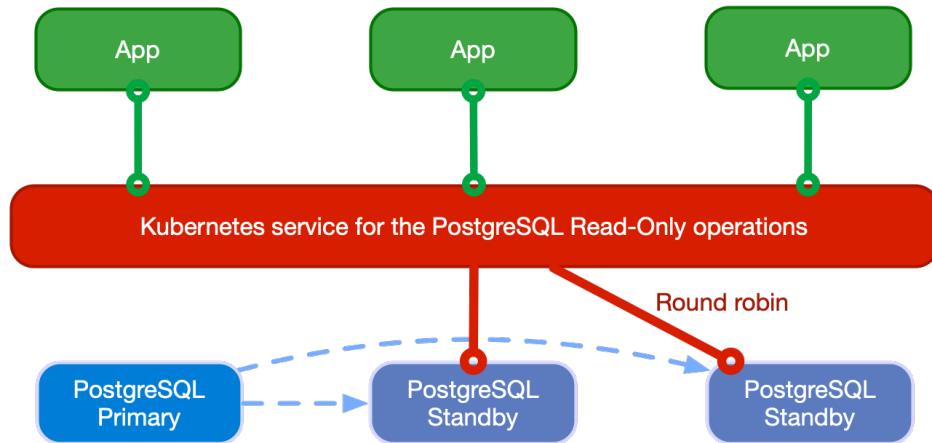


Abbildung 2.35: CloudNativePG - Kubernetes - Read-only workloads

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

#### 2.1.5.6.7 Maintenance

#### 2.1.5.6.8 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.

### 2.1.5.7 **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[73].

#### 2.1.5.7.1 **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:

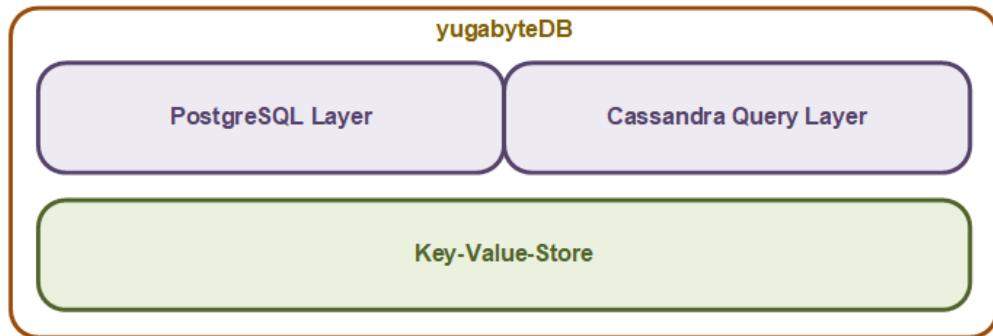


Abbildung 2.36: yugabyteDB - Grundkonzept

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

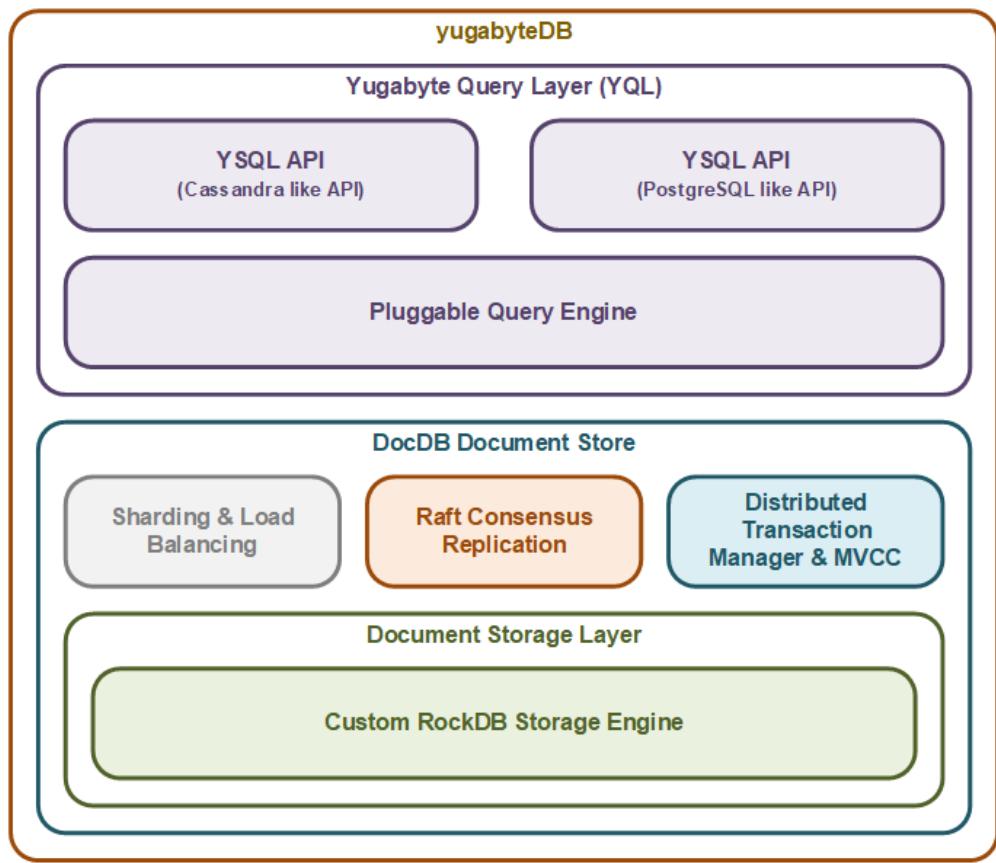


Abbildung 2.37: YugabyteDB - Architektur

#### 2.1.5.7.1.1 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 2.38: YugabyteDB - Sharding

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

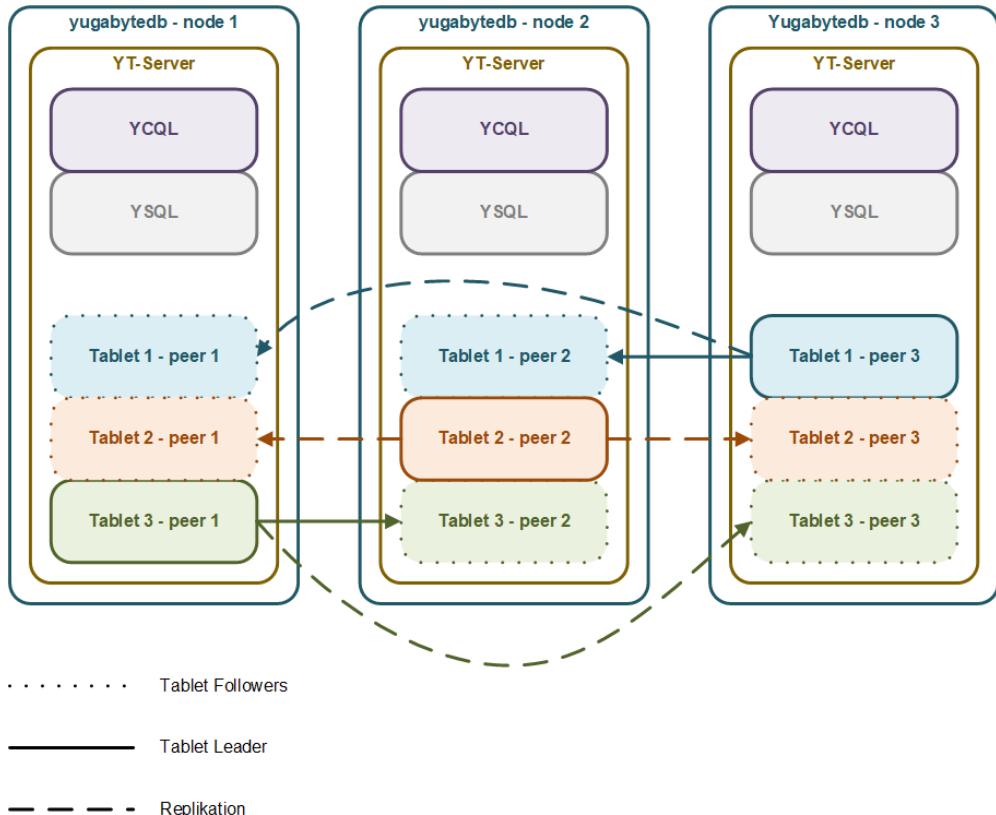


Abbildung 2.39: yugabyteDB - Tablet - Leader und Follower

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 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 [18] geregelt:

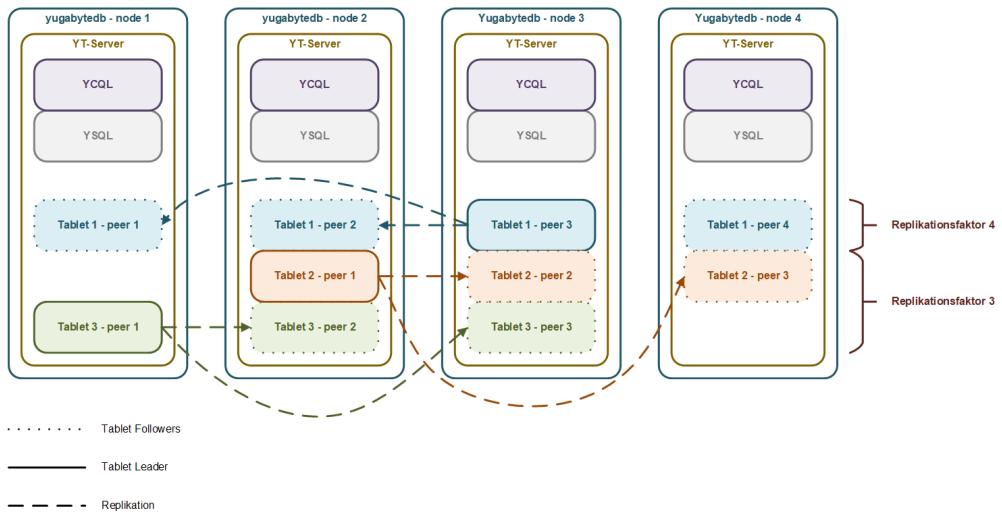


Abbildung 2.40: yugabyteDB - Tablet - Replikationsfaktor

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:

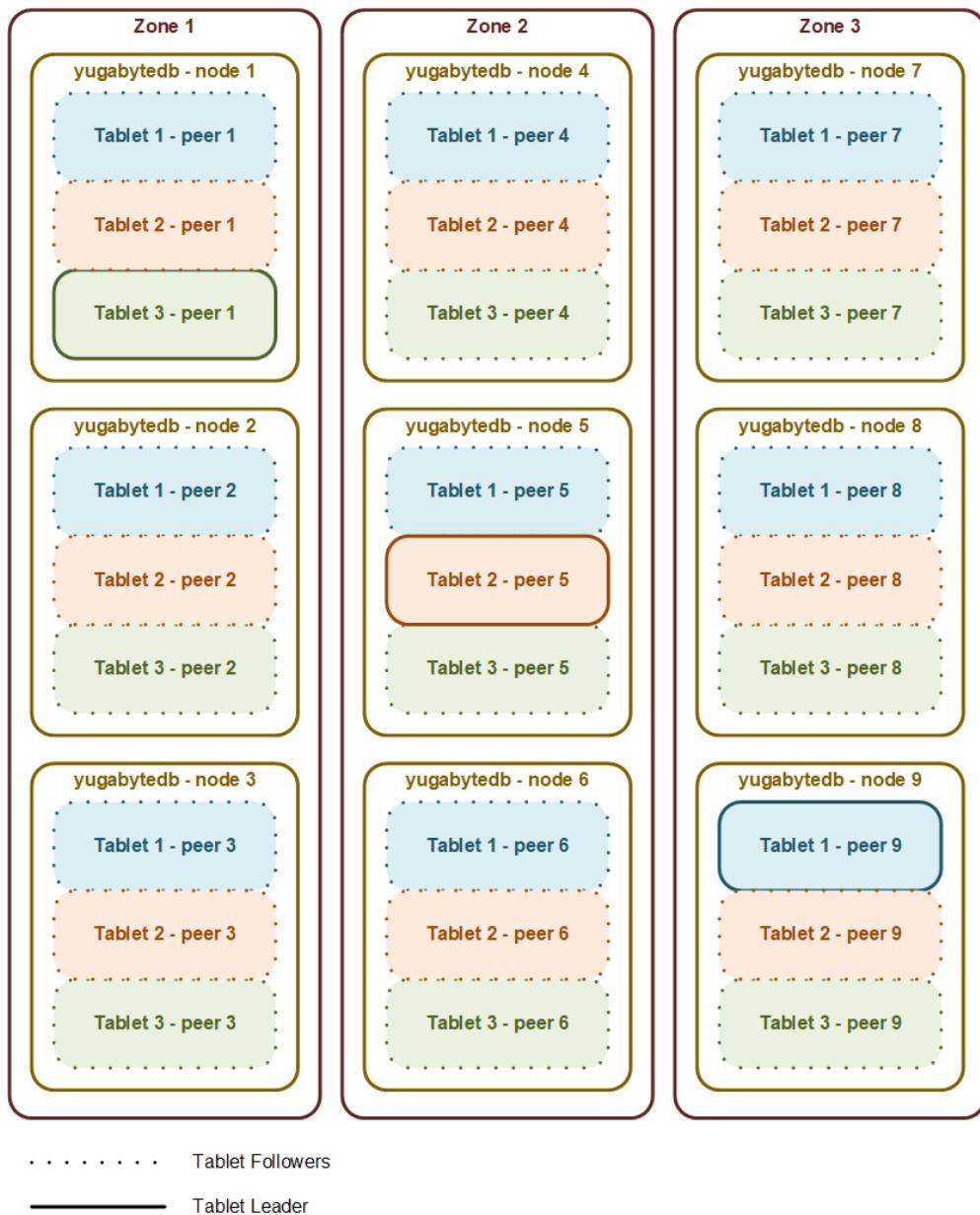


Abbildung 2.41: YugabyteDB - Zonen

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. YugabyteDB nennt dies Zone outage Tolerance[16].

