



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

Abkürzungen	4
1 Einleitung	1
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
1.6 Vorgehensweise und Methoden	24
1.7 Projektmanagement	24
1.7.1 Projektcontrolling	25
1.7.2 GANTT-Diagramm	26
1.8 Status-Reports	28
1.8.1 Initialer Statusbericht	28
1.8.2 Zweiter Statusbericht	29
1.9 Expertengespräche	30
2 Umsetzung	31
2.1 Evaluation	31
2.1.1 Exkurs Architektur	31
2.1.2 Erheben und Gewichten der Anforderungen	36
2.1.3 Testziele erarbeiten	49
2.1.4 PostgreSQL Benchmarking	49
2.1.5 Analyse gängiger PostgreSQL HA Cluster Lösungen	50
2.1.6 Vorauswahl	70
2.1.7 Installation verschiedener Lösungen	71
2.1.8 Gegenüberstellung der Lösungen	75
2.1.9 Entscheid	75
2.2 Aufbau und Implementation Testsystem	75
2.2.1 Bereitstellen der Grundinfrastruktur	75

2.2.2	Installation und Konfiguration PostgreSQL HA Cluster	75
2.2.3	Technical Review der Umgebung	75
2.3	Testing	75
2.3.1	Testing	75
2.3.2	Protokollierung	75
2.3.3	Review und Auswertung	76
2.4	Troubleshooting und Lösungsfindung	76
3	Resultate	77
3.1	Zielüberprüfung	77
3.2	Schlussfolgerung	77
3.3	Weiteres Vorgehen / offene Arbeiten	77
3.4	Persönliches Fazit	77
Abbildungsverzeichnis		78
Tabellenverzeichnis		80
Listings		81
Literatur		82
Glossar		87
Anhang		i
I	Arbeitsrapport	i
II	Protokoll - Fachgespräche	ii
III	Kommentare / Anmerkungen	iii
IV	zotero.py	iv
V	riskmatrix.py	x
VI	cost_benefit_diagram.py	xv
VII	csscatter_plotter_conf.yaml	xvii
VIII	pandas_dataframe_to_latex_table.py	xviii
IX	csv_to_latex_diplomarbeit.yaml	xxvii

Abkürzungen

ICT	information and communications technology
ibW	ibW Höhere Fachschule Südostschweiz
KSGR	Kantonsspital Graubünden
RDBMS	Relational Database Management System
DBMS	Database Management System
k8s	Kubernetes
HPE	Hewlett Packard Enterprise
HP-UX	Hewlett Packard UNIX
SAP	Systemanalyse Programmierung
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
PKI	Private Key Infrastructure

Diplomarbeit



- 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

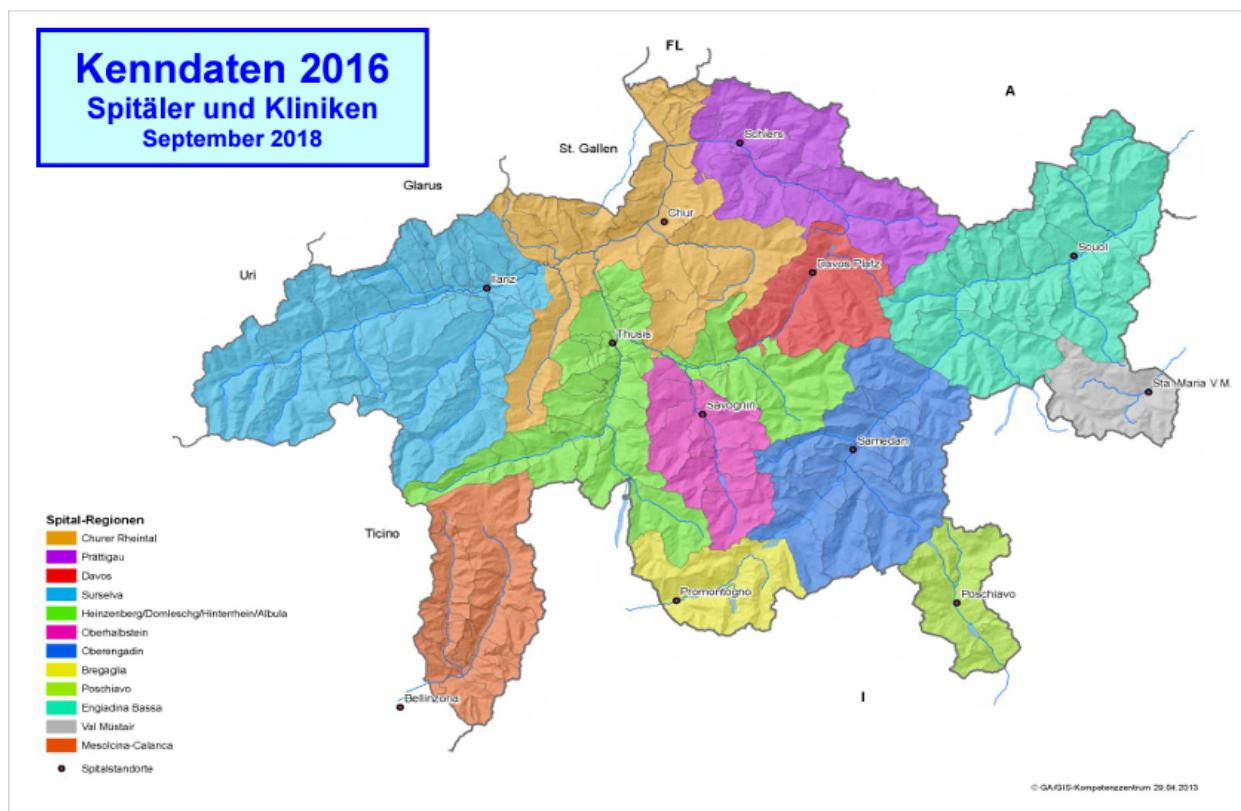


Abbildung 1.1: Spitalregionen Kanton Graubünden[32]

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[57]

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

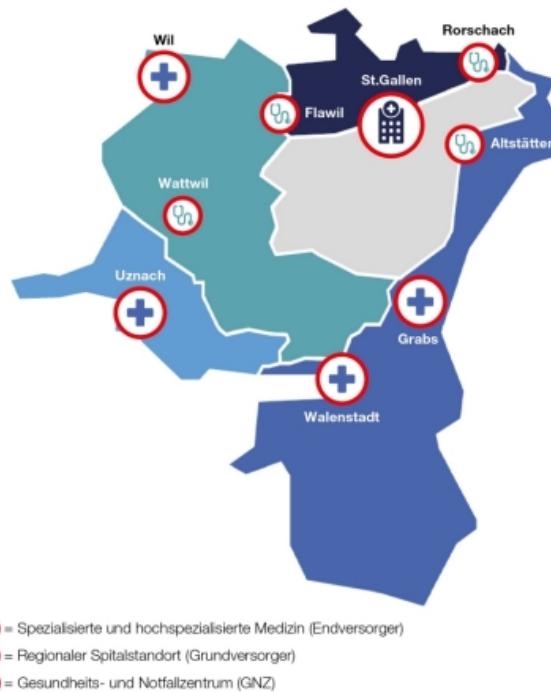


Abbildung 1.3: Spitalregionen / Spitalstrategie Kanton St. Gallen[26]

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

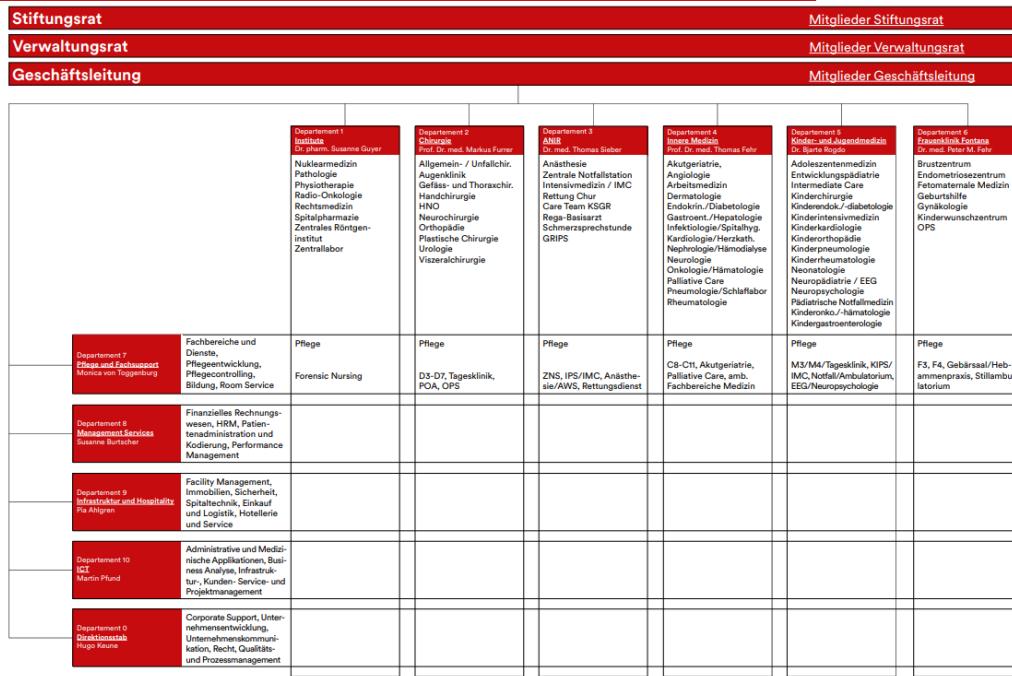
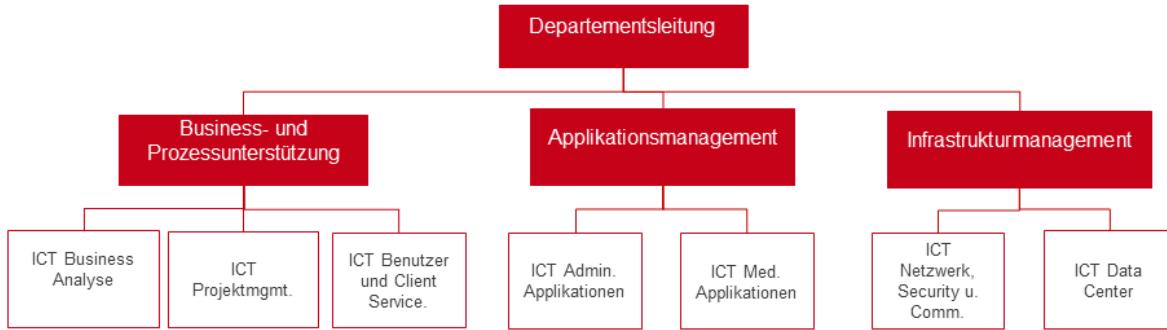