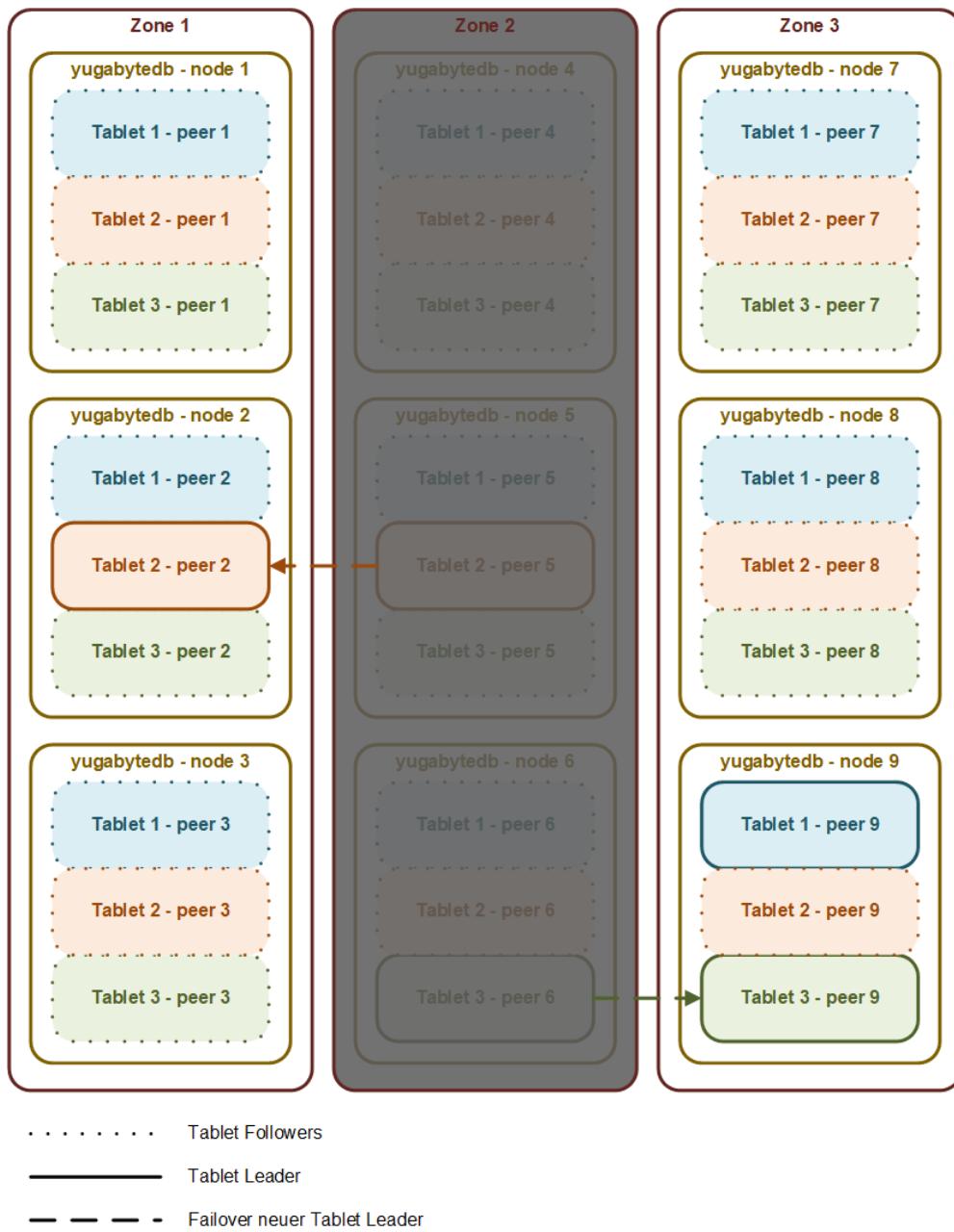


Abbildung 2.42: YugabyteDB - Zone outage Tolerance

### 2.1.5.8 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 wiederum ist in den

#### 2.1.5.8.1 Architektur

##### 2.1.5.8.1.1 Citus Coordinator und Workers

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

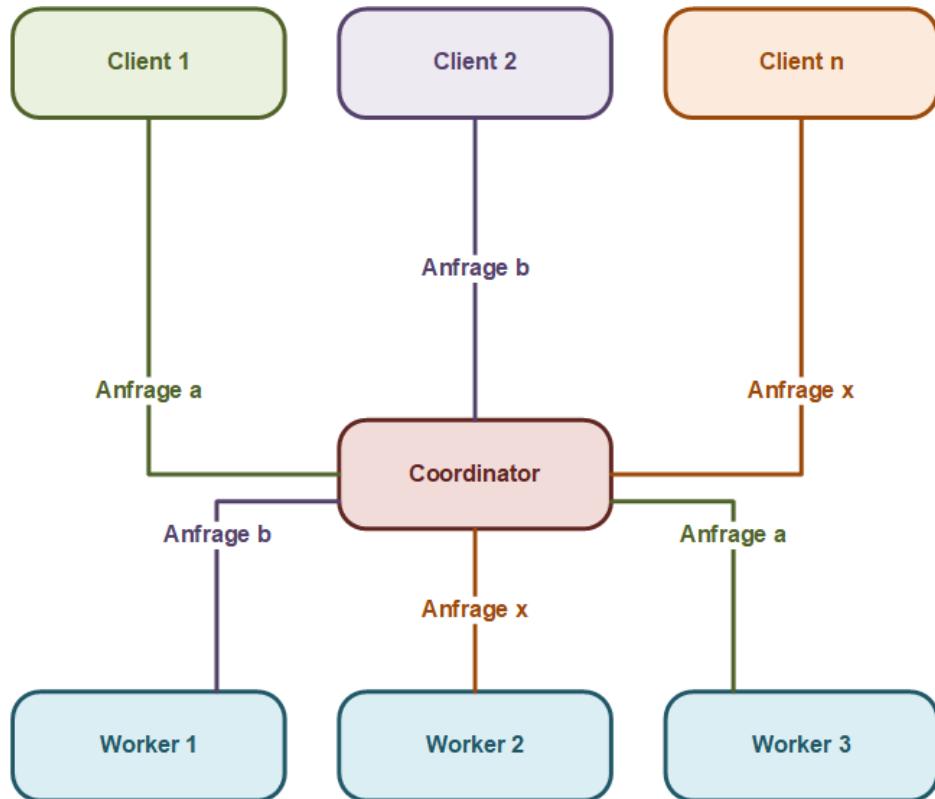


Abbildung 2.43: Citus - Coordinator und Workers

##### 2.1.5.8.1.2 Citus Sharding

Citus bietet zwei Sharding-Modelle an.

**Row-based sharding** Beim diesen sharding werden Tabellen anhand einer Distribution Column aufgeteilt. [10, 6]

# Diplomarbeit

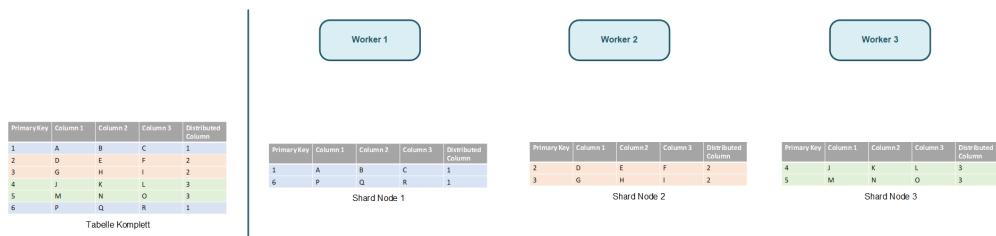


Abbildung 2.44: Citus - Row-Based-Sharding

## Schema-based sharding

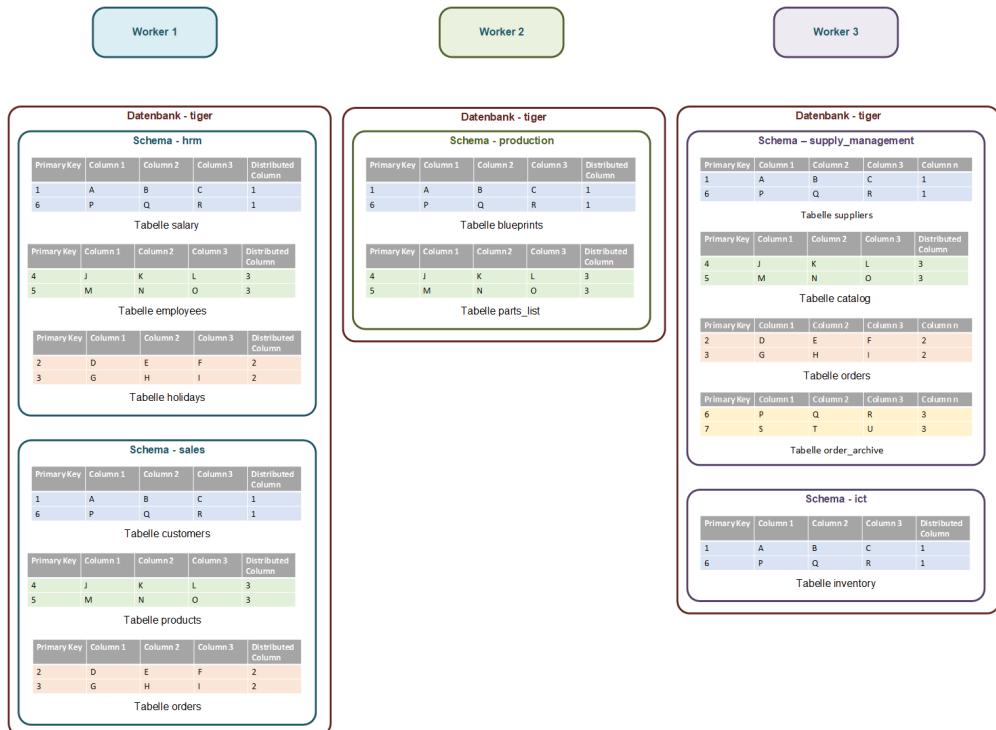


Abbildung 2.45: Citus - Schema-Based-Sharding

**Schlussfolgerung** Beide Sharding-Methoden haben eine grosse Schwäche. In Version 7.2 konnte noch ein Replikationsfaktor angegeben werden [Citus], Sie sind nicht vollständig ACID-Konform (Unterunterabschnitt 2.1.1.1) da Datenverlust entstehen kann, wenn ein Node wegfällt. Die Shards müssen mit entsprechenden mit Replikation gesichert werden [9]. Dies muss aber bei der evaluation mittels Tests noch bestätigt werden.

### 2.1.5.8.2 Maintenance

Bei Stackgres gab es im letzten Monat keine wirkliche Bewegung:

# Diplomarbeit

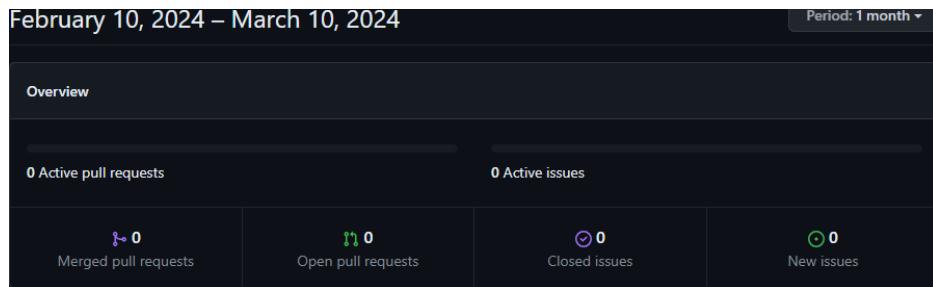


Abbildung 2.46: Stackgres - Pulse

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

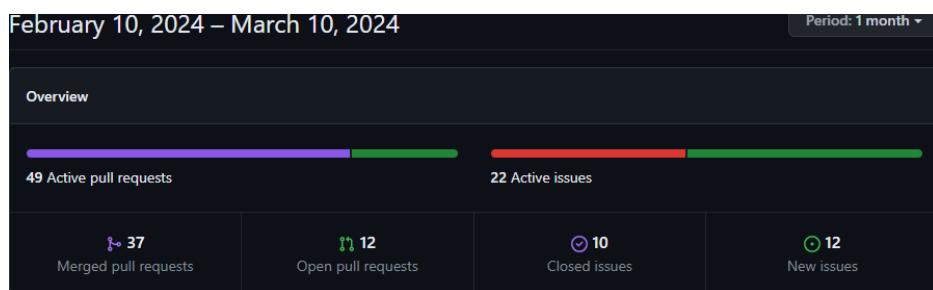


Abbildung 2.47: Citus - Pulse

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

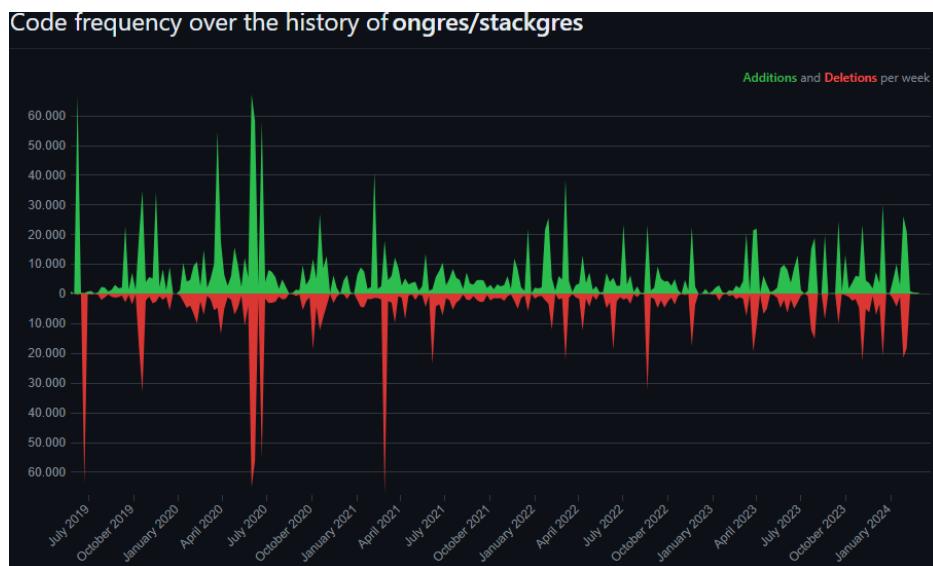


Abbildung 2.48: Stackgres - Code Frequency

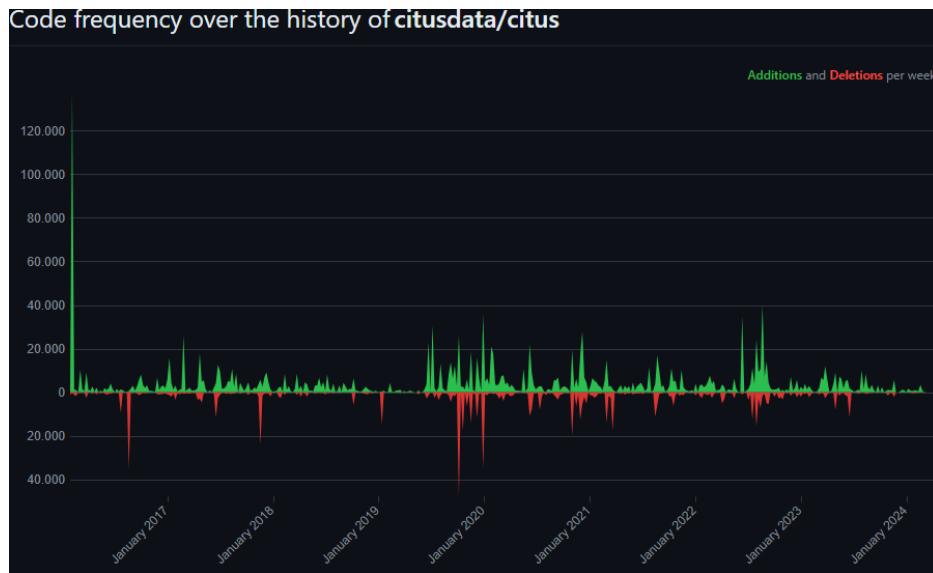


Abbildung 2.49: Citus - Code Frequency

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

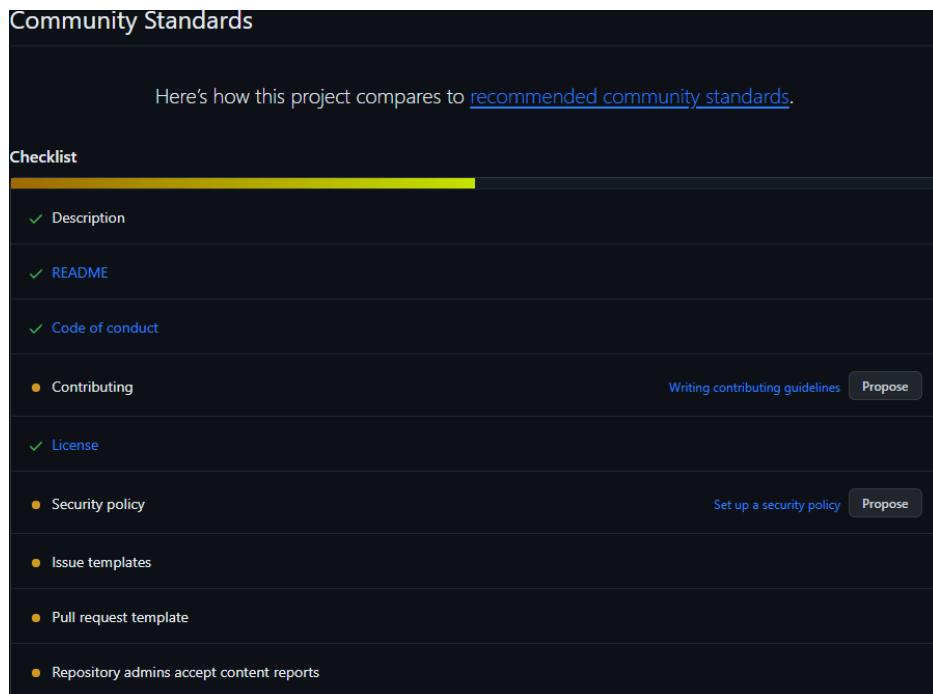


Abbildung 2.50: Stackgres - Community Standards

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 2.51: 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 2.52: Stackgres - Contributors Commits

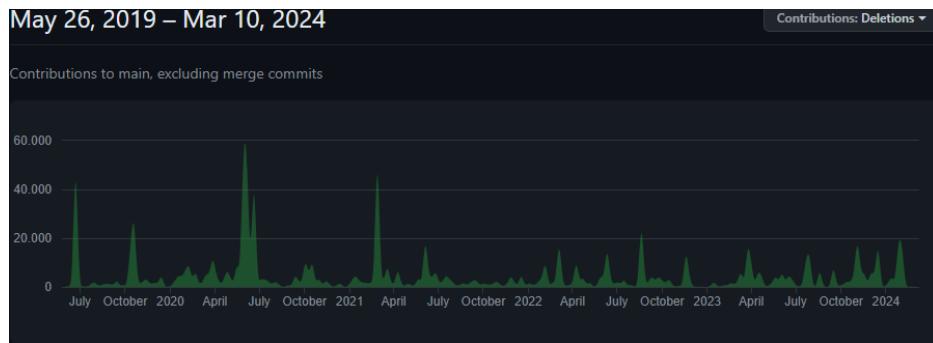


Abbildung 2.53: Stackgres - Contributors Deletions



Abbildung 2.54: Stackgres - Contributors Additions

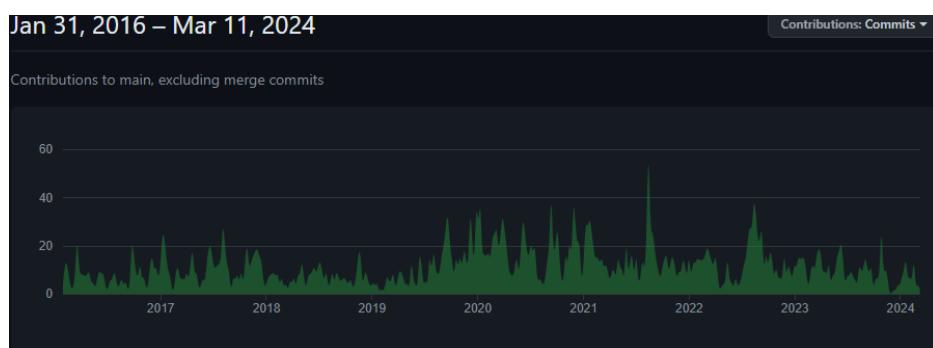


Abbildung 2.55: Citus - Contributors Commits

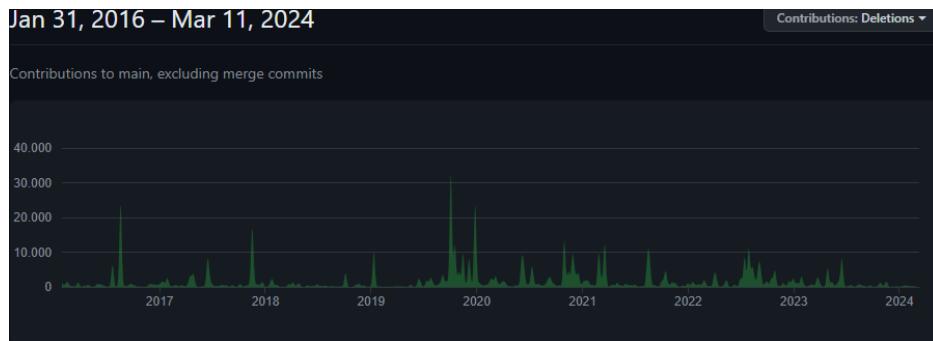


Abbildung 2.56: Citus - Contributors Deletions



Abbildung 2.57: 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:

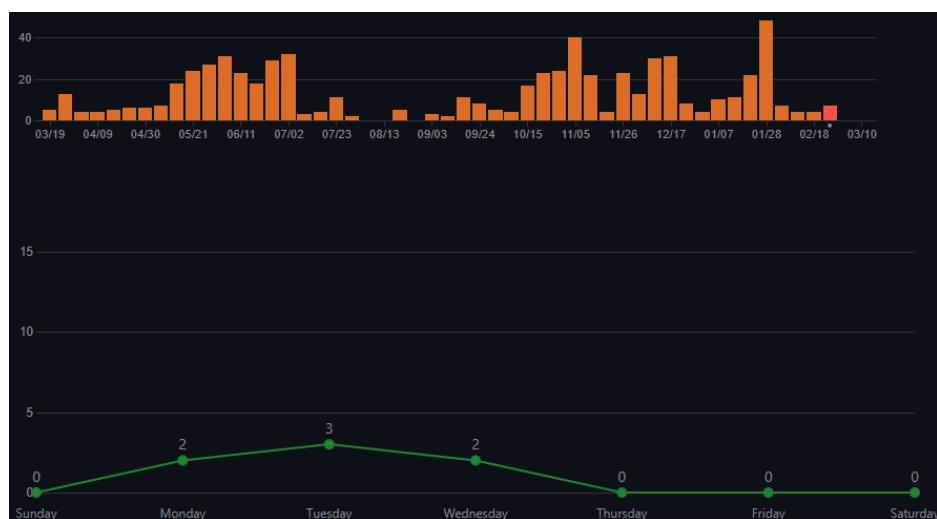


Abbildung 2.58: Stackgres - Commit Activity

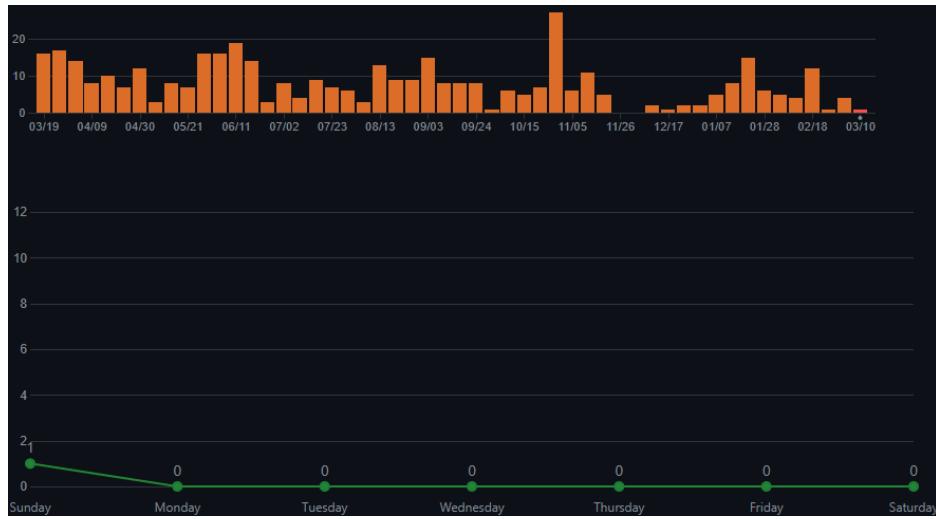


Abbildung 2.59: Citus - Commit Activity

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.

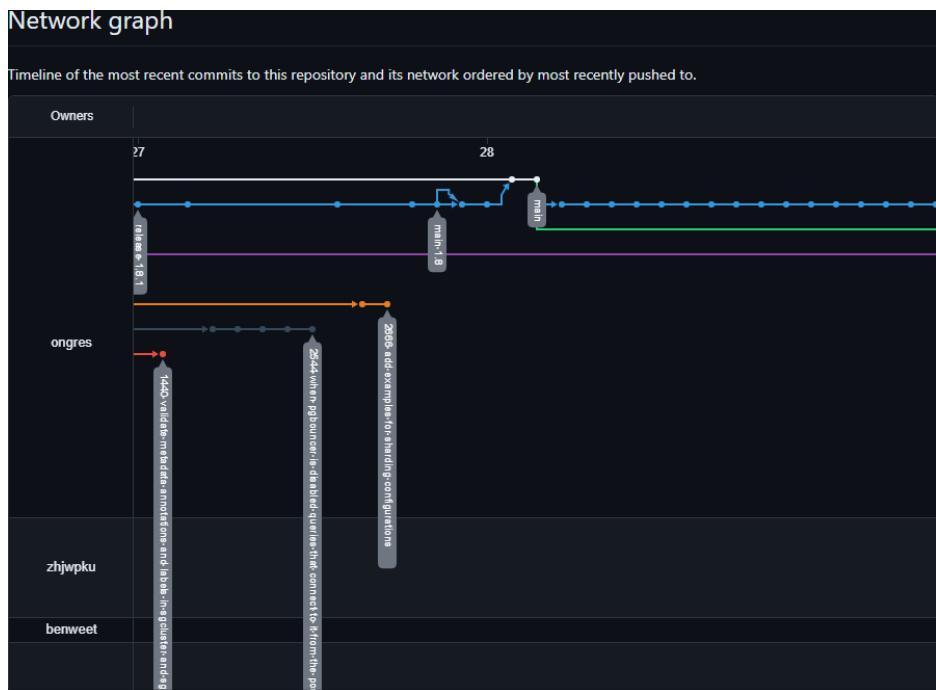


Abbildung 2.60: Stackgres - Network Graph

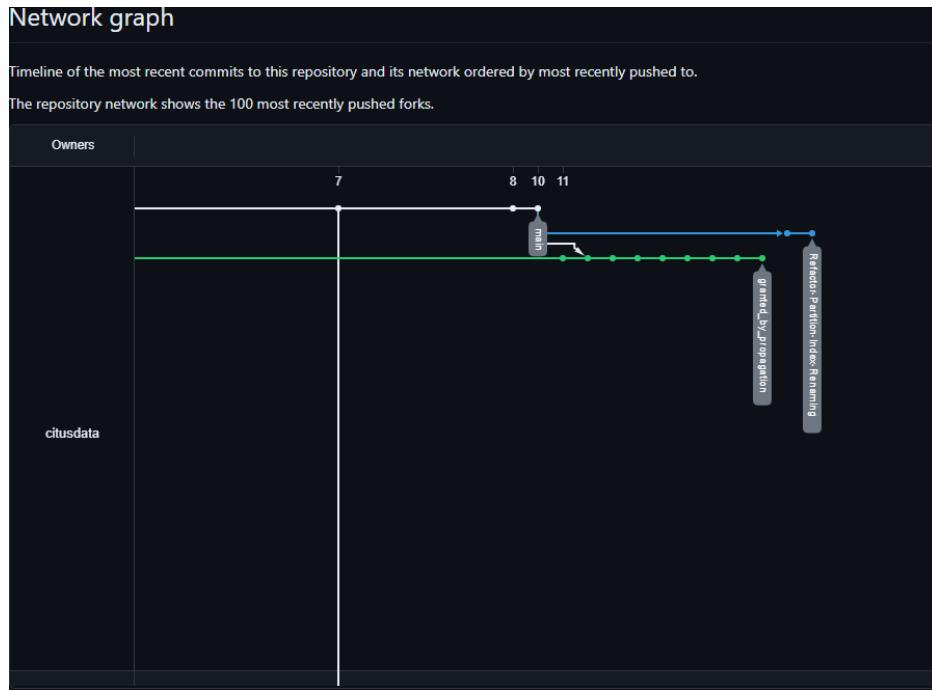


Abbildung 2.61: Citus - Network Graph

### 2.1.6 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. pg_auto_failover würde zwar Citus-Support bieten,
3	pg_auto_failover	Vorausgeschieden	allerdings gibt es keine gut dokumentierte Implementation für Kubernetes. Erfüllt daher das Kriterium für die Synergien nicht CloudNativePG ist keine vollständige Cloud Native Lösung.
4	CloudNativePG	Vorausgeschieden	Mittels Citus könnte sogar eine Distributed SQL Lösung implementiert werden. Die Grundarchitektur bleibt aber Monolithisch mit einem Primary und Repikas. 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 2.6: 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. Da Patroni unter der Haube ist, kann die API und sonstige Skripte auch auf einem Monolithischen System eingesetzt werden.
7	Yugabyte-DB	Evaluation	Ist eine reine Distributed SQL Lösung und ist Vollständig Cloud Native.