Abbildung 1.4: Organigramm Kantonsspital Graubünden

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*[23] 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:

DBMS	Datenbankmodell	Inventarisiert	Kommentar
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
Gesamtergebnis		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 bemerkern. 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

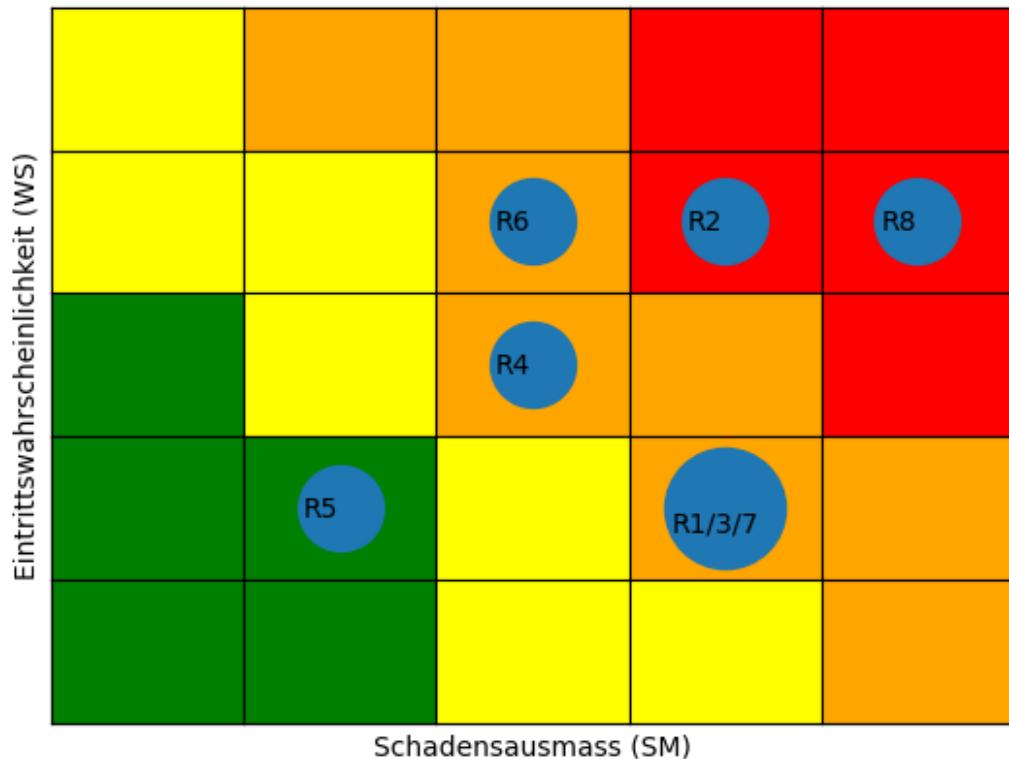


Abbildung 1.6: Risiken bestehende Lösung

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:

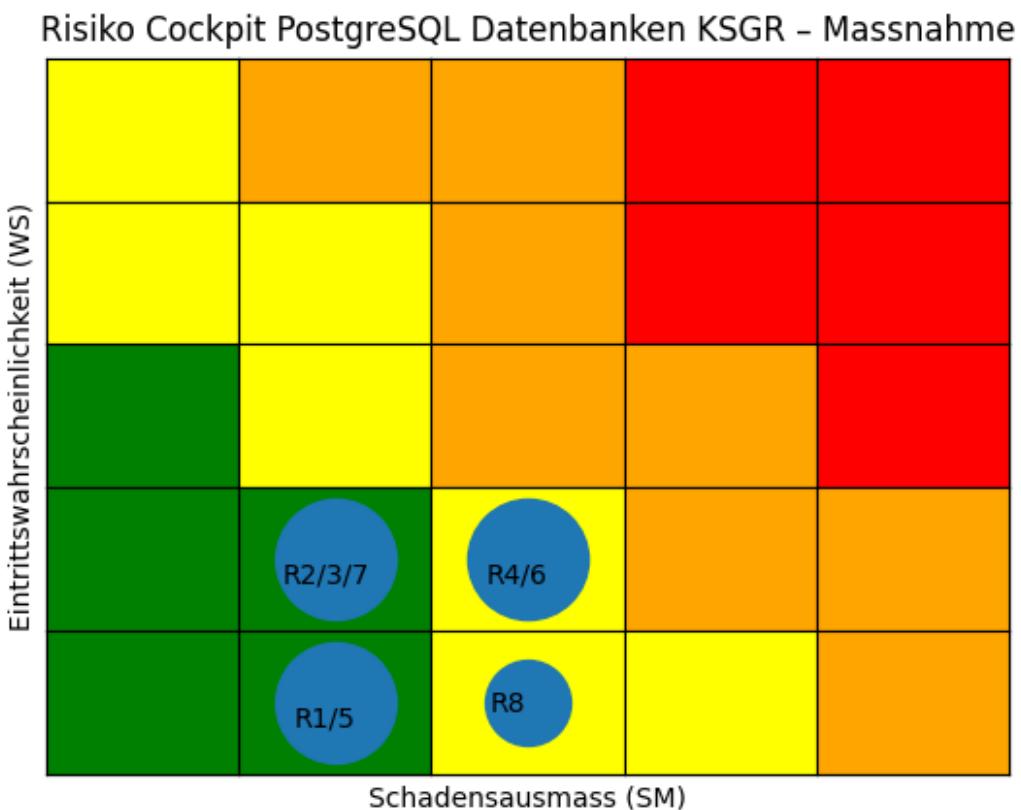


Abbildung 1.7: Risiken bestehende Lösung mit Massnahmen

1.2 Zieldefinition

Das administrieren einer PostgreSQL Datenbank umfasst i.d.R. [41, 47] 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[20]. Diese müssen regelmäßig ausgelesen werden	Tief
7	Health Checks und Aktionen (Maintenance)	Regelmäßig muss die Gesundheit der DBs überprüft werden, etwa ob Tabellen und/oder Indizes sich aufgeblättert haben oder ob Locks vorhanden sind[2]. Während der Hauptarbeitszeit muss dies mindestens alle 90 Minuten geprüft und ggf. reagiert werden.	Hoch
8	Housekeeping	Mit Housekeeping Jobs werden regelmäßig Trace- und Alertlogfiles aufgeräumt, um Platz auf den Disken zu sparen aber auch um die Übersichtlichkeit zu wahren.	Mittel
9	Verwalten von DB Objekten	Regelmäßig müssen DB Objekte wie Datenbanken, Tabellen, Trigger, Views etc. angepasst oder erstellt werden. Dies richtet sich nach den Bedürfnissen der Kunden resp. deren Applikationen.	Tief
10	User Management	Die Zugriffe der User müssen überwacht, angepasst, erfasst oder gesperrt werden. Auch diese Aufgabe richtet sich nach den Bedürfnissen der Kunden.	Tief

Tabelle 1.5: Administrative Aufgaben

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

Nr.	Aufgabe	Wichtigkeit	Zu automatisierender Task	Priorität	Muss / Kann	Spätester Termin
1	Failover	Hoch	Automatisierter Failover auf mindestens einen Sekundären DB-Node	1	Muss	Abgabe
2	Failover Restore	Hoch	Sobald der Primäre DB-Node wieder vorhanden ist, muss automatisch auf den Primären DB-Node zurückgesetzt werden. Das Filesystem muss beim erreichen von 95% Usage automatisiert vergrössert werden.	1	Muss	
3	Filesystem Management	Hoch	Die Vergrösserung muss anhand der Wachstumsrate (die mittels Linux Commands zu ermitteln ist), vergrössert werden	4	Kann	
4	Monitoring	Mittel	Der Status der Clusterumgebung und der Replikation muss im PRTG überwacht werden	2	Muss	
5	Statistiken / Cleanup Jobs justieren	Mittel	Regelmässig müssen die Parameter für den AUTOVACUUM Job berechnet werden und das Configfile postgresql.conf automatisch angepasst werden Es gibt SQL Abfragen, mit dem fehlende Indizes ermittelt werden können. Diese Indizes sollen automatisiert erstellt werden.	2	Muss	
6	SQL optimierungen	Tief	Im gleichen Zug sollen aber auch Indizes, welche nicht verwendet werden, entfernt werden. Sie tragen nicht nur nichts zu performanteren Abfragen bei sondern beziehen unnötige Ressourcen bei Datenmanipulationen[20]. Tabellen und Indizes können sich aufblähen (bloated table / bloated index)	2	Kann	
7	Health Checks und Aktionen (Maintenance)	Hoch	Ist ein Index aufgebläht, kann dies mittels eines REINDEX mit geringem Impact auf die Datenbank gelöst werden[2].	2	Muss	
8	Housekeeping	Mittel	Log Rotation muss aktiviert werden und alte Logs regelmässig gelöscht werden.	3	Kann	
9	Verwalten von DB Objekten	Tief	Keine automatisierung möglich	5		
10	User Management	Tief	Regelmässige Reports sollen User aufzeigen, die seit mehr als einer Woche nicht mehr aktiv waren.	4	Kann	

Tabelle 1.6: Automatisierung Administrativer Aufgaben

Mit der Arbeit sollen folgende Ergebnisse und Resultate erzielt werden:

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

Daraus ergeben sich folgende Ziele:

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

Tabelle 1.7: Ziele

1.3 Abgrenzungen

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

	Produkt	Beschreibung
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[71] folgende Abgrenzungen:

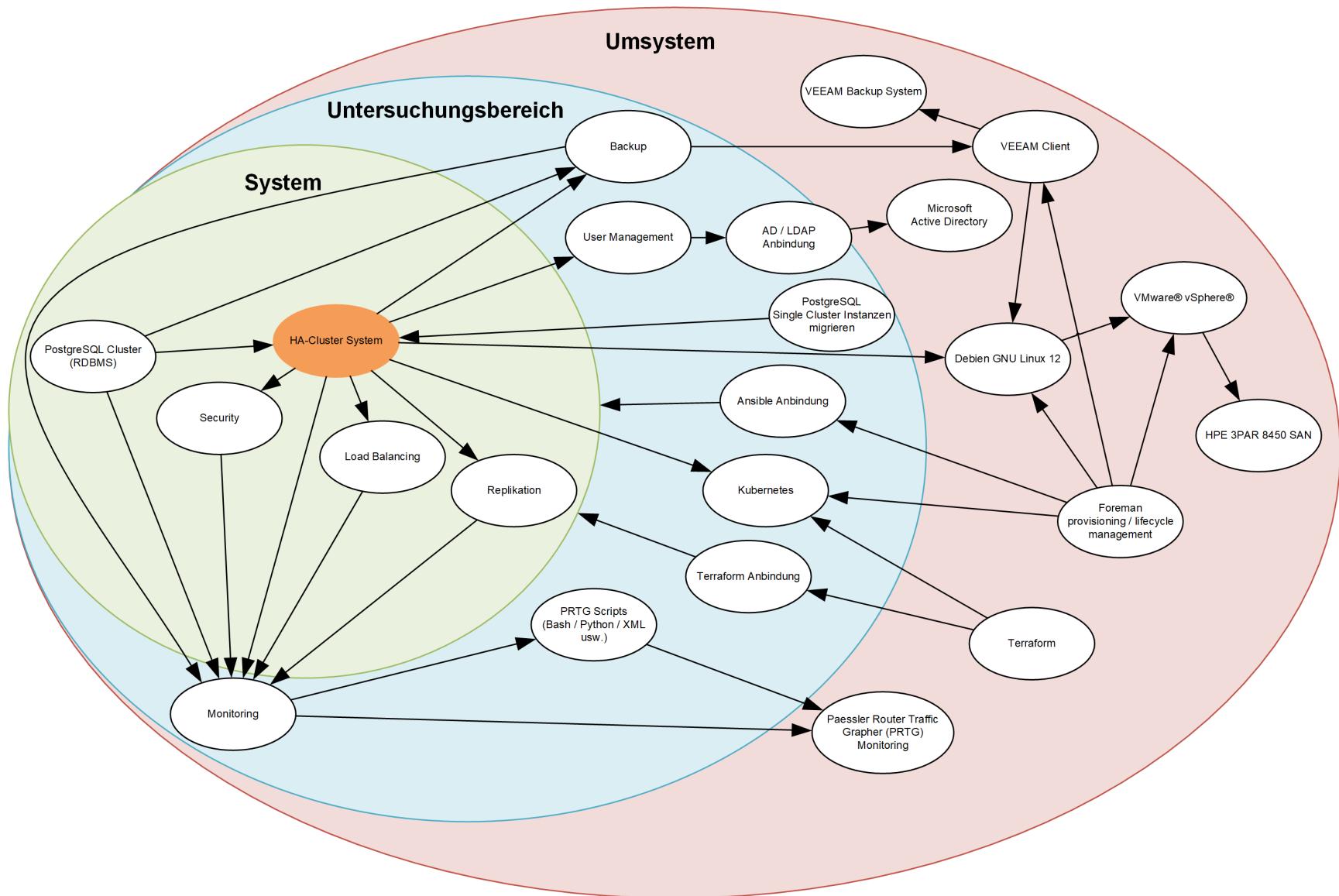


Abbildung 1.8: Systemabgrenzung

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
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.	Log Management und das SIAM werden abgelöst. Die Ausschreibung ist erfolgt	Die neue Log Management Plattform ist noch nicht betriebsbereit	Log Retention muss stark erhöht werden. 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 Teilsicherungen 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.	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.

Tabelle 1.9: Abhängigkeiten

1.5 Risikomanagement

Aus den Abhängigkeiten heraus wurden folgende Risiken identifiziert:

Identifikation			Abschätzung	Behandlung		Zielwert	Massnahme
ID	Risiko	Beschreibung / Ursache		WS	SM		
1	Fehlende Ressourcen	Viele parallele Projekte, Aufträge und der Tagesbetrieb	Ressourcen während der Diplomarbeit sind knapp bemessen	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	Das Projekt wird parallel zur Diplomarbeit sehr viele Ressourcen und Aufmerksamkeit binden	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	Bei einem Event, ausgelöst durch das Alter der HP-UX Plattform, der DELL NetWorker / Data Domain Umgebung oder dem SAN Storage, kann der ganze Betrieb zum erliegen kommen und entsprechend viele Ressourcen aufgrund der Kritikalität binden	4	4	Ja	3 3 Monitoring vorgängig ausbauen und Massnahmen definieren
4	Schwächen beim Selbstmanagement und in der Selbstoprganisation	Selbstmanagement und Organisation ist nicht meine Stärke	Das Projekt verzettelt sich, Zeit geht verloren. Auch eine Folge könnte der Scope Verlust sein	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	Verzettelung und Zeitverlust bis hin zu scheitern	3	4	Ja	2 3 Ziele klar definieren
6	Scope Creep	Der Umfang kann stark steigen wenn Ziele nicht genau genug definiert wurden	Zeitverlust bis hin zu Scheitern des Projekts	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.	Logs müssen länger auf dem System selber vor gehalten werden. Zudem müssen ggf. eigene Massnahmen zum Auslesen von Logs getroffen werden	4	1	Nein	
8	Foreman nicht betriebsbereit	Dadurch besteht das Risiko, dass sie nicht betriebsbereit zum Start der Diplomarbeit ist	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

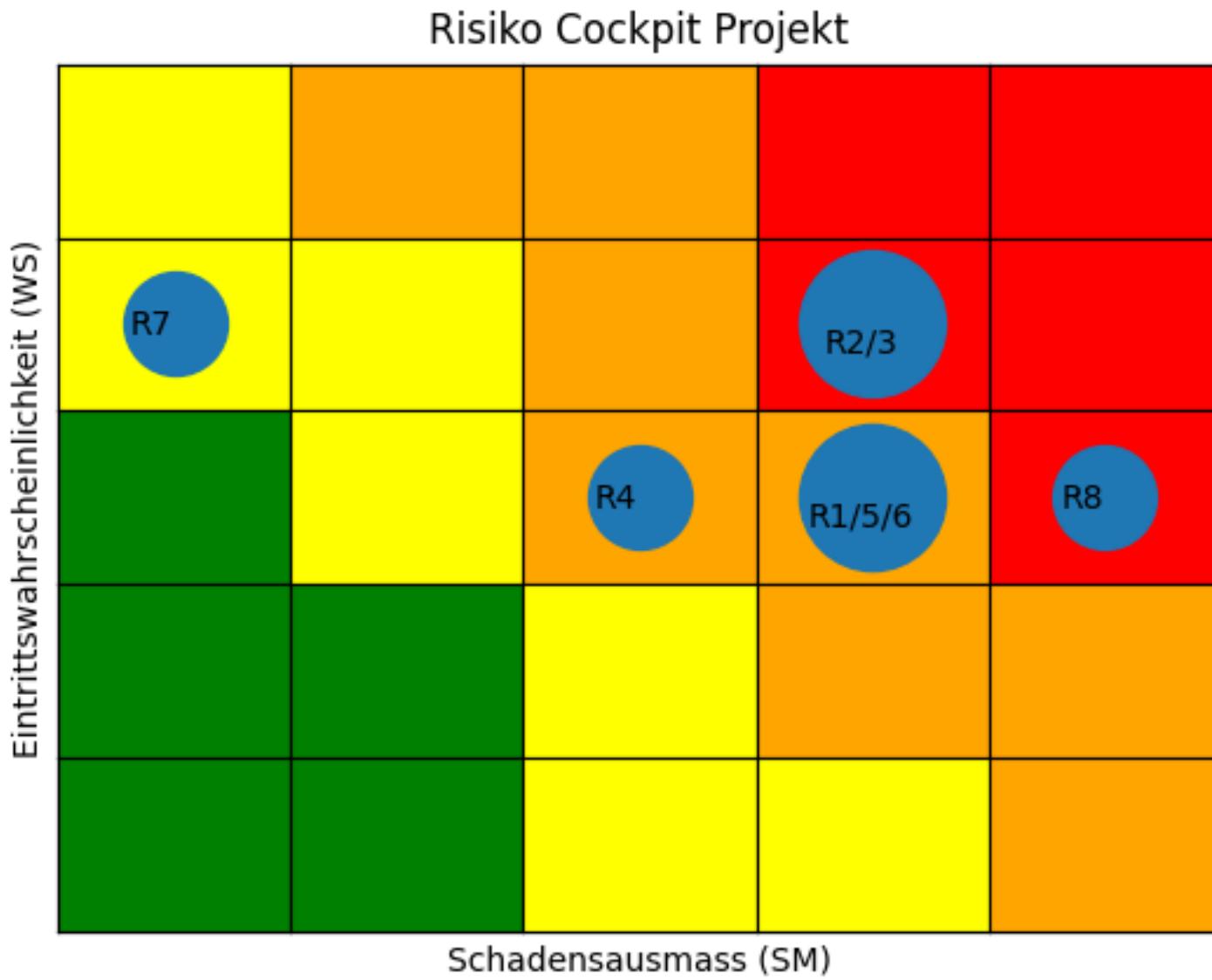


Abbildung 1.9: Projektrisiken

Mit den entsprechenden Massnahmen können die Risiken gesenkt werden:

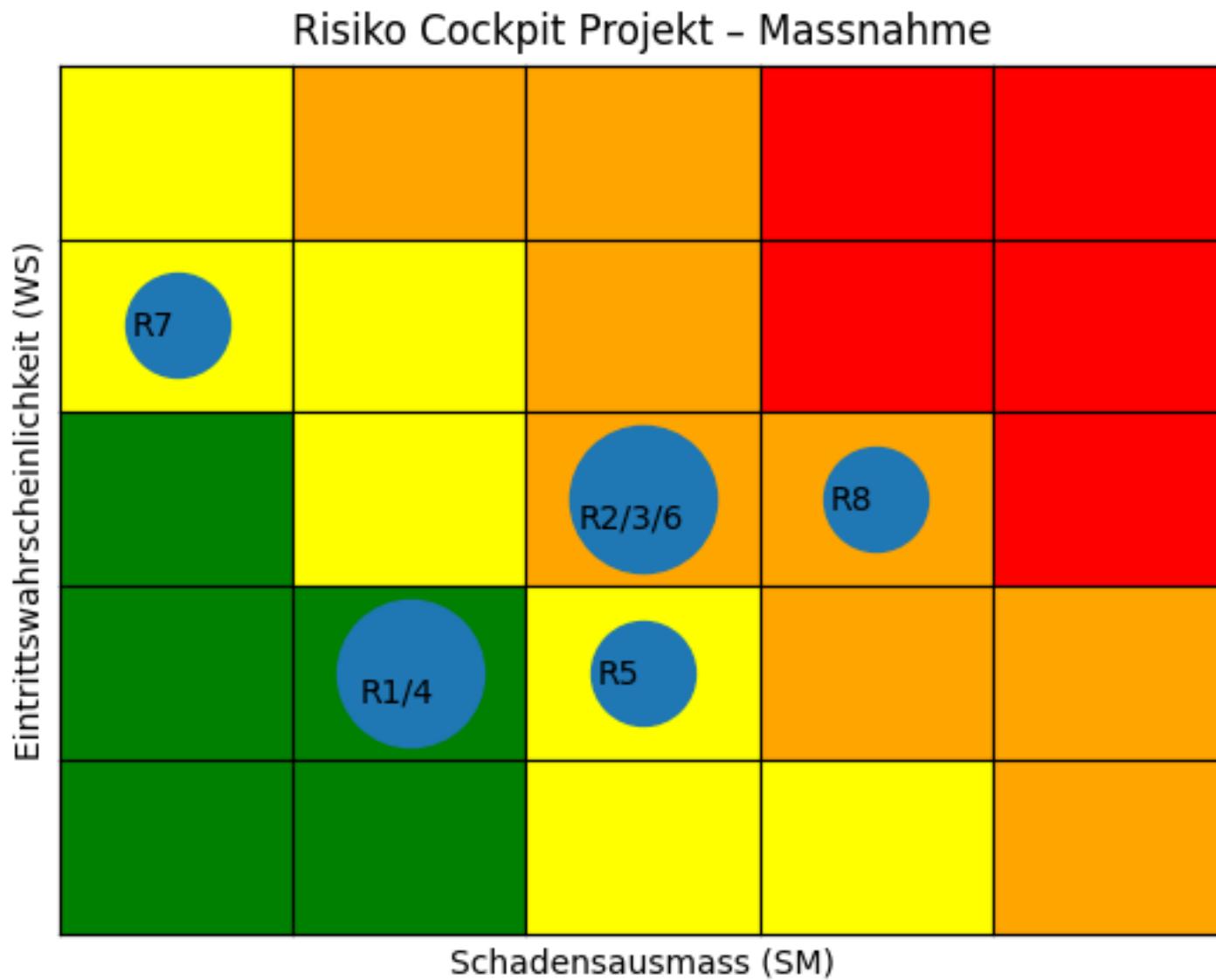


Abbildung 1.10: Projektrisiken mit Massnahmen

1.6 Vorgehensweise und Methoden

1.7 Projektmanagement

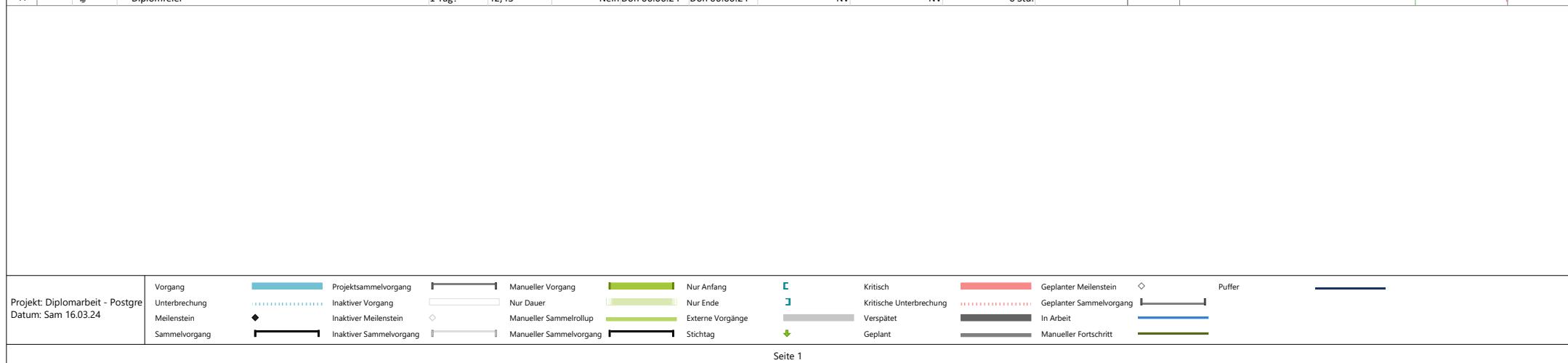
1.7.1 Projektcontrolling

Phase	Subphase	Dauer [h]	Geplante Dauer [h]	Verbleibende Zeit [h]
0 Dokumentation	-	19.0	80	61.0
1 Evaluation	Analyse PostgreSQL HA Cluster Lösungen	9.5	16	6.5
2 Evaluation	Anorderungskatalog	4.5	16	11.5
3 Evaluation	Vorbereitung Benchmarking	1.5	4	2.5

Tabelle 1.11: Projektcontrolling

1.7.2

GANNT-Diagramm



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
Gesamtprojekt			
Zeitplanung	■	↓	Projekt ist umfangreich und hat viele Teilspekte, 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
Tätigkeiten vergangene Berichtsperiode		Tätigkeiten nächste Berichtsperiode	
<ul style="list-style-type: none"> - Dokumentenstruktur erstellt - Projektplanung erstellt - Versnissage vorbereitet - Statusbericht erstellt 		<ul style="list-style-type: none"> - Anforderungskatalog erarbeiten - Vorbereitung Benchmarking 	
# 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	
Kostenübersicht		Abhängigkeiten zu anderen Projekten	
Verfügbare Finanzen bis Ende Projekt: nachr. + 200k = 24'000 CHF		Erneuerung HP UX Plattform 6002201 KSGR Provisioning System (KPS) -->Foreman Umgebung	
Bemerkungen / Informationen		Massnahmen Ressourcen reservieren Massnahmen ergreifen um die manuelle Installation so effizient wie möglich zu gestalten	
Eingereicht	Geprüft	Bemerkungen/Auftrag PMO	
PL:	PMO:		
Datum:	Datum:		
# erledigte Lieferobjekte (inkl. allfällige Links)			

Tabelle 1.12: 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 Graber		PMA	-
Status	Ampel	Tendenz	Begründung
Gesamtprojekt			
Zeitplanung	■	↓	In Verzug. Grossprojekt Erneuerung HP UX Plattform nimmt viel Zeit in Anschlag. 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			
- Anforderungskatalog erstellt - Parallel dokumentiert		Tätigkeiten nächste Berichtsperiode - Analyse der PostgreSQL HA Clusterlösungen abgeschlossen - Benchmarking abgeschlossen - Variantenentscheid getroffen - Basisystem für Testsystem aufgebaut	
# 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 Selbstoprganisation	●	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	●		
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			
Eingereicht	Geprüft	Bemerkungen/Auftrag PMO	
PL:	PMO:		
Datum:	Datum:		
# erledigte Lieferobjekte (inkl. allfällige Links)			
LO-001 Anforderungskatalog			

Tabelle 1.13: 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		Norman Süsstrunk	-	Michael Graber	

Tabelle 1.14: 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

2.1.1.3 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

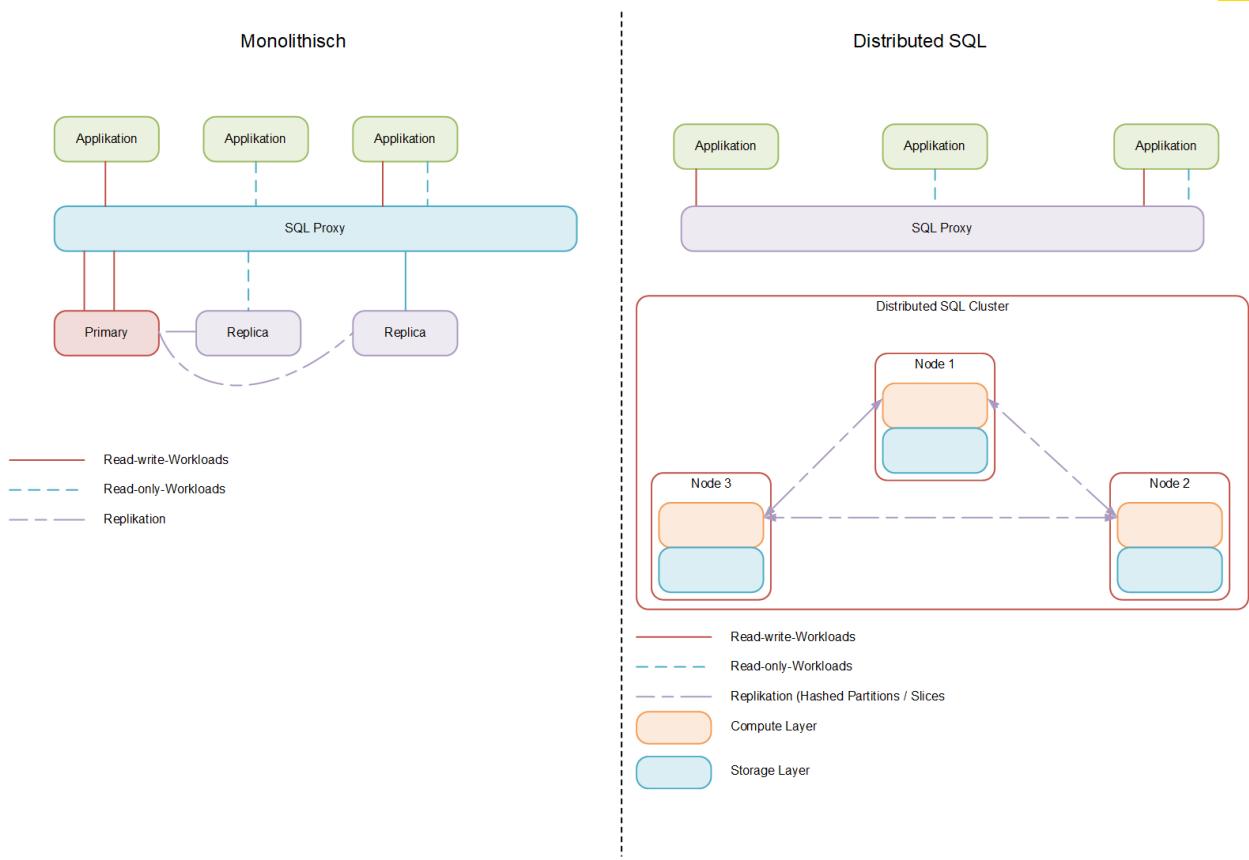


Abbildung 2.1: Monolithische vs. verteilte SQL Systeme

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[45]. Dabei wird zwischen den folgenden beiden Replikationen unterschieden:

Synchrone Replikation

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

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

Asynchrone Replikation

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

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 [43, 55, 38] 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[28].