Tabelle 2.7: Vorauswahl - Evaluation

### 2.1.7 Installation verschiedener Lösungen

Entsprechend wurden folgende Server bereitgestellt:

Server	Typ	Funktion	Full Qualified Device Name	IP
sks1183	Distributed SQL	Server	sks1183.ksgr.ch	10.0.20.97
sks1184	Distributed SQL	Agent	sks1184.ksgr.ch	10.0.20.104
sks1185	Distributed SQL	Agent	sks1185.ksgr.ch	10.0.20.105
sks1232	Monolith	Server	sks1232.ksgr.ch	10.0.20.110
sks1233	Monolith	Server	sks1233.ksgr.ch	10.0.20.111
sks1234	Monolith	Server	sks1234.ksgr.ch	10.0.20.112
sks9016	Benchmark Server	Client	sks9016.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 2.8: Evaluationssysteme

#### 2.1.7.1 rke2 - Evaluationsplattform

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

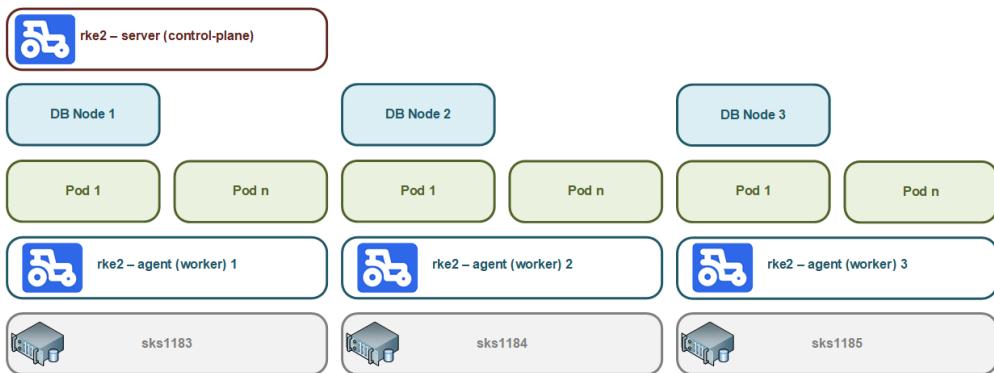


Abbildung 2.62: Evaluationssystem - Distributed SQL / Shards

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>loud Native Storage (CNS)</b>	local-path-provisioner

---

Tabelle 2.9: Evaluationssysstem - Distributed SQL / Sharding

### 2.1.7.2 Patroni

### 2.1.7.2.1 Architektur

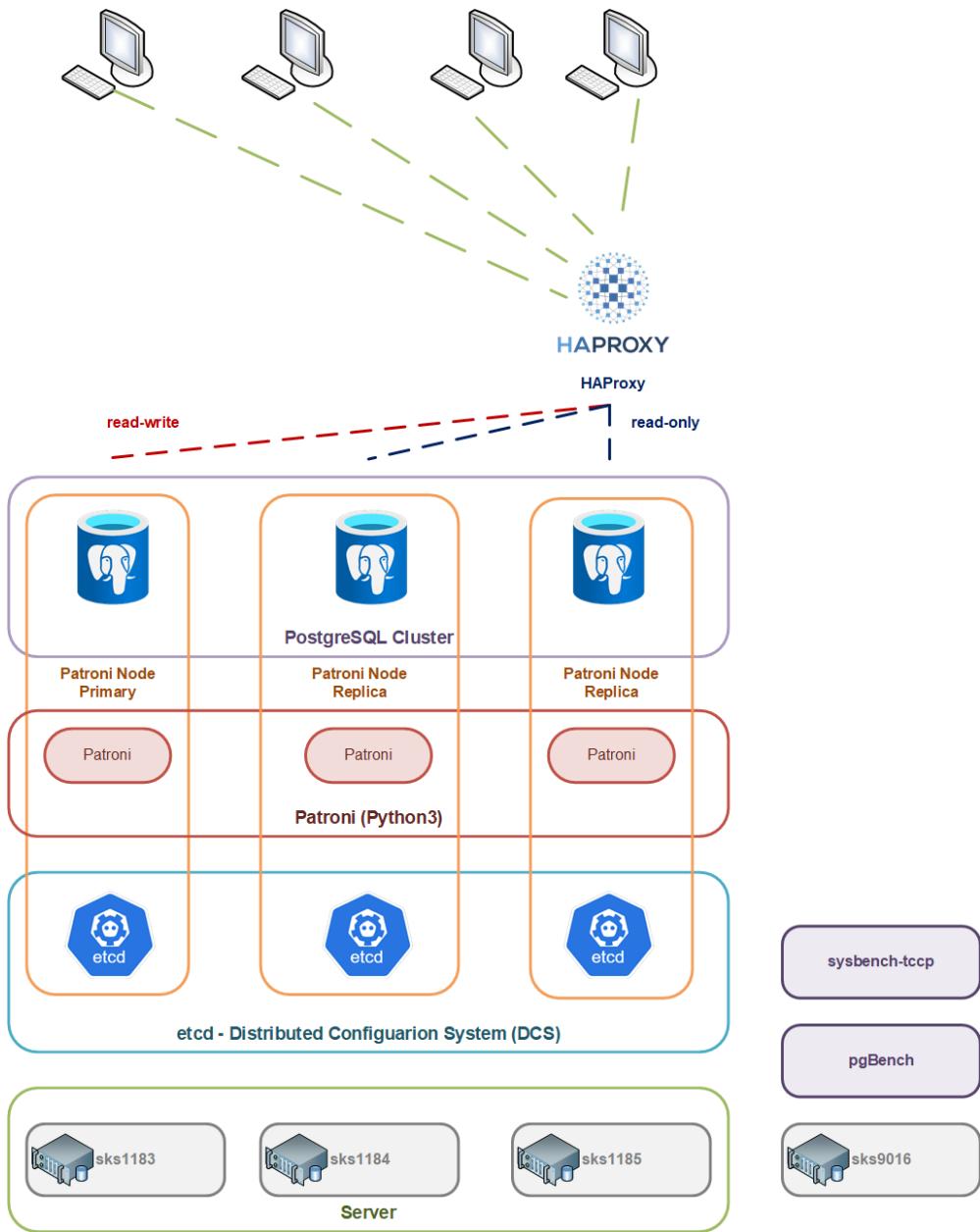


Abbildung 2.63: Patroni - Evaluationsarchitektur

### 2.1.7.3 StackGres - Citus

### 2.1.7.3.1 Architektur

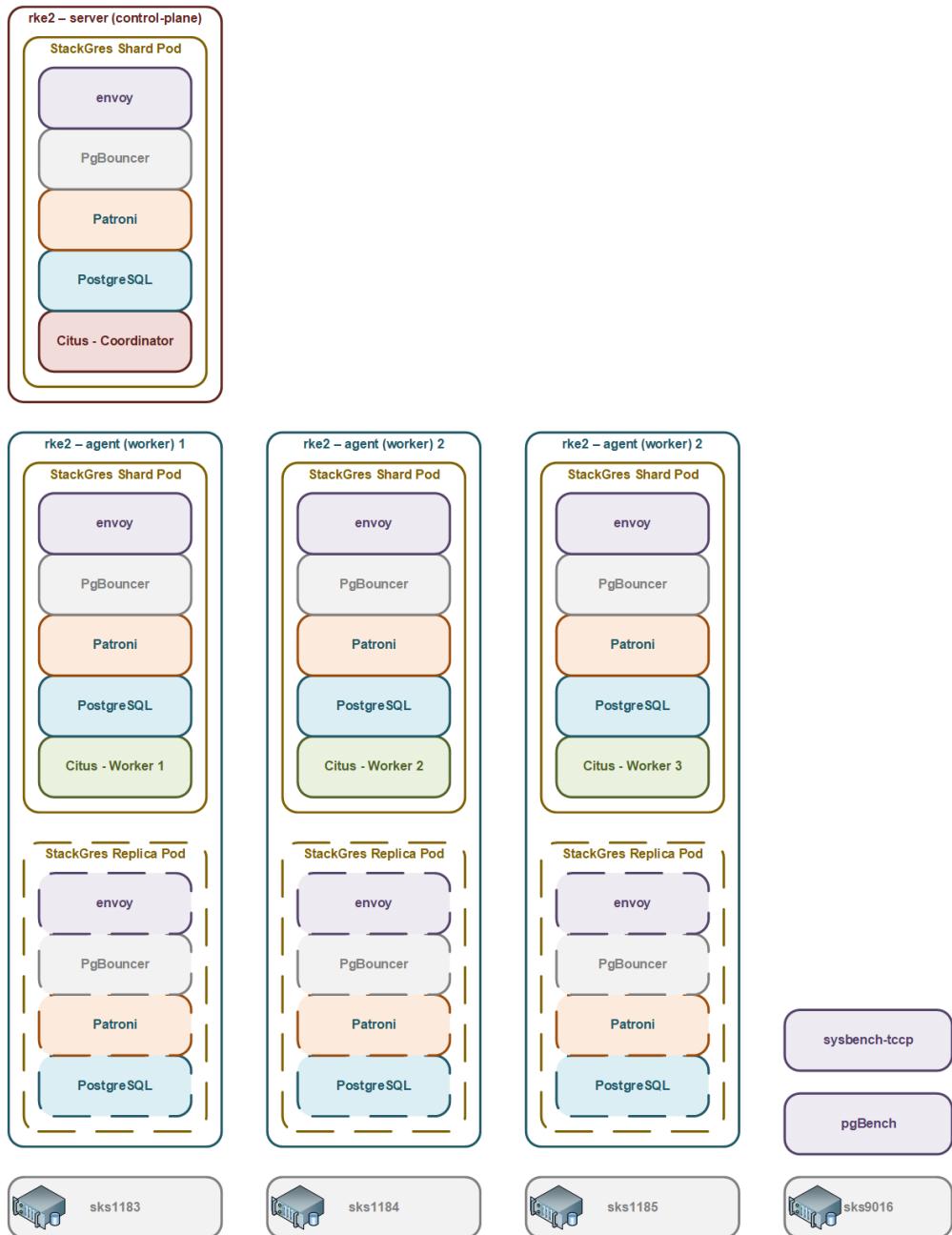


Abbildung 2.64: Stackgres - Citus - Evaluationsarchitektur

### 2.1.7.4 YugabyteDB

2.1.8 Gegenüberstellung der Lösungen

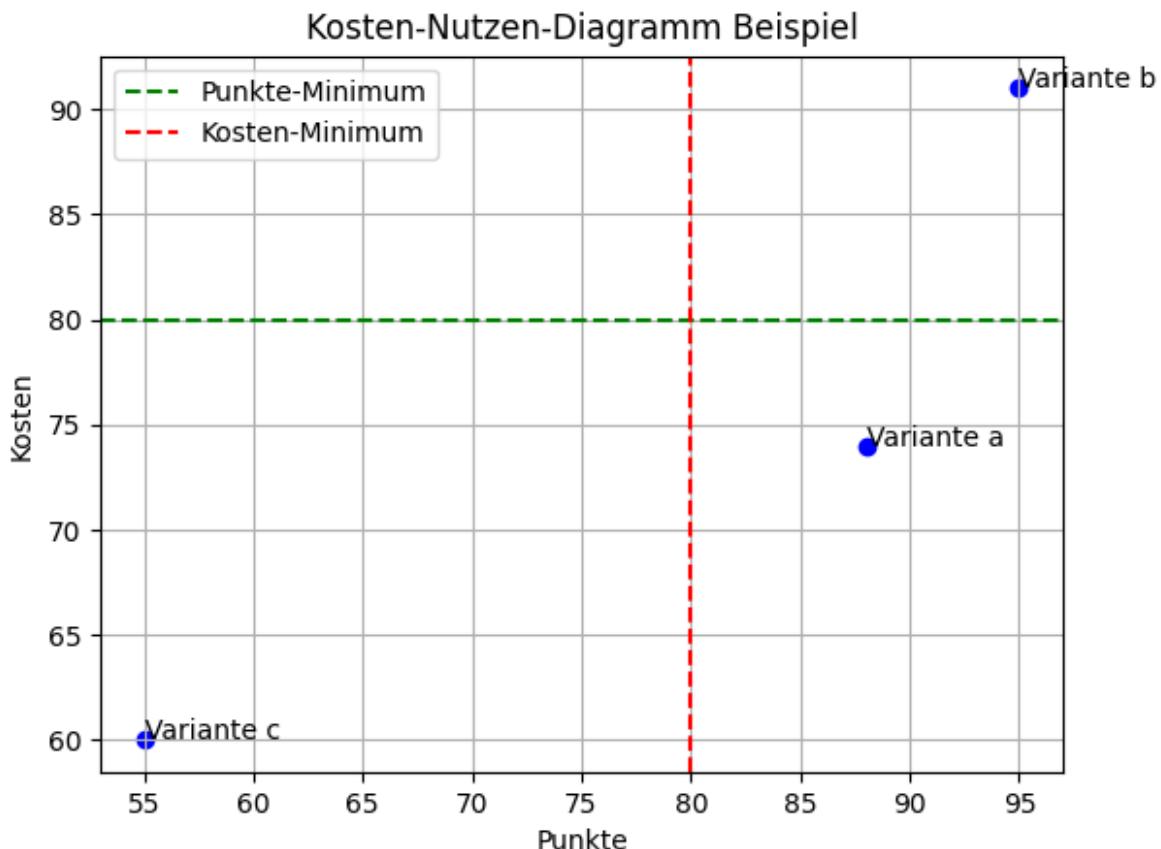


Abbildung 2.65: Kosten-Nutzen-Analyse

2.1.9 Entscheid

2.2 Aufbau und Implementation Testsystem

2.2.1 Bereitstellen der Grundinfrastruktur

2.2.2 Installation und Konfiguration PostgreSQL HA Cluster

2.2.3 Technical Review der Umgebung

2.3 Testing

2.3.1 Testing

2.3.2 Protokollierung

2.3.3      Review und Auswertung

2.4      Troubleshooting und Lösungsfindung

### **3**

#### **Resultate**

**3.1**

**Zielüberprüfung**

**3.2**

**Schlussfolgerung**

**3.3**

**Weiteres Vorgehen / offene Arbeiten**

**3.4**

**Persönliches Fazit**

## Abbildungsverzeichnis

1.1	Spitalregionen Kanton Graubünden[38]	1
1.2	Wahlkreise Kanton St. Gallen[63]	2
1.3	Spitalregionen / Spitalstrategie Kanton St. Gallen[32]	3
1.4	Organigramm Kantonsspital Graubünden	4
1.5	Organigramm Departement 10 - ICT	5
1.6	Risiken bestehende Lösung	11
1.7	Risiken bestehende Lösung mit Massnahmen	12
1.8	Systemabgrenzung	17
1.9	Projektrisiken	20
1.10	Projektrisiken mit Massnahmen	21
1.11	Riskikomatrix - Assessment 21.03.2024	25
2.1	Sharding - Vertikale Partitionierung	33
2.2	Sharding - Horizontales Partitionierung	34
2.3	Monolithische vs. verteilte SQL Systeme	36
2.4	CAP-Theorem	39
2.5	Datenbankskalierung	40
2.6	Präferenzmatrix	44
2.7	Präferenzmatrix - Failover	45
2.8	Präferenzmatrix - Switchover	46
2.9	Präferenzmatrix - Restore	47
2.10	Präferenzmatrix - Replikation	48
2.11	Präferenzmatrix - Sharding	49
2.12	Präferenzmatrix - Quorum	50
2.13	Präferenzmatrix - Management-API	51
2.14	Präferenzmatrix - Backup	52
2.15	Präferenzmatrix - Performance	53
2.16	Benchmark Settings - Zabbix - Systeminformationen	55
2.17	Benchmark Settings - Zabbix - Connections per Seconds	55
2.18	Benchmark Settings - Zabbix - Queries per Seconds	55
2.19	Benchmark Settings - Zabbix - Client Queries per Seconds	56
2.20	Benchmark Settings - Zabbix - DB Size	56
2.21	pg_auto_failover-Architektur - Single Standby	59
2.22	pg_auto_failover-Architektur - Multi-Node Standby	60
2.23	pg_auto_failover-Architektur - Citus	60

2.24 Patroni-Architektur . . . . .	62
2.25 Patroni - Pulse . . . . .	63
2.26 Patroni - Code Frequency . . . . .	63
2.27 Patroni - Community Standards . . . . .	64
2.28 Patroni - Contributors Commits . . . . .	64
2.29 Patroni - Contributors Deletations . . . . .	65
2.30 Patroni - Contributors Additions . . . . .	65
2.31 Patroni - Commit Activity . . . . .	65
2.32 Patroni - Network Graph . . . . .	66
2.33 CloudNativePG - Kubernetes - PostgreSQL . . . . .	67
2.34 CloudNativePG - Kubernetes - Read-write workloads . . . . .	68
2.35 CloudNativePG - Kubernetes - Read-only workloads . . . . .	68
2.36 YugabyteDB - Grundkonzept . . . . .	69
2.37 YugabyteDB - Architektur . . . . .	70
2.38 YugabyteDB - Sharding . . . . .	70
2.39 YugabyteDB - Tablet - Leader und Follower . . . . .	71
2.40 YugabyteDB - Tablet - Replikationsfaktor . . . . .	72
2.41 YugabyteDB - Zonen . . . . .	73
2.42 YugabyteDB - Zone outage Tolerance . . . . .	74
2.43 Citus - Coordinator und Workers . . . . .	75
2.44 Citus - Row-Based-Sharding . . . . .	76
2.45 Citus - Schema-Based-Sharding . . . . .	76
2.46 Stackgres - Pulse . . . . .	77
2.47 Citus - Pulse . . . . .	77
2.48 Stackgres - Code Frequency . . . . .	77
2.49 Citus - Code Frequency . . . . .	78
2.50 Stackgres - Community Standards . . . . .	78
2.51 Citus - Community Standards . . . . .	79
2.52 Stackgres - Contributors Commits . . . . .	79
2.53 Stackgres - Contributors Deletations . . . . .	80
2.54 Stackgres - Contributors Additions . . . . .	80
2.55 Citus - Contributors Commits . . . . .	80
2.56 Citus - Contributors Deletations . . . . .	81
2.57 Citus - Contributors Additions . . . . .	81
2.58 Stackgres - Commit Activity . . . . .	81
2.59 Citus - Commit Activity . . . . .	82
2.60 Stackgres - Network Graph . . . . .	82
2.61 Citus - Network Graph . . . . .	83
2.62 Evaluationssystem - Distributed SQL / Shards . . . . .	85

## Diplomarbeit



2.63 Patroni - Evaluationsarchitektur . . . . .	86
2.64 Stackgres - Citus - Evaluationsarchitektur . . . . .	87
2.65 Kosten-Nutzen-Analyse . . . . .	88

## Tabellenverzeichnis

1.1	Inventarisierte Datenbanksysteme . . . . .	7
1.2	Datenbankinventar . . . . .	8
1.3	Datenbankinventor - Nach Betriebssystemen aufgeschlüsselt . . . . .	8
1.4	Risiko-Matrix aktuelle Situation PostgreSQL Datenbanken . . . . .	10
1.5	Administrative Aufgaben . . . . .	13
1.6	Automatisierung Administrativer Aufgaben . . . . .	14
1.7	Ziele . . . . .	15
1.8	Gegebene Systeme . . . . .	16
1.9	Abhängigkeiten . . . . .	18
1.10	Risiko-Matrix der Diplomarbeit . . . . .	19
1.11	Neu Erkannte / Erfasste Risiken . . . . .	22
1.12	Risiko-Assessment 21.03.2024 . . . . .	24
1.13	Projektcontrolling . . . . .	26
1.14	Initialer Statusbericht . . . . .	29
1.15	Zweiter Statusbericht . . . . .	30
1.16	Fachgespräche . . . . .	31
2.1	Quorum Beispiele . . . . .	37
2.2	Anforderungskatalog . . . . .	42
2.3	Stakeholder . . . . .	43
2.4	Benchmark Settings - Mixed Transaktionen . . . . .	56
2.5	Benchmark Settings - DQL Transaktionen . . . . .	57
2.6	Vorauswahl - Ausgeschieden . . . . .	83
2.7	Vorauswahl - Evaluation . . . . .	84
2.8	Evaluationssyssteme . . . . .	84
2.9	Evaluationssysstem - Distributed SQL / Sharding . . . . .	85
I	Arbeitsrapport . . . . .	i
II	Fachgespräche - Protokoll . . . . .	ii
III	Kommentare - Anmerkung . . . . .	iii

## Listings

1	Python LaTex - zotero.py - Zotero BibLaTex Importer . . . . .	iv
2	Python LaTex - zotero_bibtex_configuration.yaml - Konfigurationsdatei - Zotero Bi- bLaTex Importer . . . . .	x
3	Python LaTex - zotero_biblatex_keystore.yaml - x-y-Achse Konfigurationsdatei - Zo- tero BibLaTex Importer . . . . .	x
4	Python LaTex - riskmatrix.py - Risikomatrizen . . . . .	xvii
5	Python LaTex - riskmatrix_plotter_conf.yaml - Konfigurationsdatei - Risikomatrizen	xxi
6	Python LaTex - riskmatrix_xy_axis_tuple_matrix.yaml - Konfigurationsdatei - Risi- komatrizen - X-Y-Achsen Tuples . . . . .	xxv
7	Python LaTex - cost_benefit_diagram.py - Kosten-Nutzen-Diagramm . . . . .	xxvi
8	Python LaTex - cost_benefit_diagram_plotter_conf.yaml - Konfigurationsdatei - Kosten- Nutzen-Diagramm . . . . .	xxviii
9	Python LaTex - pandas_dataframe_to_latex_table.py CSV - LaTex Tabelle . . . . .	xxix
10	Python LaTex - csv_to_latex_diplomarbeit.yaml - Konfigurationsdatei - CSV - LaTex- Tabelle . . . . .	xxxviii

## Literatur

- [1] *About pgbench-tools*. <https://github.com/gregs1104/pgbench-tools>. original-date: 2010-02-17T13:33:28Z. 2023.
- [2] *API Reference - CloudNativePG*. <https://cloudnative-pg.io/documentation/1.22/cloudnative-pg-v1/>.
- [3] Satyadeep Ashwathnarayana und Inc. Netdata. *How to monitor and fix Database bloats in PostgreSQL? / Netdata Blog*. <https://blog.netdata.cloud/postgresql-database-bloat/>. 2022.
- [4] unknown author. *#1 Backup-Lösung für Kubernetes*. <https://www.veeam.com/de/kubernetes-native-backup.html?ck=1697900263871>.
- [5] unknown author. *Architecture Basics — pg\_auto\_failover 2.0 documentation*. <https://pg-auto-failover.readthedocs.io/en/main/architecture.html>.
- [6] 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).
- [7] unknown author. *Citus Support — pg\_auto\_failover 2.0 documentation*. <https://pg-auto-failover.readthedocs.io/en/main/citus.html>.
- [8] unknown author. *CloudNativePG - Main Features*. <https://cloudnative-pg.io/documentation/1.22/#main-features>.
- [9] 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).
- [10] 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).
- [11] 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).
- [12] unknown author. *etcd*. <https://etcd.io/>.
- [13] unknown author. *HAProxy Documentation Converter*. <https://docs.haproxy.org/>.
- [14] unknown author. *HAProxy version 2.9.6 - Starter Guide*. <https://docs.haproxy.org/2.9/intro.html#3.2>.
- [15] unknown author. *Multi-node Architectures — pg\_auto\_failover 2.0 documentation*. <https://pg-auto-failover.readthedocs.io/en/main/architecture-multi-standby.html>.
- [16] unknown author. *Replication in DocDB - Zone Fault Tolerance*. <https://docs.yugabyte.com/preview/architecture/docdb-replication/replication/>. Section: preview.

- [17] unknown author. *Tuning PostgreSQL with pgbench*. <https://www.cloudbees.com/blog/tuning-postgresql-with-pgbench>. 2017.
- [18] unknown author. *YB-TServer service*. <https://docs.yugabyte.com/preview/architecture/concepts/yb-tserver/>. Section: preview.
- [19] GitLab B.V. und GitLab Inc. *The DevSecOps Platform | GitLab*. <https://about.gitlab.com/>.
- [20] Fernando Laudares Camargos Avinash Vallarapu. *Tuning PostgreSQL for sysbench-tpcc*. <https://www.percona.com/blog/tuning-postgresql-for-sysbench-tpcc/>. 2018.
- [21] Alexandre Cassen und Read the Docs. *Introduction — Keepalived 1.2.15 documentation*. <https://keepalived.readthedocs.io/en/latest/introduction.html>. 2017.
- [22] Microsoft Corporation. *Azure SQL-Datenbank – ein verwalteter Clouddatenbankdienst | Microsoft Azure*. <https://azure.microsoft.com/de-de/products/azure-sql/database>. 2023.
- [23] Microsoft Corporation. *Datenbank-Software und Datenbankanwendungen | Microsoft Access*. <https://www.microsoft.com/de-de/microsoft-365/access>. 2023.
- [24] Microsoft Corporation. *Microsoft Data Platform | Microsoft*. <https://www.microsoft.com/de-ch/sql-server>.
- [25] ORACLE CORPORATION. „Oracle (Active) Data Guard 19c“. In: (2019), S. 14.
- [26] Varun Dhawan und data-nerd.blog. *PostgreSQL-Diagnostic-Queries – data-nerd.blog*. <https://data-nerd.blog/2018/12/30/postgresql-diagnostic-queries/>.
- [27] *EDB: Open-Source, Enterprise Postgres Database Management*. <https://www.enterprisedb.com/>.
- [28] Elektronik-Kompendium.de und Schnabel Schnabel. *SAN - Storage Area Network*. <https://www.elektronik-kompendium.de/sites/net/0906071.htm>. 2023.
- [29] DB-Engines und solidIT consulting & software development gmbh. *DB-Engines Ranking*. <https://db-engines.com/en/ranking>.
- [30] DB-Engines und solidIT consulting & software development gmbh. *relationale Datenbanken - DB-Engines Enzyklopädie*. <https://db-engines.com/de/article/relationale+Datenbanken?ref=RDBMS>.
- [31] The Linux Foundation. *Harbor*. <https://goharbor.io/>. 2023.
- [32] 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>.
- [33] Git. *About - Git*. <https://git-scm.com/about>.

- [34] IBM Deutschland GmbH. *Was ist das CAP-Theorem?* / IBM. <https://www.ibm.com/de-de/topics/cap-theorem>. 2023.
- [35] IBM Deutschland GmbH. *Was ist OLAP?* / IBM. <https://www.ibm.com/de-de/topics/olap>.
- [36] Jedox GmbH. *Was ist OLAP? Online Analytical Processing im Überblick.* <https://www.jedox.com/de/blog/was-ist-olap/>. Section: Knowledge.
- [37] 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>.
- [38] 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/InstitutionenGesundeitwsens/Spitaeler/Dok%20Spitler/Kenndaten%202016%20Spit%C3%A4ler.pdf>.
- [39] The Pgpool Global Development Group. *What is Pgpool-II?* <https://www.pgpool.net/docs/44/en/html/intro-whatis.html>. 2023.
- [40] The PostgreSQL Global Development Group. *25.1. Routine Vacuuming.* <https://www.postgresql.org/docs/16/routine-vacuuming.html>. 2023.
- [41] The PostgreSQL Global Development Group. *27.1. Comparison of Different Solutions.* <https://www.postgresql.org/docs/16/different-replication-solutions.html>. 2023.
- [42] The PostgreSQL Global Development Group. *pgbench.* <https://www.postgresql.org/docs/16/pgbench.html>. 2023.
- [43] Inc. HashiCorp. *Terraform by HashiCorp.* <https://www.terraform.io/>.
- [44] Patrick Hunt, Mahadev Konar, Flavio P Junqueira und Benjamin Reed. „ZooKeeper: Wait-free coordination for Internet-scale systems“. In: (2010).
- [45] Splunk Inc. *Splunk / Der Schlüssel zu einem resilienten Unternehmen.* [https://www.splunk.com/de\\_de](https://www.splunk.com/de_de). 2023.
- [46] Sebastian Insausti. *Scaling PostgreSQL for Large Amounts of Data.* <https://severalnines.com/blog/scaling-postgresql-large-amounts-data/>. 2019.
- [47] Shiv Iyer und MinervaDB. *PostgreSQL DBA Daily Checklist.* <https://minervadb.xyz/postgresql-dba-daily-checklist/>. 2020.
- [48] jobinau/pg\_gather. [https://github.com/jobinau/pg\\_gather](https://github.com/jobinau/pg_gather). original-date: 2021-01-19T08:12:07Z. 2024.
- [49] Unmesh Joshi. *Quorum.* <https://martinfowler.com/articles/patterns-of-distributed-systems/quorum.html>. 2020.
- [50] Martin Keen und IBM Deutschland GmbH. *IBM Db2.* <https://www.ibm.com/de-de/products/db2>.

- [51] Pasha Kostohrys. *Database replication — an overview*. <https://medium.com/@pkostohrys/database-replication-an-overview-f7ade110477>. 2020.
- [52] Anatoli Kreyman. *Was ist eigentlich Splunk?* <https://www.kreyman.de/index.php/splunk/76-was-ist-eigentlich-splunk-big-data-platform-monitoring-security>.
- [53] 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>.
- [54] Red Hat Limited. *Was ist Ansible?* <https://www.redhat.com/de/technologies/management/ansible/what-is-ansible>.
- [55] 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>.
- [56] Switzerland Linuxfabrik GmbH Zurich. *Keepalived — Open Source Admin-Handbuch der Linuxfabrik*. <https://docs.linuxfabrik.ch/software/keepalived.html>. 2023.
- [57] Nico Litzel, Stefan Luber und Vogel IT-Medien GmbH. *Was ist Elasticsearch?* <https://www.bigdata-insider.de/was-ist-elasticsearch-a-939625/>. 2020.
- [58] SRA OSS LLC. *pgpool Wiki*. [https://www.pgpool.net/mediawiki/index.php/Main\\_Page](https://www.pgpool.net/mediawiki/index.php/Main_Page). 2023.
- [59] Hewlett Packard Enterprise Development LP. *Was ist SAN-Speicher? / Glossar*. <https://www.hpe.com/ch/de/what-is/san-storage.html>.
- [60] Julian Markwort. „Benchmarking four Different Replication Solutions“. In: () .
- [61] Diego Ongaro. „Consensus: Bridging Theory and Practice“. In: (2014) .
- [62] 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.
- [63] 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>.
- [64] Ed Reckers und SnapLogic Inc. *Was ist die Snowflake-Datenplattform?* <https://www.snaplogic.com/de/blog/snowflake-data-platform>. 2023.
- [65] IONOS SE. *Apache Cassandra: Verteilte Verwaltung großer Datenbanken*. <https://www.ionos.de/digitalguide/hosting/hosting-technik/apache-cassandra-vorgestellt/>. 2021.
- [66] IONOS SE. *Datenbankmanagementsystem (DBMS) erklärt*. <https://www.ionos.de/digitalguide/hosting/hosting-technik/datenbankmanagementsystem-dbms-erklaert/>. 2020.
- [67] IONOS SE. *MongoDB – die flexible und skalierbare NoSQL-Datenbank*. <https://www.ionos.de/digitalguide/websites/web-entwicklung/mongodb-vorstellung-und-vergleich-mit-mys>. 2019.