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 2.2: 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[40]:

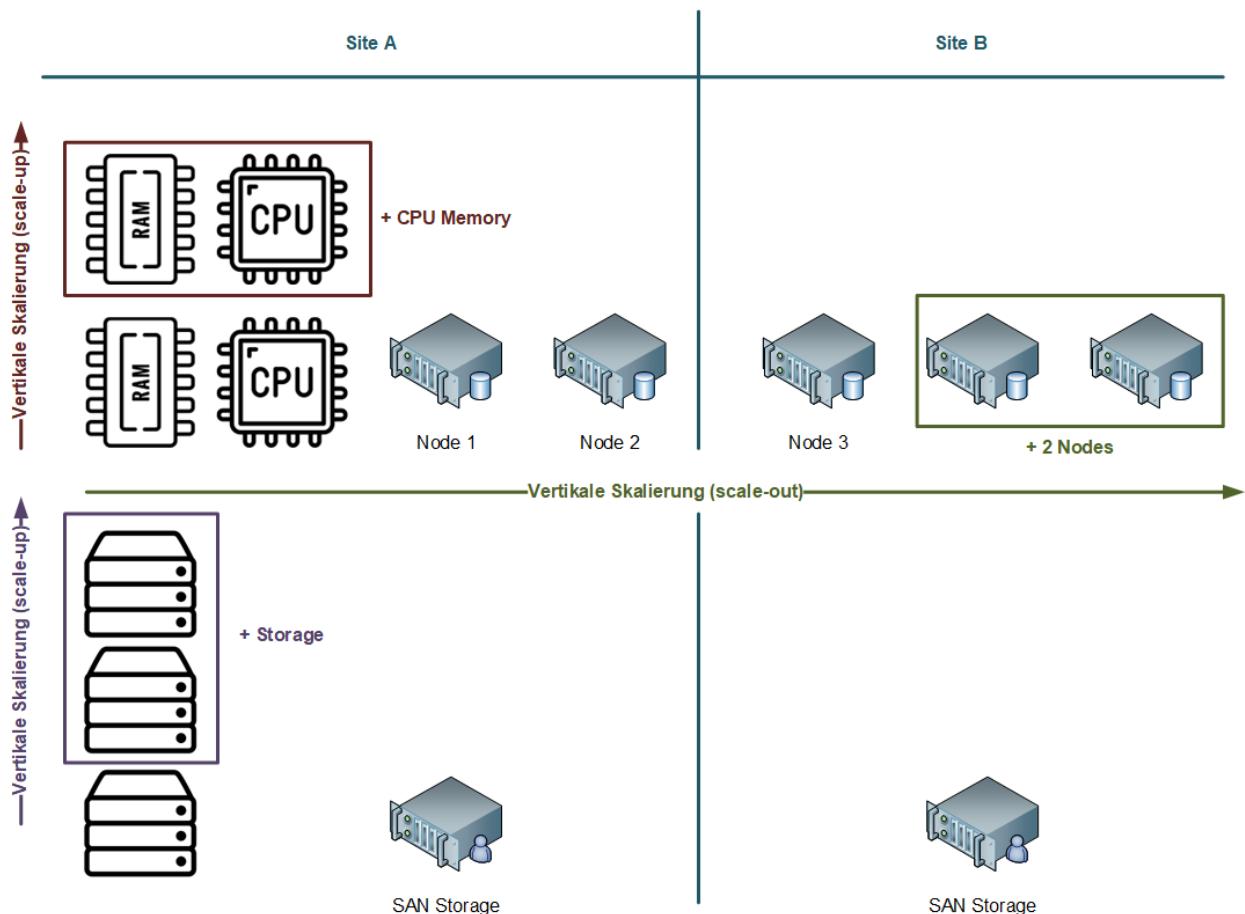


Abbildung 2.3: Datenbankskalierung

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 Replikas gesendet und Committed werden. Dieser Wert ist bei Synchrone Replikation entscheidend da Commits auf allen Replicas abgesetzt sein müssen.	Beides	MUSS
16	Sharding	Datenschutz- und integrität	Die Datenkonsistenz und Datenintegrität auf den Shards muss sichergestellt werden	Distributed SQL	MUSS
17	Sharding	Schutz vor Datenverlust	Die Synchronisation der Shards muss sicherstellen, dass es zu keinem Datenverlust kommt	Distributed SQL	MUSS
18	Quorum	Quorum-System vorhanden	Das Clustersystem muss über ein Quorum-System besitzen	Beides	MUSS
19	Quorum	Robustheit	Das Quorum des Clustersystems muss robust genug sein, um eine Split-Brain-Situation zu verhindern	Beides	MUSS
20	Connection		Das Clustersystem muss sicherstellen, dass eine Applikation ohne Entwicklungsaufwand mittels dem PostgreSQL Wired Connector zugreifen kann	Beides	MUSS
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															
100	120																																	

Legende

Eingabefelder

Zellbezüge

berechnete Felder

Abbildung 2.4: 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
11	6					

Legende

Eingabefelder

Zellbezüge

berechnete Felder

Abbildung 2.5: Präferenzmatrix - Failover

Beim Switchover wurde die Gewichtung wie folgt aufgeteilt:

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

Legende

Eingabefelder

Zellbezüge

berechnete Felder

Abbildung 2.6: 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 2.7: 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
8	6					

Legende

Eingabefelder

Zellbezüge

berechnete Felder

Abbildung 2.8: 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 2.9: 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				

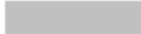
Legende Eingabefelder Zellbezüge berechnete Felder

Abbildung 2.10: 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
3	6					

Legende

Eingabefelder

Zellbezüge

berechnete Felder

Abbildung 2.11: 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 2.12: 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
6	10						

Legende

Eingabefelder

Zellbezüge

berechnete Felder

Abbildung 2.13: Präferenzmatrix - Performance

2.1.3 Testziele erarbeiten

2.1.4 PostgreSQL Benchmarking

2.1.4.1 pgBench - Basis-Benchmarking

PostgreSQL bietet ein Benchmarking-Tool,[36, 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 [14, 12, 70, 1] sind dies folgende Anforderungen:

pgGather

Mit pgGather [42] 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 [66] 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 [54].

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[35].

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[56]. 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[52, 33]:

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 [19] 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 Replikation

2.1.5.4.2 Replikation

2.1.5.4.3 Proxy

pg_auto_failover benötigt einen HAProxy, um Load Balancing usw. [9]

2.1.5.4.4 API / Skripte

2.1.5.4.5 Architektur

Die Dokumentation von pg_auto_failover [3] zeigt auf, wie der Failover funktioniert:

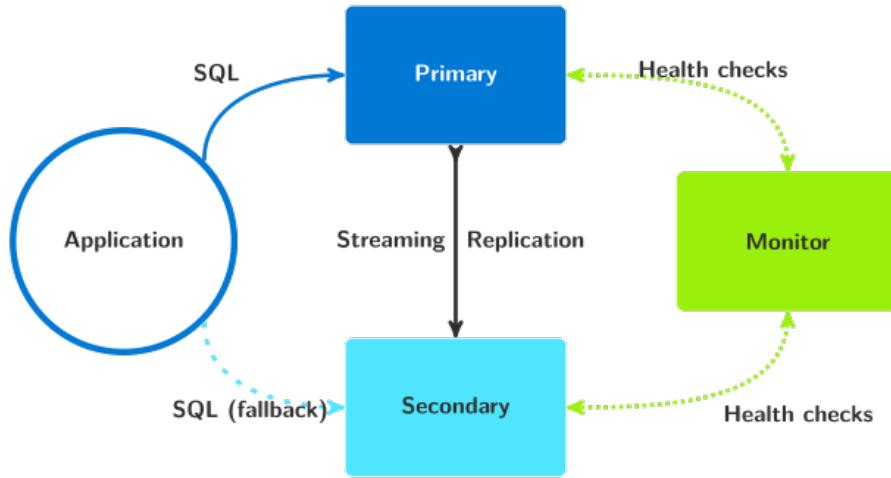


Abbildung 2.14: pg_auto_failover-Architektur - Single Standby

Aber auch Multi-Nodes können eingebunden werden[11]:

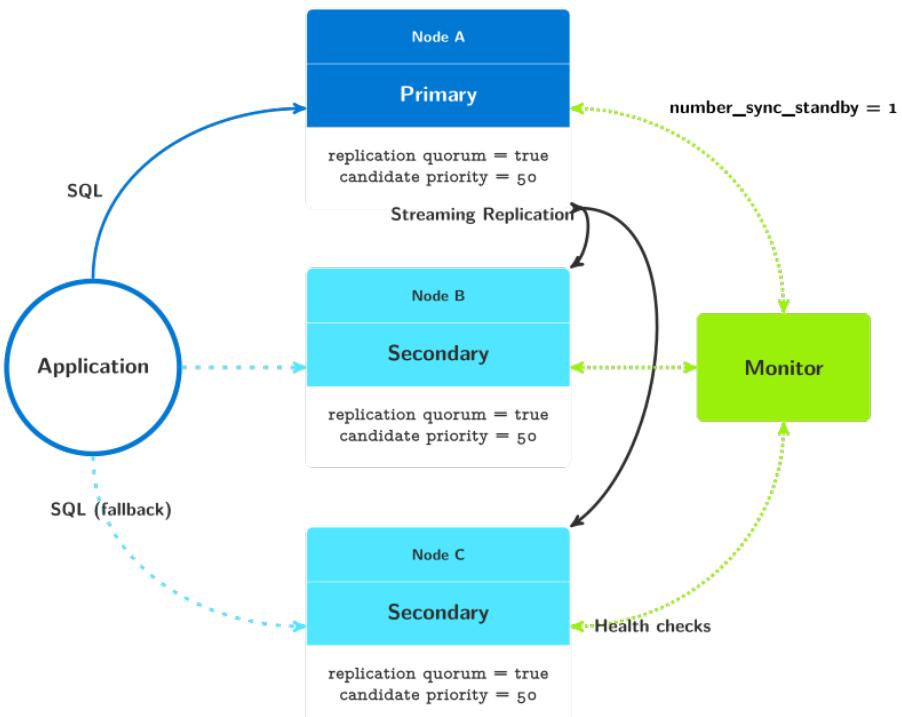


Abbildung 2.15: pg_auto_failover-Architektur - Multi-Node Standby

pg_auto_failover kann Citus einbinden[5]. Allerdings bleibt die Architektur im Kern immer Monolithisch.