- [68] IONOS SE. *SQLite: Die bekannte Programmbibliothek im Detail vorgestellt.* <https://www.ionos.de/digitalguide/websites/web-entwicklung/sqlite/>. 2023.
- [69] IONOS SE. *Terraform.* <https://www.ionos.de/digitalguide/server/tools/was-ist-terraform/>. 2020.
- [70] IONOS SE. *Was ist Redis? Die Datenbank vorgestellt.* <https://www.ionos.de/digitalguide/hosting/hosting-technik/was-ist-redis/>. 2020.
- [71] IONOS SE. *Was ist SIEM (Security Information and Event Management)?* <https://www.ionos.de/digitalguide/server/sicherheit/was-ist-siem/>. 2020.
- [72] Naveed Shaikh. *It's All About Replication Lag in PostgreSQL.* <https://www.percona.com/blogreplication-lag-in-postgresql/>. 2023.
- [73] Sami Ahmed Siddiqui. *Distributed SQL 101.* <https://www.yugabyte.com/distributed-sql/>.
- [74] Inc. Snowflake. *Datenbanken, Tabellen und Ansichten – Überblick | Snowflake Documentation.* <https://docs.snowflake.com/de/guides-overview-db>.
- [75] 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).
- [76] Mahmut Can Uçanefe. *Pgbench Load Test.* <https://medium.com/@c.ucanefe/pgbench-load-test-166b203>. 2023.
- [77] Rainer Züst. „Einstieg ins Systems Engineering“. In: (2002).

## 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[54].. 16

**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[40].. 14, 15, 54

**Cassandra** Cassandra ist eine Spaltenorganisierte NoSQL-Datenbank die 2008 veröffentlicht[65] wurde.. 7, 69

**CI/CD** Continuous Integration/Continuous Delivery bedeutet, dass Anpassungen kontinuierlich in die Entwicklungsumgebungen integriert und auf die Zielplattformen verteilt werden[55].. 4

**DBMS** Ein Database Management System regelt und organisiert die Datenbasis einer Datenbank[66].. 4

**DCS** Der DCS ist eine Kernkomponente von Patroni [11]. Realisiert wird der DCS bei Patroni mit Etcd.. 5

**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.. 16

**Elasticsearch** Elasticsearch ist eine 2010 veröffentlichte Open-Source Suchmaschine die auf Basis von JSON-Dokumenten und einer NoSQL-Datenbank arbeitet[57].. 7

**etcd** etcd ist [12]. 61, 101

**Failover** In einem Fehlerfall wird in einem HA-System meist ein Primary Node auf den Secondary ungeplant geswitched.. 15, 36, 37, 58, 102

**Foreman** Foreman ist ein Lifecycle Management und Provisioning System für Virtuelle und Physische Server. Ab Version 6 basierte der Red Hat Satellite auf Foreman. 16, 19

**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[33, 75].. 102

**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[19].. 15, 58

**HAProxy** HAProxy [14]. 59, 61

**Harbor** Harbor ist ein Open-Source-Tool zur Registrierung von Richtlinien rollenbasierten Zugriffssteuerung[31]. Harbor wird beim KSGR zur Verwaltung der Kubernetes-Plattform verwendet.. 15, 58

**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.. 4, 5, 8, 19

**IBM DB2** IBM DB2 ist eine Relationale Datenbank[50] deren Vorläufer System-R von IBM zwischen 1975 und 1979 entwickelt wurde. DB2 selber wurde 1983 von IBM veröffentlicht.. 7, 39

**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[56, 21].. 58

**Key-Value-orientierte** Siehe Key-Value-Datenbank. 105

**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. 102, 103

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

**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.. 4, 8, 16, 102

**Linux** Linux ist ein Open-Source Betriebssystem, welches von Linus Torvalds 1991 in seiner frühesten Form entwickelt wurde und lose 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, 103

**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, 8

**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[22].. 7

**Microsoft Access** Access wurde 1992 veröffentlicht und ist Entwicklungsumgebung, Front- und Backend-Software und Relationale Datenbank in einem[23].. 7

**Microsoft SQL Server** MS SQL Server ist das RDBMS von Microsoft[24]. 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, 103

**MongoDB** MongoDB ist eine dokumentenorientierte NoSQL-Datenbank, die zum ersten Mal 2007 veröffentlicht wurde[67].. 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 Microsystem 2008 MySQL übernahm, hielt sich die Option frei, bei einem Kauf von Sun Microsystem durch Oracle gründen zu dürfen. Seit Oracle Sun Microsystem 2010 gekauft hat, wurden immer mehr Funktionalitäten von der Community Edition zu der Enterprise Edition verschoben worden. Aus diesem Grund hat heute der MySQL Fork MariaDB MySQL mehrheitlich aus allen Linux Distributionen als Standard Datenbank verdrängt.. 5, 7, 8

**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, 101, 102, 103, 105

**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[36, 35]. OLAP-Systeme bieten eine Hochperformante Analyse grosser Datenmengen und sind oftmals zentraler Teil eines Data-Warehouses.. 4, 7

**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.. 16

**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 erschienenn 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 Performamce oder grosse Datenmengen geht.. 5, 7, 8, 39, 103

## PKI . 5

**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, 8, 9, 13, 39, 57, 58, 69

**PostgreSQL HA Cluster** Der HA Cluster des PostgreSQL Clusters. 15

**PostgreSQL Cluster** Ein PostgreSQL Cluster entspricht einer Instanz bei MS SQL oder einer Container Database wei Oracle.. 14, 15, 58, 104

**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.. 4, 5, 14, 16

**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 Konstelationen wird daher darauf geachtet, eine ungerade Anzahl

Nodes im Cluster zu halten um keine Pat-Situation zu provozieren. Im Kapitel [Unterabschnitt 2.1.1.5](#) wird genauer auf die Mechanik eines Quorums eingegangen. . 58, 102

**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[30].. 4, 69

**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 aderen Linux Distributionen ab.. 16

**Redis** Redis ist eine Key-Value-orientierte NoSQL In-Memory-Datenbank, dh. die Daten liegen Primär im Memory und nicht auf dem Storage[70]. Redis wurde 2009 zum ersten Mal veröffentlicht.. 7, 102

**Rocky Linux** Rocky Linux basierte auf der offen zugänglichen Linux Distribution CentOS welche RHEL Binärkompatibel war und gilt als inoffizieller Nachfolger von CentOS.. 16

**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[28, 59, 37]. 4, 5, 16, 19

**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[71].. 4, 16

**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[74, 64]. 7

**Split-brain** Im Kapitel ?? werden die ursachen und folgenden eines Split-brains genauer besprochen. . 37, 104

**Splunk** Splunk ist Big Data Plattform, Monitoring- und Security-Tool in einem[45, 52]. . 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[68].. 7

**Switchover** In einem Maintenance-Fall in einem HA-System meist ein Primary Node auf den Secondary geplant geswitchen.. 15

**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..  
4

**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[69, 43].. 16

**Transaktion** Eine Transaktion ist beinhaltet Schreib-, Lese-, Mutatations- oder Löschoperationen auf Daten.. 32, 54, 56

**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 entstanden im Laufe der Zeit viele offene und Proprietäre Derivate deren Einfluss weit über die Welt der Informatik reicht.. 4

**VRRP** VRRP . 4, 102

**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. 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.. 8, 16

## **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
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 erweitern			
27.02.2024	13:00	16:00	3.0	Dokumentation	-	Dokumentation erweitern		Viele LaTeX Tabellen.	
28.02.2024	09:00	11:00	2.0	Dokumentation	-	Dokumentation erweitern		Viele LaTeX Tabellen.	Generator mit python pandas gebaut für alle möglichen Tabellen. Inkl. Aggregation und Pivot-Mechaniken 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 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. Das macht es unmöglich, diese Lösung mit all den anderen zu vergleichen.	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 erweitern			
11.03.2024	16:45	17:30	0.5	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		Citus row-based-sharding		
14.03.2024	20:45	21:30	0.8	Dokumentation	-	Projektcontrolling Arbeiten	Citus row-based-sharding Dokumentieren		
16.03.2024	17:45	18:30	0.8	Dokumentation	-	Projektcontrolling Arbeiten			
17.03.2024	14:45	16:30	1.8	Dokumentation	-	Dokumentation erweitern	Zweiter Statusbericht verfassen		
17.03.2024	19:30	20:00	0.5	Dokumentation	-	Dokumentation erweitern	ACID Exkurs erfassen		
							Listings sauber machen.		
17.03.2024	20:15	21:00	0.8	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 erweitern		Statusbericht 2 fertig Schreiben und Mail an Norman Süstrunk senden	
18.03.2024	20:20	21:50	1.5	Evaluation	Vorbereitung Benchmarking	pgbench analysieren			
19.03.2024	08:00	10:00	2.0	Evaluation	Analyse PostgreSQL HA Cluster Lösungen	yugabytedb		Percona ist Dein Freund	
19.03.2024	10:00	10:30	0.5	Evaluation	Analyse PostgreSQL HA Cluster Lösungen	Backup Anbindungen		Veeam Kast K10 wird nicht vor Angabe Diplomarbeit fertig sein	Backups lokal speichern. Veeam Integration wird im Nachgang implementiert.
19.03.2024	14:00	16:00	2.0	Dokumentation	-	Dokumentation erweitern	yugabytedb		
20.03.2024	16:00	16:15	0.2	Dokumentation	-	Termin für 2. Fachgespräch organisieren			
21.03.2024	18:00	20:00	2.0	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 erweitern	Projektcontrolling und Dokumentation		

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

==

### III Kommentare / Anmerkungen

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

Woche	Beschreibung / Event / Problem	Unnamed: 2
KW10	Vier ganze Tage war ich in Thalwil für die Oracle Multitenant-Schulung für das ExaCC Projekt (Ablösung HP-UX). Am Freitag war ich ebenfalls fast den ganzen Tag dran. Weitere Termine werden folgen, das Risiko durch das Projekt tritt langsam ein. Projekt Zeitlich im Verzug.	
KW11	Nebst dem HP-UX Ablösungsprojekt schlagen auch diverse Betriebsthemen ein. Die Analyse der PostgreSQL HA Cluster nimmt ebenfalls mehr Zeit in Anspruch, als erwartet. - HP-UX Probleme am Montag. Backups sind über das Weekend nicht durchgelaufen.	
KW12	Die ganze Montagsplanung wurde über den Haufen geworfen. - Besprechung bezüglich Backup. Veeam Kasten steht noch nicht zur Verfügung. - Mittwochvormittag in Zürich, am Nachmittag Probleme mit dfs-Shares.	
KW12	So wenig Zeit. - Mit Norman Termin für nächste Woche Fachgespräch organisiert. Freue mich darauf.	

Tabelle III: Kommentare - Anmerkung

## IV 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
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     if total_elemets < result_limit:
46         URL_ALL_ITEMS = 'https://api.zotero.org/' + str(access_type) + '/' + str(
47             zotero_access_id) + '/collections/' + str(collection_id) + '/items?' +
48             limit=' + str(
49                 result_limit) + '?format=json&sort=dateAdded&direction=asc'
50         zotero_result = downlaod_zotero_datas(URL_ALL_ITEMS , API_KEY)
51
52         zotero_data.extend(zotero_result)
53     else:
54         runs = int(math.ceil(total_elemets / result_limit))
55         index = 0
56         start_index = 0
57         while index < runs:
58             URL_Separated = 'https://api.zotero.org/' + str(access_type) + '/' +
59             str(
60                 zotero_access_id) + '/collections/' + str(collection_id) + '/items?' +
61             limit=' + str(
62                 result_limit) + '?format=json&sort=dateAdded&direction=asc' + '&' +
63             start=' + str(start_index)
64             zotero_result = downlaod_zotero_datas(URL_Separated , API_KEY)
65
66             zotero_data.extend(zotero_result)
67
68             start_index += result_limit
69             index += 1
70
71     return zotero_data
72
73 # Convert String to Datetime
74 def convert_to_datetime(input_str, parserinfo=None):
75     return parse(input_str, parserinfo=parserinfo)
76
77 # Get Dates from Datetime
78 def get_dates(date, bibtex_item_type, bibtex_month_attributes):
79     dated_date = convert_to_datetime(date)
80     return_value = dict()
81     if bibtex_item_type in bibtex_month_attributes:
82         year = dated_date.year
83         month = dated_date.month
84         return_value = {'year': year, 'month': month}
85     else:
86         year = dated_date.year
87         return_value = {'year': year}
88
89     return return_value
90
91 # Split Creators into bibtex Creators
92 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
179             # First Master thesis
180             if any(item in bibtex_item_titel for item in master_list):
181                 bibtex_item_key = 'masterthesis'
182             # Second PHD Thesis

```

```

181         elif any(item in bibtex_item_titel for item in phd_list):
182             bibtex_item_key = 'phdthesis'
183         else:
184             bibtex_item_key = 'masterthesis',
185         else:
186             if zotero_bibtex_keys.get(zotero_item_type).get('key'):
187                 bibtex_item_key = zotero_bibtex_keys.get(zotero_item_type).get(
188                     'key')
189             else:
190                 bibtex_item_key = 'misc'
191
192     # get all Keys for the zotero item type
193     entryset = '\n'
194     entry = ''
195
196     zotero_item_attributes = zotero_bibtex_keys.get(zotero_item_type).get(
197         'attributes').keys()
198     item_attributes = sorted(zotero_item_attributes, reverse=True)
199
200     for index, item_attribute in enumerate(item_attributes):
201         bibtex_item_attribute = zotero_bibtex_keys.get(zotero_item_type).get(
202             'attributes').get(item_attribute)
203         zotero_item_value = biblio_item.get(item_attribute)
204         zotero_item_value_extra = '',
205         bibtex_item_attribute_extra = '',
206
207         # Special Cases
208         if bibtex_item_attribute == 'SPECIALCHECK' and zotero_item_value not
209         in ['', None]:
210             bibtex_special_attribute = zotero_bibtex_attributes_special.get(
211                 item_attribute)
212
213             match bibtex_special_attribute:
214                 case 'get_dates':
215                     zotero_item_value = get_dates(zotero_item_value,
216                     bibtex_item_key, bibtex_month_attributes)
217                     if zotero_item_value.get('month'):
218                         zotero_item_value_extra = zotero_item_value.get('month')
219
220                     bibtex_item_attribute_extra = 'month'
221
222                     zotero_item_value = zotero_item_value.get('year')
223                     bibtex_item_attribute = 'year'
224                     case 'split_creators':
225                         authors = split_creators(zotero_item_value)
226                         entryset = entryset + authors
227                     elif bibtex_item_attribute == 'howpublished':
228                         if zotero_item_value not in ['', None, []]:

```

```

222         zotero_item_value = '\url{' + zotero_item_value + '}'
223
224     if bibtex_item_attribute not in ['', 'None', 'author', 'SPECIALCHECK'] and zotero_item_value not in ['', None, []]:
225         if zotero_item_value_extra:
226
227             if type(zotero_item_value_extra) == "string":
228                 entryset = entryset + str(bibtex_item_attribute_extra) + '=' + str(zotero_item_value_extra) + '\n'
229             else:
230                 entryset = entryset + str(bibtex_item_attribute_extra) + '=' + str(zotero_item_value_extra)
231
232             if index != len(item_attributes) - 1:
233                 entryset = entryset + ',\n'
234             else:
235                 entryset = entryset + '\n'
236
237             if type(zotero_item_value) == str and not zotero_item_value.isnumeric():
238                 entryset = entryset + str(bibtex_item_attribute) + '=' + str(zotero_item_value) + '\n'
239             else:
240                 entryset = entryset + str(bibtex_item_attribute) + '=' + str(zotero_item_value)
241
242             if index != len(item_attributes) - 1:
243                 entryset = entryset + ',\n'
244             else:
245                 entryset = entryset + '\n'
246
247     # create the Entry
248     entry = '@' + bibtex_item_key + '{' + zotero_item_key + ',\n'
249     entry = entry + entryset + '}' + '\n'
250     bib_data = bib_data + '\n' + entry
251
252 # parse String to pybtex.database Object
253 # bib_datas = pybtex.database.parse_string(bib_data, bib_format="bibtex",
254 encoding='ISO-8859-1')
255
256     bib_datas = pybtex.database.parse_string(bib_data, bib_format="bibtex",
257 encoding='Utf-8')
258
259     # Save pybtex.database to file
260     # BibliographyData.to_file(bib_datas, file_path, bib_format="bibtex", encoding=
261     # 'ISO-8859-1')
262     BibliographyData.to_file(bib_datas, file_path, bib_format="bibtex", encoding=
263     'utf-8')

```