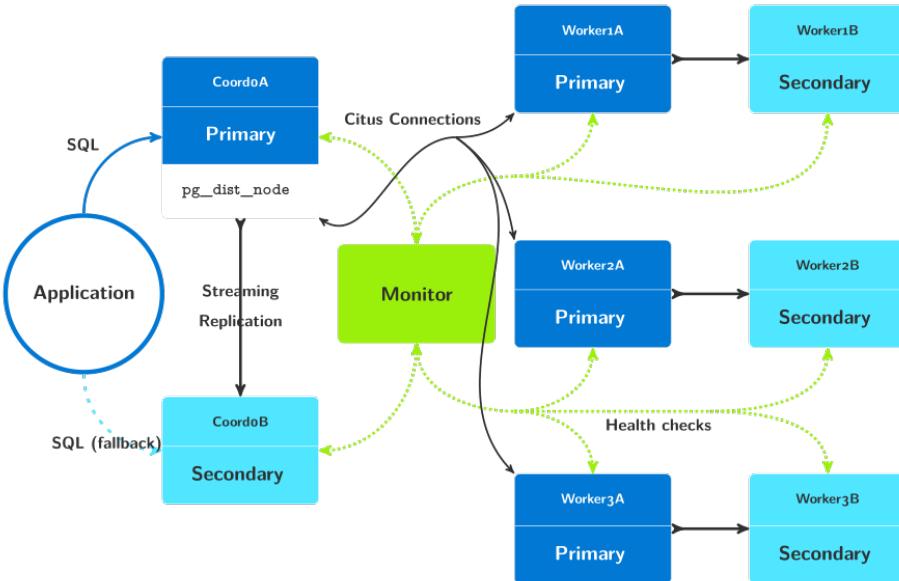


Abbildung 2.16: pg_auto_failover-Architektur - Citus

2.1.5.5 Patroni

2.1.5.5.1 Replikation

2.1.5.5.2 Proxy

Patroni benötigt einen HAProxy, um Load Balancing usw. [9]

2.1.5.5.3 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.

2.1.5.5.4 etcd

Patroni benötigt etcd als key-value-store

2.1.5.5.5 Architektur

Das Architektur-Schaubild sieht folgendermassen aus:

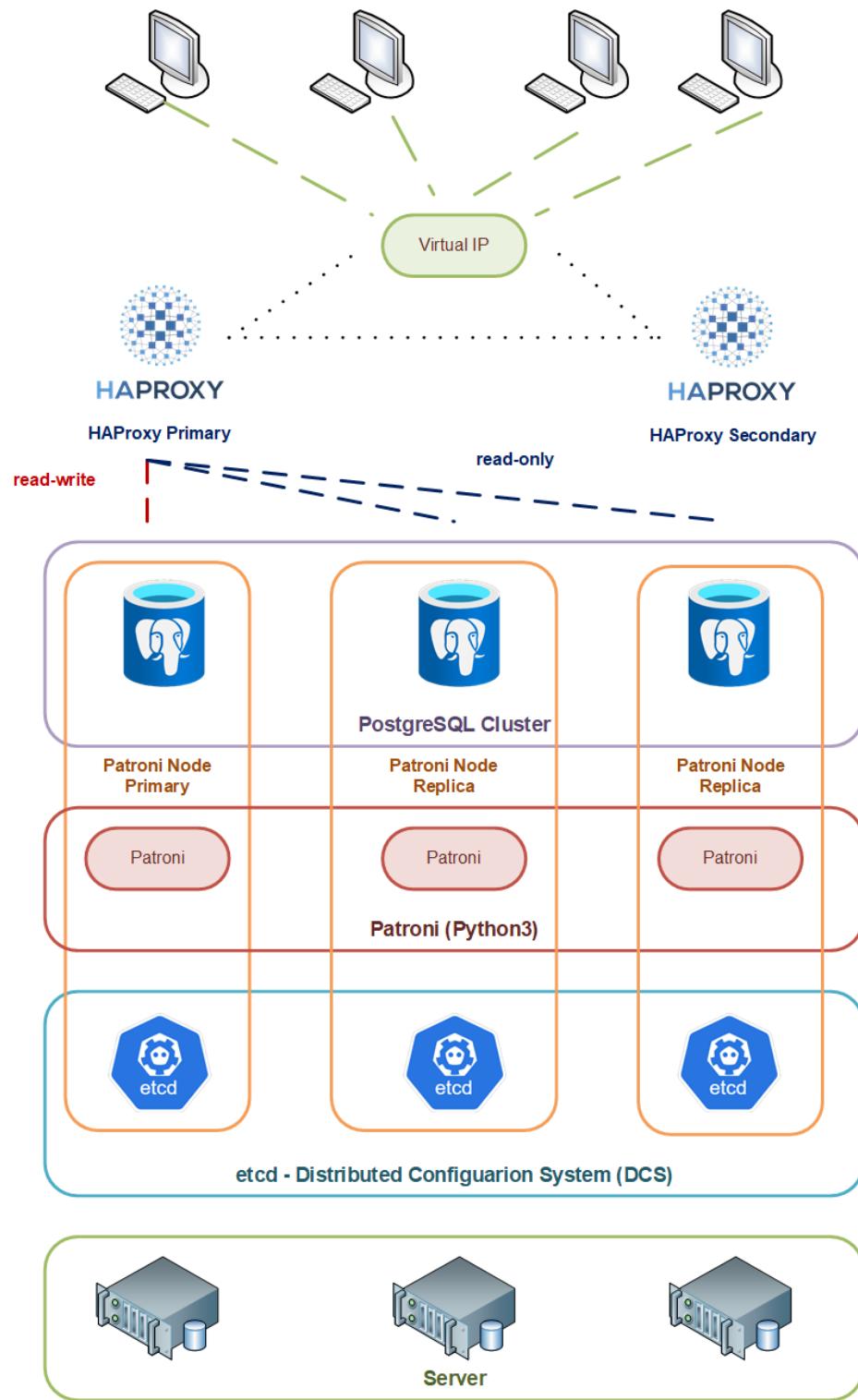


Abbildung 2.17: Patroni-Architektur

2.1.5.5.6 Maintenance

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

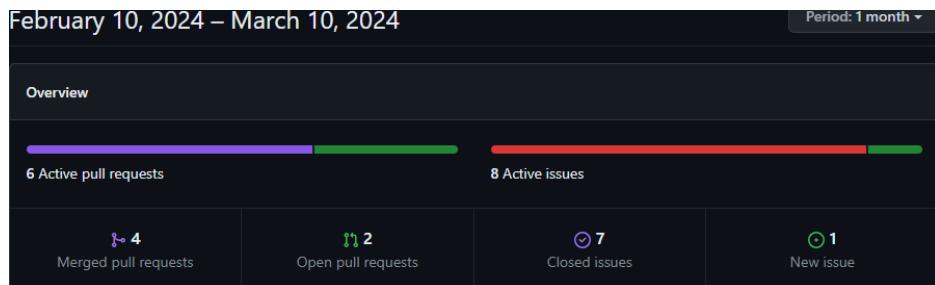


Abbildung 2.18: Patroni - Pulse

Code wird Regelmässig hinzugefügt und entfernt:

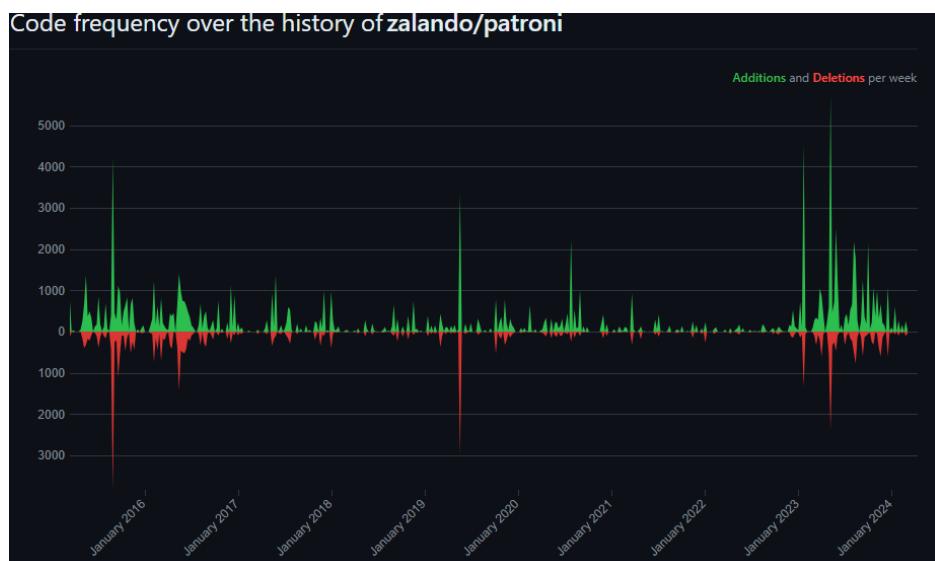


Abbildung 2.19: Patroni - Code Frequency

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 2.20: Patroni - Community Standards

Die Contributors commiten, löschen und erweitern Patroni Regelmässig:

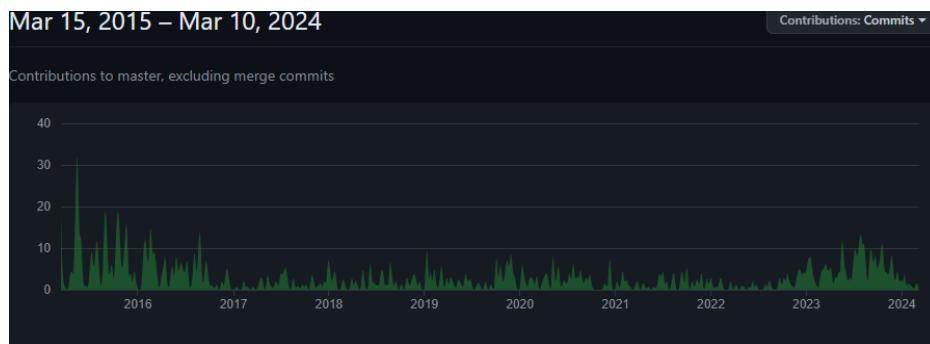


Abbildung 2.21: Patroni - Contributors Commits

Diplomarbeit



Abbildung 2.22: Patroni - Contributors Deletations



Abbildung 2.23: Patroni - Contributors Additions

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

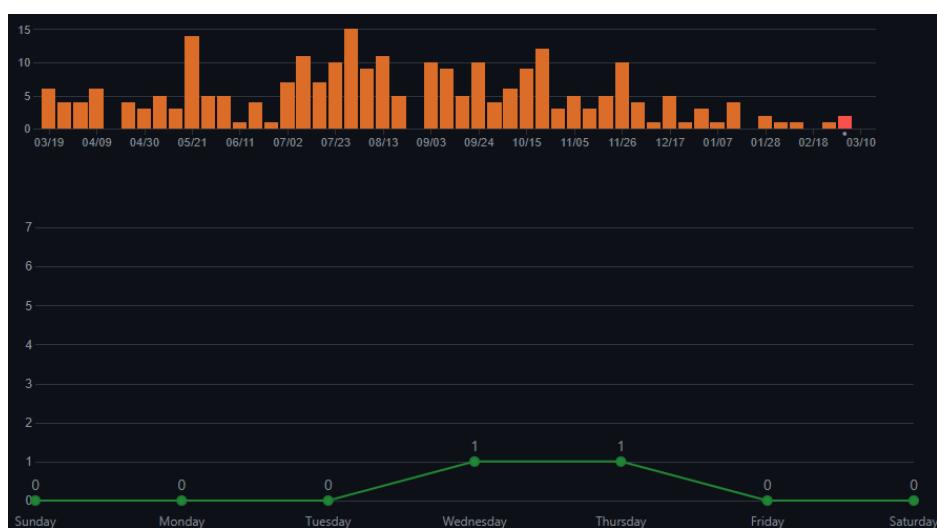


Abbildung 2.24: Patroni - Commit Activity

Nebst Zalando selbst, hat auch EnterpriseDB[21] ein grösseres Repository eingebunden. Dies weil EnterpriseDB stark auf Patroni setzt.



Abbildung 2.25: Patroni - Network Graph

2.1.5.6 CloudNativePG

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[67].

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:

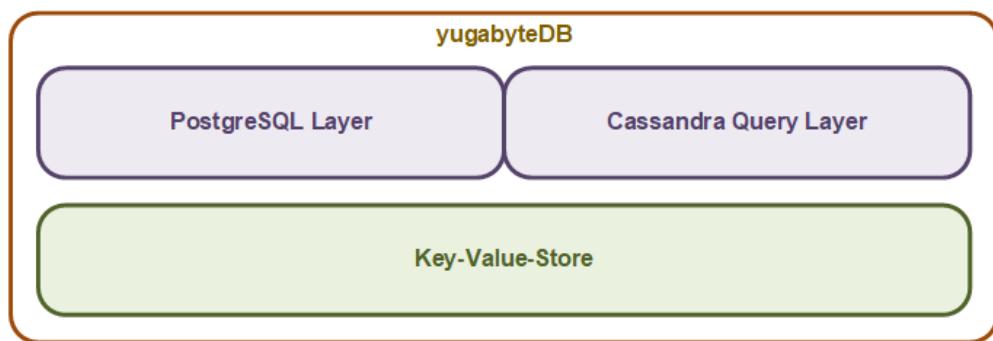


Abbildung 2.26: YugabyteDB - Grundkonzept

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

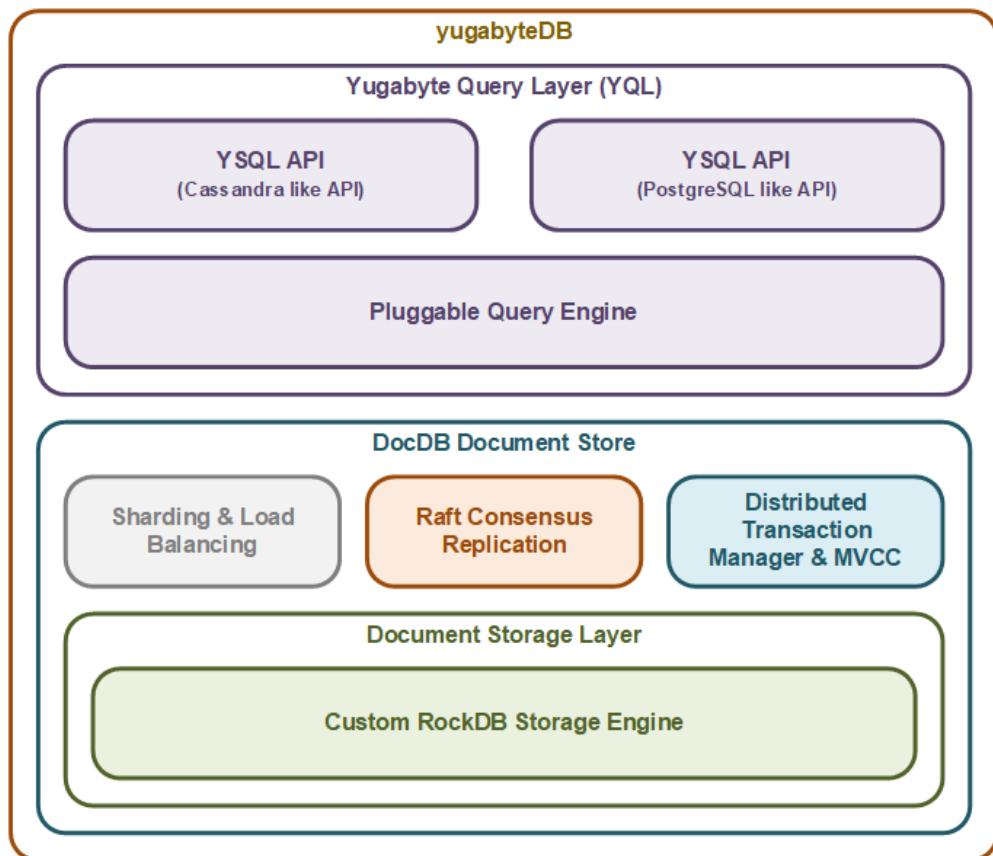


Abbildung 2.27: YugabyteDB - Architektur

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 2.28: YugabyteDB - Sharding

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

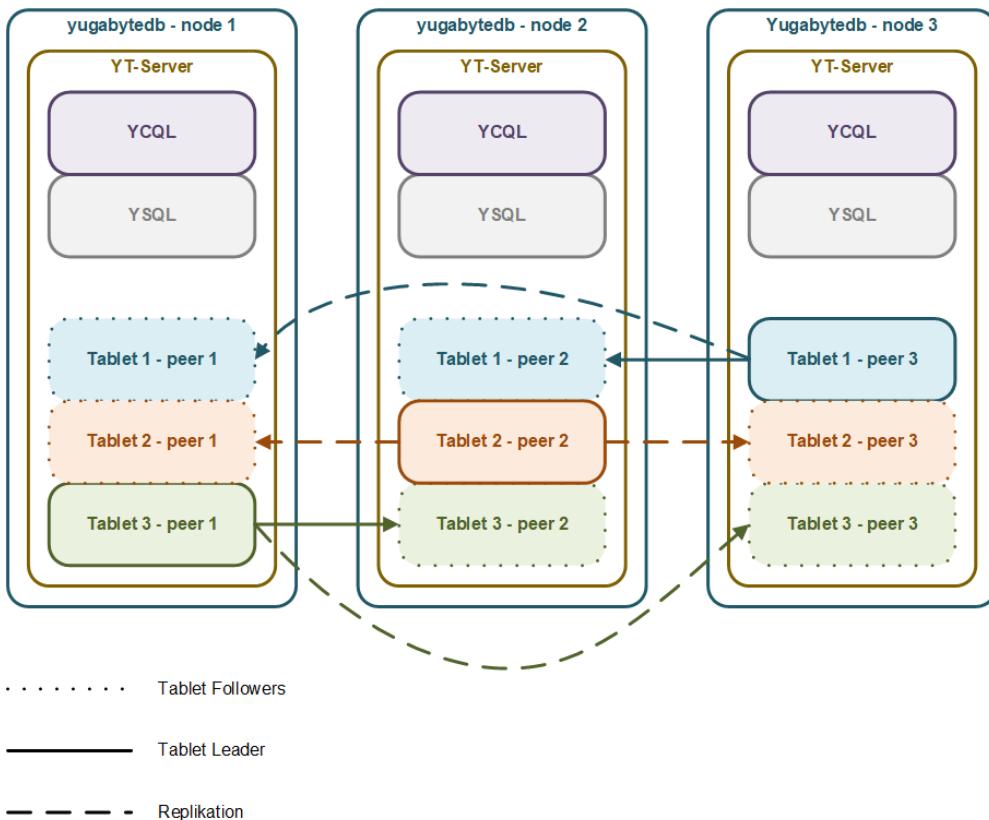


Abbildung 2.29: YugabyteDB - Tablet - Leader und Follower

Mit dem Replikationsfaktor kann angegeben werden, auf wie vielen Nodes ein Tablet repliziert werden soll.

Durch das Raft-Protokoll werden die Tablet-Leader regelmässig gewechselt.
 Fällt ein Tablet-Leader aus, so wird ein Follower als neuer Leader ausgehandelt:

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 Microsoft Konzern eingebettet

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.

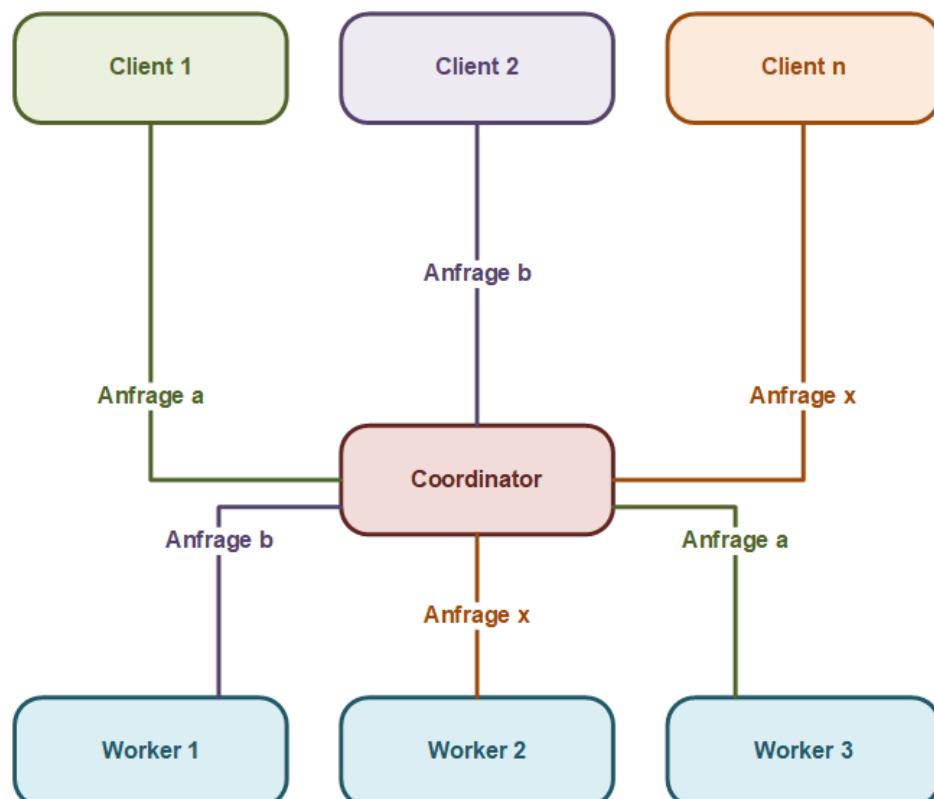


Abbildung 2.30: Citus - Coordinator und Workers

Diplomarbeit



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. [7, 4]