```

260 zotero_bibtex_config = load_configuration('zotero_bibtex_configuration.yaml')
261 zotero_data = get_data(zotero_bibtex_config)
262 write_bibliography(zotero_data, zotero_bibtex_config)

```

Listing 1: Python LaTex - zotero.py - Zotero BibLaTex Importer

## V 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_biblatex_keystore.yaml"
7 keystore_filepath: "source/configuration"
8 bibtex_filepath: "source"
9 bibtex_filename: "Diplomarbeit_Michael_Graber.bib"

```

Listing 2: Python LaTex - zotero\_bibtex\_configuration.yaml - Konfigurationsdatei - Zotero BibLaTex Importer

## VI zotero\_biblatex\_keystore.yaml

```

1 ---
2 artwork:
3   key: misc
4   attributes:
5     title: title
6     date: SPECIALCHECK
7     creators: SPECIALCHECK
8     url: howpublished
9     extra: note
10 audioRecording:
11   key: misc
12   attributes:
13     title: title
14     date: SPECIALCHECK
15     creators: SPECIALCHECK
16 bill:
17   key: misc
18   attributes:
19     title: title
20     date: SPECIALCHECK
21     creators: SPECIALCHECK
22     url: howpublished
23     extra: note

```

```
24 blogPost:  
25   key: misc  
26   attributes:  
27     title: title  
28     date: SPECIALCHECK  
29     creators: SPECIALCHECK  
30     url: howpublished  
31     extra: note  
32 book:  
33   key: book  
34   attributes:  
35     title: title  
36     date: SPECIALCHECK  
37     creators: SPECIALCHECK  
38     publisher: publisher  
39     place: address  
40 bookSection:  
41   key: inbook  
42   attributes:  
43     title: title  
44     date: SPECIALCHECK  
45     creators: SPECIALCHECK  
46     pages: pages  
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
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
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
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:
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:
287         title: title
288         date: SPECIALCHECK
289         creators: SPECIALCHECK
290         url: howpublished
291         extra: note
292 note:
293     key: misc
294     attributes:
295         title: title
296         date: SPECIALCHECK
297         creators: SPECIALCHECK
298         url: howpublished
299         extra: note
300 standard:
301     key: misc
302     attributes:
303         title: title
304         date: SPECIALCHECK
305         creators: SPECIALCHECK
306         url: howpublished
307         extra: note
308 preprint:
309     key: misc
310     attributes:
311         title: title
```

```

312     date: SPECIALCHECK
313     creators: SPECIALCHECK
314     url: howpublished
315     extra: note
316 dataset:
317   key: misc
318   attributes:
319     title: title
320     date: SPECIALCHECK
321     creators: SPECIALCHECK
322     url: howpublished
323     extra: note
324 thesis:
325   key: thesis
326   attributes:
327     title: title
328     date: SPECIALCHECK
329     creators: SPECIALCHECK
330     place: address
331     university: school

```

Listing 3: Python LaTex - zotero\_biblatex\_keystore.yaml - x-y-Achse Konfigurationsdatei - Zotero BibLaTex Importer

## VII 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(
21         'configfile_path')
22     riskmatrix_xy_axis_tuples_config = riskmatrix_config.get('riskmatrix').get(
23         'configfile_name')
24
25     if startpath == 'homedir':
26         directory = os.path.join(os.getcwd(), riskmatrix_xy_axis_tuples_dir)
27     else: # parentdir
28         directory = os.path.join(os.path.dirname(os.getcwd()),
29             riskmatrix_xy_axis_tuples_dir)
30
31     riskmatrix_xy_axis_tuples_path = os.path.join(directory,
32         riskmatrix_xy_axis_tuples_config)
33     riskmatrix_xy_axis_tuples = dict()
34     riskmatrix_xy_axis_tuples_aux = dict()
35
36     with open(riskmatrix_xy_axis_tuples_path, "r") as file:
37         riskmatrix_xy_axis_tuples_aux = yaml.load(file, Loader=yaml.FullLoader)
38
39     for string_key in riskmatrix_xy_axis_tuples_aux:
40         value = riskmatrix_xy_axis_tuples_aux.get(string_key)
41         int_key = eval(string_key)
42         riskmatrix_xy_axis_tuples.update({int_key:value})
43
44     return riskmatrix_xy_axis_tuples
45
46
47 # Load Data from csv
48 def get_data(data_path):
49
50     with open(data_path) as f:
51         csv_list = [[val.strip() for val in r.split(",")] for r in f.readlines()]
52
53     (_, *header), *data = csv_list
54     datas = {}
55     for row in data:
56         key, *values = row
57         datas[key] = {key: value for key, value in zip(header, values)}
58
59     return datas
60
61
62 # Generate Riskmatrix Image
63 #def riskmatrix(risk, conf, matrix):
64 def riskmatrix(conf, matrix):
65     risks = conf.get('risk_inventory')
66     for risk_conf in risks:
67         # get the risk config datas
68         startpath = conf.get('risks').get(risk_conf).get('startpath')
69         destination = conf.get('risks').get(risk_conf).get('destination_path')
70         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     # Identify the index of the axes
72     green = conf.get('risks').get(risk_conf).get('settings').get('green-boxes',
73   )
73     yellow = conf.get('risks').get(risk_conf).get('settings').get('yellow-
74       boxes')
74     orange = conf.get('risks').get(risk_conf).get('settings').get('orange-
75       boxes')
75     red = conf.get('risks').get(risk_conf).get('settings').get('red-boxes')

76
77     if startpath == 'homedir':
78         directory = os.path.join(os.getcwd(), destination)
79     else: # parentdir
80         directory = os.path.join(os.path.dirname(os.getcwd()), destination)

81
82     data_path = os.path.join(directory, datafilename)
83     image_path = os.path.join(directory, imagename)

84
85     # get the Data as direct
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([])
106        ax.set_yticks([])
```

```
107         ax.set_xlim(0, 5)
108         ax.set_ylim(0, 5)
109
110     # Add background colors
111     # This has been done manually for more fine-grained control
112     # Run the loop below to identify the indice of the axes
113     for _ in green:
114         axes[_].set_facecolor('green')
115
116     for _ in yellow:
117         axes[_].set_facecolor('yellow')
118
119     for _ in orange:
120         axes[_].set_facecolor('orange')
121
122     for _ in red:
123         axes[_].set_facecolor('red')
124
125     # run through datas and generate axis datas
126     dict_bubble_axis = dict()
127     bubble_axis = list()
128     for datasets in datas:
129         # get the datas
130         riskid = datas.get(datasets).get('risk-id')
131         x_axis = int(datas.get(datasets).get('x-axis'))
132         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 riks 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 4: Python LaTex - riskmatrix.py - Risikomatrizen

## VIII 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  
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
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 5: Python LaTex - riskmatrix\_plotter\_conf.yaml - Konfigurationsdatei - Risikomatrizen

## IX 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>))
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 6: Python LaTex - riskmatrix\_xy\_axis\_tuple\_matrix.yaml - Konfigurationsdatei - Risikomatrizen - X-Y-Achsen Tuples

## X 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 == 'homedir':
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 direct
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('destination_path')
55     imagename = cost_benefit_config.get('imagename')
56
57     if startpath == 'homedir':
58         directory = os.path.join(os.getcwd(), destination)
59     else: # parentdir
56         directory = os.path.join(os.path.dirname(os.getcwd()), destination)
57
58     # get the Datas as direct
59     data_path = os.path.join(directory, imagename)
60
61     # Extract the Datas
62     labels, values = zip(*cost_benefit_data.items())
63     x, y = zip(*values)
64
65     # Create Scatter-Diagram
66     plt.scatter(x, y, color=cost_benefit_config.get('scatter-point-color'))
67
68     # X-Lines
69     plt.axhline(y=cost_benefit_config.get('y-axis-line-pos'), color=
70     cost_benefit_config.get('y-axis-line-color'), linestyle=cost_benefit_config.
71     get('y-axis-line-type'), label=cost_benefit_config.get('y-axis-line-label'))
72
73     # Y-Lines
74     plt.axvline(x=cost_benefit_config.get('x-axis-line-pos'), color=
75     cost_benefit_config.get('x-axis-line-color'), linestyle=cost_benefit_config.
76     get('x-axis-line-type'), label=cost_benefit_config.get('x-axis-line-label'))
77
78     # Add Labels
79     plt.xlabel(cost_benefit_config.get('x-axis-title'))

```

```

80     plt.ylabel(cost_benefit_config.get('y-axis-title'))
81     plt.title(cost_benefit_config.get('title'))
82
83     # Labling Data Points
84     for label, x_point, y_point in zip(labels, x, y):
85         plt.text(x_point, y_point, label)
86
87     # Show Legends
88     plt.legend()
89
90     # Show Grid
91     plt.grid(True)
92
93     # Save Diagram as PNG
94     plt.savefig(data_path)
95
96 cost_benefit_config = load_configuration()
97 cost_benefit_data = get_data(cost_benefit_config)
98 cost_benefit_diagram(cost_benefit_config, cost_benefit_data)

```

Listing 7: Python LaTex - cost\_benefit\_diagram.py - Kosten-Nutzen-Diagramm

## XI            cost\_benefit\_diagram\_plotter\_conf.yaml

```

1 startpath: "parentdir"
2 desitination_path: "source/cost_benefit_diagram"
3 datafile: "cost_benefit_diagram.csv"
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 8: Python LaTex - cost\_benefit\_diagram\_plotter\_conf.yaml - Konfigurationsdatei - Kosten-Nutzen-Diagramm

## XII            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 == 'omedir':
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     # if alternative_csv_load:
37     #     with open(data_path, 'r', encoding=encoding) as file:
38     #         reader = csv.reader(file)
39     #         data = list(reader)
40     #
41     #         # panda_table_data = pd.DataFrame(data, columns=data[0])
42     #         # panda_table_data = pd.read_csv(data_path, sep=separator, decimal=decimal,
43     #         #                                     encoding=encoding, lineterminator='\n', engine='python')
44     #         # panda_table_data = pd.read_csv(data_path, sep=separator, decimal=decimal,
45     #         #                                     encoding=encoding, lineterminator='\n')
46     #         #
47     #         df_dtype = {
48     #             "Nr.": int,
```

```
45     #         "Anforderung": str,
46     #         "Beschreibung": str,
47     #         "System": str,
48     #         "Muss / Kann": str
49     }
50     #     # panda_table_data = pd.read_csv(data_path, sep=",", decimal=".",
51     #     encoding=encoding, lineterminator='\n', dtype=df_dtype)
52     #     # panda_table_data = pd.read_csv(data_path, sep=",", decimal=".",
53     #     encoding=encoding)
54     # else:
55     #     panda_table_data = pd.read_csv(data_path, sep=separator, decimal=decimal,
56     #     , encoding=encoding)
57     # panda_table_data = pd.read_csv(data_path, sep=",", decimal=".",
58     #     encoding=encoding, low_memory=False, engine='python')
59     # panda_table_data = pd.read_csv(data_path, sep=",", decimal=".",
60     #     encoding=encoding, engine='python', dtype='unicode')
61     # readed = open(data_path, 'r', encoding=encoding)
62     # panda_table_data = pd.read_csv(open(data_path, 'r', encoding=encoding), sep
63     #     =",", decimal=".",
64     #     encoding=encoding)
65     # panda_table_data = pd.read_csv(data_path, sep=",", decimal=".",
66     #     encoding="ISO-8859-1")
67     # panda_table_data = pd.read_csv(data_path, sep=",", decimal=".",
68     #     encoding=encoding, chunkszie=10)
69
70     # for chunk in pd.read_csv(data_path, sep=",", decimal=".",
71     #     encoding=encoding, chunkszie=5):
72     #     print(chunk)
73     # panda_table_data = pd.DataFrame()
74     # temp = pd.read_csv(data_path, iterator=True, sep=",", decimal=".",
75     #     encoding=encoding, chunkszie=1000)
76     # panda_table_data = pd.concat(temp, ignore_index=True)
77
78     df_dtype = {
79         "Nr.": int,
80         "Anforderung": str,
81         "Beschreibung": str,
82         "System": str,
83         "Muss / Kann": str
84     }
85     # panda_table_data = pd.read_csv(data_path, sep=",", decimal=".",
86     #     encoding=encoding, engine='python', dtype=df_dtype)
87     # panda_table_data = pd.read_csv(data_path, sep=",", decimal=".",
88     #     encoding=encoding, dtype=df_dtype)
89
90     # import dask.dataframe as dd
91     # df = dd.read_csv(data_path, sep=",", decimal=".",
92     #     encoding=encoding)
93     # panda_table_data = df
94     print(encoding)
```

```
81     panda_table_data = pd.read_csv(data_path, sep=separator, decimal=decimal,
82                                     encoding=encoding)
83     # return data
84     return panda_table_data
85
86 def create_latex_tables(panda_latex_tables_config):
87     plt_tables = panda_latex_tables_config.get('tables_inventory')
88     for table_item in plt_tables:
89         # id and filesystem informations
90         table_id = panda_latex_tables_config.get('tables').get(table_item).get('id')
91
92         isbigfile = panda_latex_tables_config.get('tables').get(table_item).get('isbigfile')
93         has_longtexts = panda_latex_tables_config.get('tables').get(table_item).get('has_longtexts')
94         if isbigfile or has_longtexts:
95             alternative_cvs_load = True
96         else:
97             alternative_cvs_load = False
98         startpath = panda_latex_tables_config.get('tables').get(table_item).get('startpath')
99         destination = panda_latex_tables_config.get('tables').get(table_item).get('destination_path')
100        tablefilename = panda_latex_tables_config.get('tables').get(table_item).get('tablefilename')
101        datafile_path = panda_latex_tables_config.get('tables').get(table_item).get('datafile_path')
102        datafile = panda_latex_tables_config.get('tables').get(table_item).get('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('separator')
109        decimal = panda_latex_tables_config.get('tables').get(table_item).get('decimal')
110
111        # column operations
112        column_operations = panda_latex_tables_config.get('tables').get(table_item).get('column_operations').get('datas')
113
114        # group by / aggregation
115        groupby_values = panda_latex_tables_config.get('tables').get(table_item).get('group_by')
116        group_by_function = panda_latex_tables_config.get('tables').get(table_item).get('group_by_function')
```

```

116     ).get('group_by_function')
117         # selected_rows = panda_latex_tables_config.get('tables').get(table_item).
118         get('selected_rows')
119         agg_funtion = panda_latex_tables_config.get('tables').get(table_item).get(
120             'agg_funtion')
121         agg_colums = panda_latex_tables_config.get('tables').get(table_item).get(
122             'agg_colums')
123         # dropping and renaming columns
124         drop_columns = panda_latex_tables_config.get('tables').get(table_item).get(
125             'drop_columns')
126         rename_columns = panda_latex_tables_config.get('tables').get(table_item).
127             get('rename_columns')
128
129         # table filtering and sorting
130         where_clausel = panda_latex_tables_config.get('tables').get(table_item).
131             get('where_clausel')
132         order_by = panda_latex_tables_config.get('tables').get(table_item).get(
133             'sorting').get('order_by')
134         sort_acending = panda_latex_tables_config.get('tables').get(table_item).
135             get('sorting').get('sort_acending')
136         sort_inplace = panda_latex_tables_config.get('tables').get(table_item).get(
137             'sorting').get('sort_inplace')
138
139         # pivot settings
140         pivot = panda_latex_tables_config.get('tables').get(table_item).get('pivot'
141             ')
142         pivot_column = panda_latex_tables_config.get('tables').get(table_item).get(
143             'pivot_columns')
144         pivot_value = panda_latex_tables_config.get('tables').get(table_item).get(
145             'pivot_values')
146
147         # pivot_table settings
148         pivot_table = panda_latex_tables_config.get('tables').get(table_item).get(
149             'pivot_table')
150         pivot_table_column = panda_latex_tables_config.get('tables').get(
151             table_item).get('pivot_table').get(
152                 'pivot_columns')
153         pivot_table_value = panda_latex_tables_config.get('tables').get(table_item
154             ).get('pivot_table').get(
155                 'pivot_values')
156         pivot_table_agg_function = panda_latex_tables_config.get('tables').get(
157             table_item).get('pivot_table').get(
158                 'pivot_agg_func')
159         pivot_table_indizes = panda_latex_tables_config.get('tables').get(
160             table_item).get('pivot_table').get(
161                 'pivot_index').get('pivot_indizes')
162         pivot_table_indizes_visible = panda_latex_tables_config.get('tables').get(
163             table_item).get('pivot_table').get(
164                 'pivot_index')

```

```

145         'pivot_index').get('pivot_indizes_visible')
146     pivot_table_rename_indizes = panda_latex_tables_config.get('tables').get(
147       table_item).get('pivot_table').get(
148         'pivot_index').get('pivot_rename_indizes')
149
150     # margins (subtotals)
151     margin = panda_latex_tables_config.get('tables').get(table_item).get(
152       'margins').get('margin')
153     margin_name = panda_latex_tables_config.get('tables').get(table_item).get(
154       'margins').get('margin_name')
155
156     # table settings
157     table_caption = panda_latex_tables_config.get('tables').get(table_item).
158       get('caption')
159     table_label = panda_latex_tables_config.get('tables').get(table_item).get(
160       'label')
161     table_style = panda_latex_tables_config.get('tables').get(table_item).get(
162       'table_styles')
163     sparse_columns = panda_latex_tables_config.get('tables').get(table_item).
164       get('table_styles').get(
165         'sparse_columns')
166     table_caption_position = panda_latex_tables_config.get('tables').get(
167       table_item).get('table_styles').get(
168         'props').get('caption-side')
169     table_position = panda_latex_tables_config.get('tables').get(table_item).
170       get('table_styles').get('props').get(
171         'position')
172     longtable = panda_latex_tables_config.get('tables').get(table_item).get(
173       'table_styles').get('props').get(
174         'longtable')
175     linebreak_columns = panda_latex_tables_config.get('tables').get(table_item).
176       get('table_styles').get('props').get(
177         'linebreak_columns')
178     resize_textwidth = panda_latex_tables_config.get('tables').get(table_item).
179       get('table_styles').get('props').get(
180         'resize_textwidth')
181
182     # get the pandas (panda data)
183     panda_table_data = get_data(startpath, destination, tablefilename,
184       datafile_path, datafile, alternative_csv_load, separator, decimal)
185
186     # filter by where clause
187     if where_clausel:
188       panda_table_data = panda_table_data.query(where_clausel)
189
190     # Drop unused columns
191     if drop_columns:
192       panda_table_data = panda_table_data.drop(columns=drop_columns)

```

```
180
181     # set aggregation functions
182     # if groupby_values and not agg_funtion and not pivot_column and not
183     # pivot_table_column:
184     if groupby_values and not (pivot_column or (pivot_table_column or
185     pivot_table_value or pivot_table_indizes)):
186         match group_by_function:
187             case 'max':
188                 panda_table_data = panda_table_data.groupby(groupby_values,
189                 as_index=False).max()
190             case 'min':
191                 panda_table_data = panda_table_data.groupby(groupby_values,
192                 as_index=False).min()
193             case 'head':
194                 panda_table_data = panda_table_data.groupby(groupby_values,
195                 as_index=False).head()
196             case 'sum':
197                 panda_table_data = panda_table_data.groupby(groupby_values,
198                 as_index=False).sum()
199             case 'mean':
200                 panda_table_data = panda_table_data.groupby(groupby_values,
201                 as_index=False).mean()
202
203     else:
204         panda_table_data = panda_table_data
205
206     # pivot if pivot is selected
207     if pivot_table_column or pivot_table_value or pivot_table_indizes:
208         if type(pivot_table_agg_function) is list:
209             agg_tuple = tuple(pivot_table_agg_function)
210             panda_table_data = pd.pivot_table(panda_table_data, index=
211                 pivot_table_indizes,
212                                         columns=pivot_table_column,
213                                         values=pivot_table_value,
214                                         aggfunc=agg_tuple, margins=
215                                         margin, margins_name=margin_name)
216         elif type(pivot_table_agg_function) is dict:
217             panda_table_data = pd.pivot_table(panda_table_data, index=
218                 pivot_table_indizes,
219                                         columns=pivot_table_column,
220                                         values=pivot_table_value,
221                                         aggfunc=pivot_table_agg_function,
222                                         margins=margin, margins_name=
223                                         margin_name)
224         else:
225             panda_table_data = pd.pivot_table(panda_table_data, index=
226                 pivot_table_indizes,
227                                         columns=pivot_table_column,
228                                         values=pivot_table_value,
```

```

213                                     aggfunc=pivot_table_agg_function
214                                     , margins=margin,
215                                     margins_name=margin_name)
216
217     # set column operations
218     if column_operations:
219         for column_ops in column_operations:
220             operation_function = panda_latex_tables_config.get('tables').get(
221                 table_item).get('column_operations').get('operations').get(column_ops).get(
222                 'operation_function')
223             operation_columns = panda_latex_tables_config.get('tables').get(
224                 table_item).get('column_operations').get('operations').get(column_ops).get(
225                 'columns')
226             operation_axis = panda_latex_tables_config.get('tables').get(
227                 table_item).get('column_operations').get('operations').get(column_ops).get(
228                 'axis_number')
229             match operation_function:
230                 case 'max':
231                     panda_table_data[column_ops] = panda_table_data[
232                         operation_columns].max()
233                 case 'min':
234                     panda_table_data[column_ops] = panda_table_data[
235                         operation_columns].min()
236                 case 'head':
237                     panda_table_data[column_ops] = panda_table_data[
238                         operation_columns].head()
239                 case 'sum':
240                     panda_table_data[column_ops] = panda_table_data[
241                         operation_columns].sum(axis=operation_axis)
242                 case 'mean':
243                     panda_table_data[column_ops] = panda_table_data[
244                         operation_columns].mean()
245                 case 'diff':
246                     panda_table_data[column_ops] = panda_table_data[
247                         operation_columns[1]] - panda_table_data[operation_columns[0]]
248
249
250     # order by
251     if order_by:
252         panda_table_data.sort_values(by=order_by, inplace=sort_inplace,
253                                     ascending=sort_acending)
254
255     # rename columns
256     if rename_columns:
257         panda_table_data = panda_table_data.rename(columns=rename_columns)
258
259     # rename indices
260     if pivot_table_rename_index:

```

```

247         panda_table_data = panda_table_data.rename_axis(index=
pivot_table_rename_index)
248
249     # frame carriage return columns in subtable
250     if linebreak_columns:
251         for lbr_column in linebreak_columns:
252             panda_table_data[lbr_column] = "\\begin{tabular}[c]{@{}l@{}}" +
panda_table_data[lbr_column].astype(str) + "\\end{tabular}"
253             # convert python panda to latex table
254             latex_table = panda_table_data.to_latex(header=True, bold_rows=False,
longtable=longtable,
255                                         sparsify=sparse_columns, label=
table_label, caption=table_caption,
256                                         position=table_position, na_rep='',
257                                         index=pivot_table_index_visible)
258
258     # textwidth resize
259     if resize_textwidth:
260         with open(tablefile, 'w') as wrlt:
261             wrlt.write(latex_table)
262
263         with open(tablefile) as file:
264             lines = file.readlines()
265
266         # replace table with resize
267         resize_line_nr = 0
268         resize_line = ""
269         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                     latex_table = latex_table.replace(resize_line, line_table_resize)

```

```
289
290     # replace table end with bracket
291     resize_line_nr = 0
292     resize_line = ""
293     if longtable:
294         table_type = '\\end{longtable}'
295     else:
296         table_type = '\\end{table}'
297
298     for number, line in enumerate(lines, 1):
299
300         # Condition true if the key exists in the line
301         # If true then display the line number
302         if table_type in line:
303             # print(f'{key} is at line {number}')
304             resize_line_nr = number
305             resize_line = line
306
307         line_table_resize = "}" + "\n" + resize_line
308         latex_table = latex_table.replace(resize_line, line_table_resize)
309
310     # caption below is not supported yet (pandas 2.2)
311     # replace caption and replace table end with the caption line and table
312     end
313     if table_caption_position == 'below':
314         caption_label = "\\caption{" + table_caption + "}" "\\label{" +
315         table_label + "}" "\\\""
316         caption_label_nbr = "\\caption{" + table_caption + "}" "\\label{" +
317         table_label + "}"
318         caption_only = "\\caption{" + table_caption + "}" "\\\""
319         caption_only_nbr = "\\caption{" + table_caption + "}"
320         label_only = "\\label{" + table_label + "}" "\\\""
321         label_only_nbr = "\\label{" + table_label + "}"
322         latex_table = latex_table.replace(caption_label, '')
323         latex_table = latex_table.replace(caption_only, '')
324         latex_table = latex_table.replace(label_only, '')
325         latex_table = latex_table.replace(caption_label_nbr, '')
326         latex_table = latex_table.replace(caption_only_nbr, '')
327         latex_table = latex_table.replace(label_only_nbr, '')
328
329     if longtable:
330         table_string = '\\end{longtable}'
331         new_caption = caption_label_nbr + "\n" + table_string
332         latex_table = latex_table.replace(table_string, new_caption)
333     else:
334         table_string = '\\end{table}'
335         new_caption = caption_label_nbr + "\n" + table_string
336         latex_table = latex_table.replace(table_string, new_caption)
```

```

334
335
336     # write latex table to filesystem
337     with open(tablefile, 'w') as wrlt:
338         wrlt.write(latex_table)
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 9: Python LaTex - pandas\_dataframe\_to\_latex\_table.py CSV - LaTex Tabelle

### XIII            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"
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"
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_clausel:
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:
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_indexes_visible:
133     pivot_rename_indexes:
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_clauses:
143     sorting:
144     order_by:
145     - "rdbms"
146     sort_ascending: 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
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 üaufgeschlüsselt"
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"
205             - "rdbms"
206         pivot_indizes_visible: True
207         pivot_rename_indizes:
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_clausel:
224         sorting:
```

```
225     order_by:
226     sort_acending: 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"
253     tablefilename: "anforderungskatalog.tex"
254     decimal_format:
255     group_by:
256     group_by_function:
257     agg_funtion:
258     agg_colums:
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_indexes:  
274     pivot_columns:  
275     pivot_values:  
276     pivot_agg_func:  
277     rename_columns:  
278     where_clauses:  
279     sorting:  
280         order_by:  
281             - "Nr."  
282         sort_ascending: 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:  
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_colums:  
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:
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_function:
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"
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_indizes_visible:
410   pivot_rename_indizes:
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"
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_columns:
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_indexizes_visible: False  
468         pivot_rename_indexizes:  
469         pivot_columns:  
470             pivot_values:  
471             pivot_agg_func:  
472             rename_columns:  
473             where_clausel:  
474             sorting:  
475                 order_by:  
476                     - "Server"  
477                     - "Typ"  
478             sort_ascending: 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  
493             dependencis:  
494                 id: "dependencis"  
495                 isbigfile:  
496                 has_longtexts: False  
497                 separator: ";"  
498                 decimal: "."  
499                 caption: "Ähnlichkeiten"  
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_function:  
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_indexes_visible: False
525     pivot_rename_indexes:
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"
541     props:
542     caption-side: "below"
543     position: "H"
544     sparse_columns: True
545     longtable: False
546     resize_textwidth: True
547     linebreak_columns:
548     - "Ähnlichkeit"
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_indizes_visible: False
587     pivot_rename_indizes:
588     pivot_columns:
589     pivot_values:
590     pivot_agg_func:
591     rename_columns:
592     where_clausel: "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:
637             axis_number:
638             columns:
639             pivot:
640               pivot_columns:
641               pivot_values:
642               pivot_table:
643               pivot_index:
644               pivot_indizes_visible: False
645               pivot_rename_indizes:
646               pivot_columns:
647               pivot_values:
648               pivot_agg_func:
649               rename_columns:
650               where_clause1: "hide_state == 2"
651               sorting:
652                 order_by:
653                 - "Nr."
654                 sort_acending: 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:
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_indexes_visible: False
702               pivot_rename_indexes:
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_ascending: 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: "."  
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_indexizes_visible: False  
759     pivot_rename_indexizes:  
760 pivot_columns:  
761     pivot_values:  
762     pivot_agg_func:  
763 rename_columns:  
764 where_clauses:  
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:  
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_function:  
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:
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: "äFachgesprche - 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_function:
860 agg_columns:
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_indexes_visible: False
875 pivot_rename_indexes:
876 pivot_columns:
877 pivot_values:
878 pivot_agg_func:
879 rename_columns:
880 where_clausel:
881 sorting:
882 order_by:
883 - "äFachgesprch"
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_function:
921     agg_columns:
922     drop_columns:
923     column_operations:
924     datas:
925     operations:
926     dauer_summe:
927     operation_function:
928     axis_number:
929     columns:
930     pivot:
931     pivot_columns:
932     pivot_values:
933     pivot_table:
934     pivot_index:
935     pivot_indizes_visible: False
936     pivot_rename_indizes:
937     pivot_columns:
938     pivot_values:
939     pivot_agg_func:
940     rename_columns:
941     where_clausel:
942     sorting:
943     order_by:
944     sort_acending: True
```

```
945     sort_inplace: True
946     margins:
947       margin: False
948       margin_name:
949     table_styles:
950       selector: "caption"
951     props:
952       caption-side: "below"
953       position: "H"
954     sparse_columns: True
955     longtable: False
956     resize_textwidth: True
957     linebreak_columns:
958     table_header: True
```

Listing 10: Python LaTex - csv\_to\_latex\_diplomarbeit.yaml - Konfigurationsdatei - CSV - LaTex-Tabelle