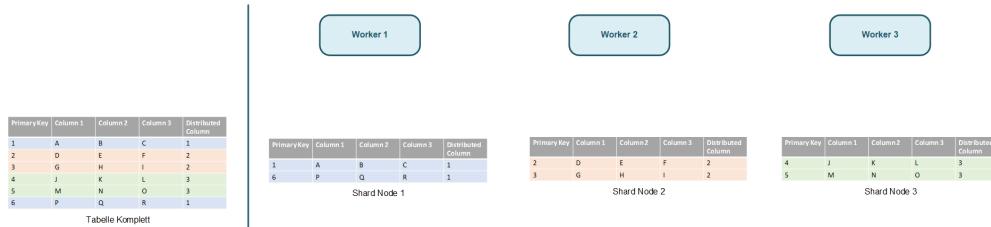


Abbildung 2.31: Citus - Row-Based-Sharding

Schema-based sharding

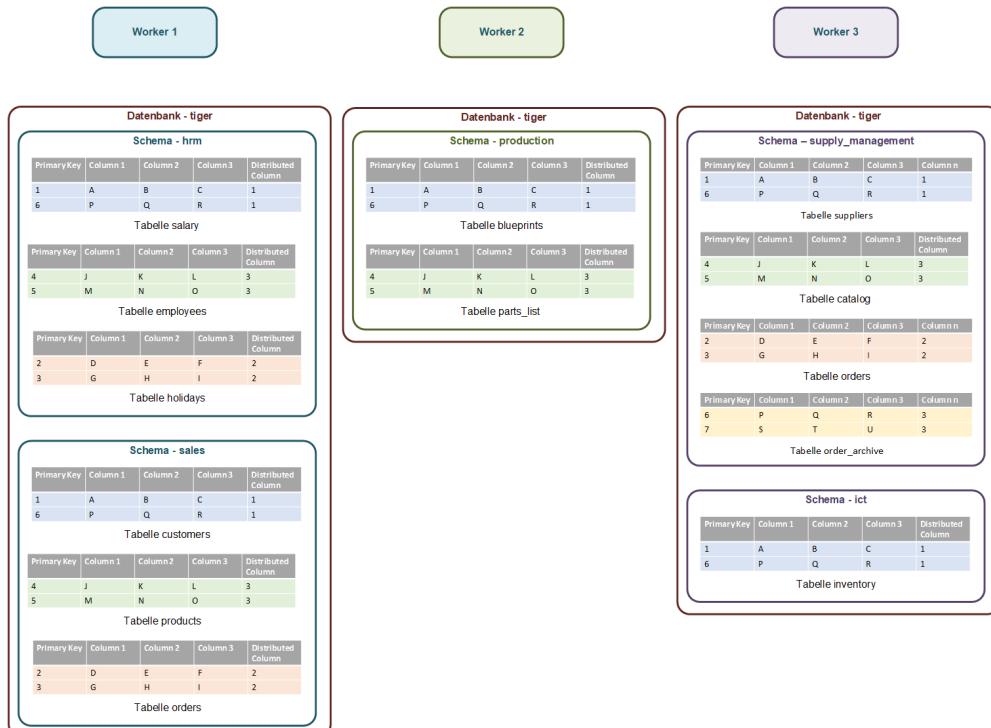


Abbildung 2.32: Citus - Schema-Based-Sharding

Schlussfolgerung Beide Sharding-Methoden haben eine grosse Schwäche. 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[6]. 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:

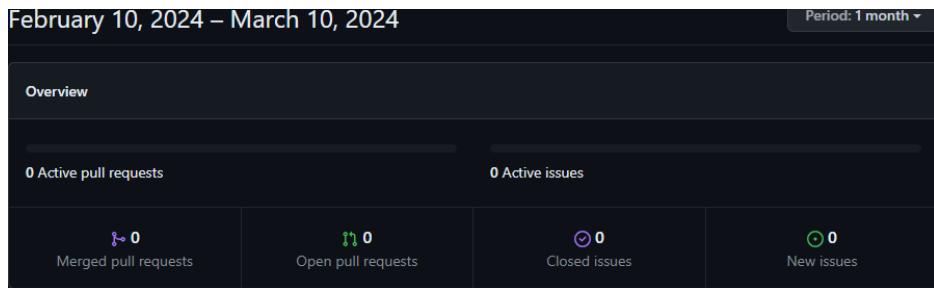


Abbildung 2.33: Stackgres - Pulse

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

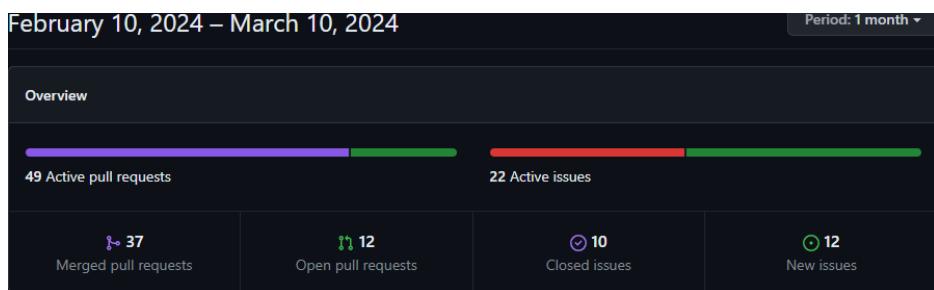


Abbildung 2.34: Citus - Pulse

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

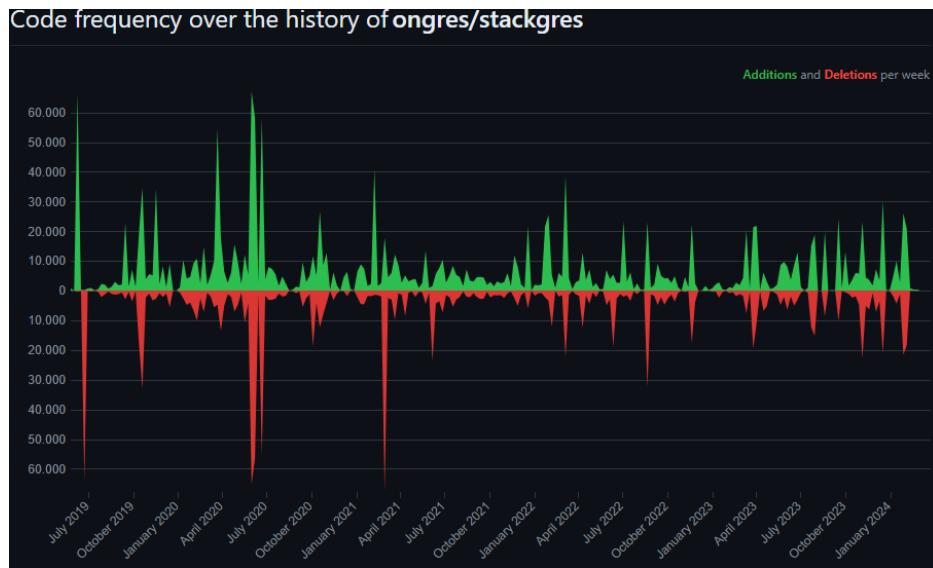


Abbildung 2.35: Stackgres - Code Frequency

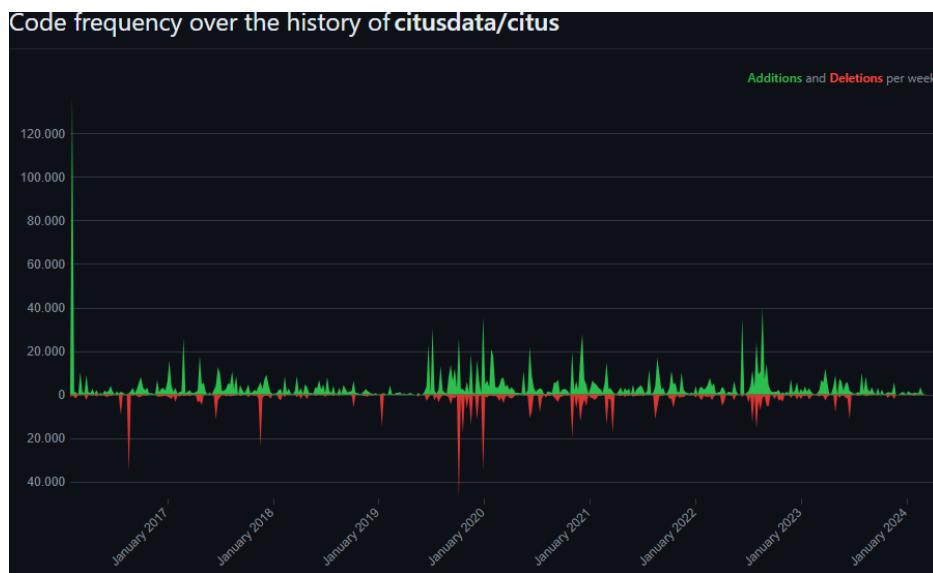


Abbildung 2.36: Citus - Code Frequency

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

Community Standards

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

Checklist

✓ Description	
✓ README	
✓ Code of conduct	
● Contributing	Writing contributing guidelines Propose
✓ License	
● Security policy	Set up a security policy Propose
● Issue templates	
● Pull request template	
● Repository admins accept content reports	

Abbildung 2.37: Stackgres - Community Standards

Community Standards

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

Checklist

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

Abbildung 2.38: 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.

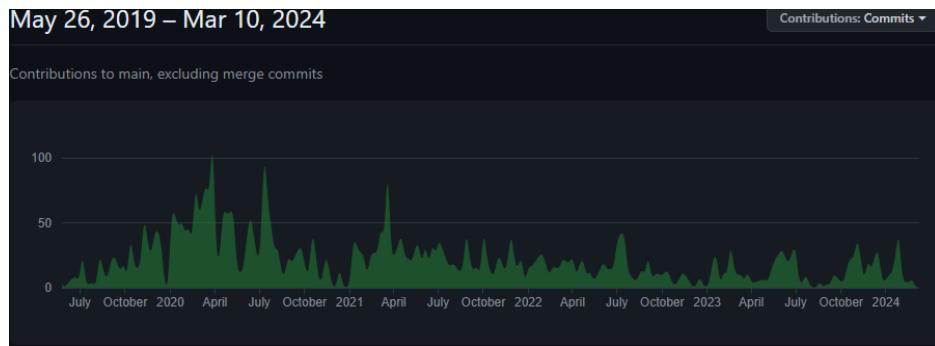


Abbildung 2.39: Stackgres - Contributors Commits



Abbildung 2.40: Stackgres - Contributors Deletations

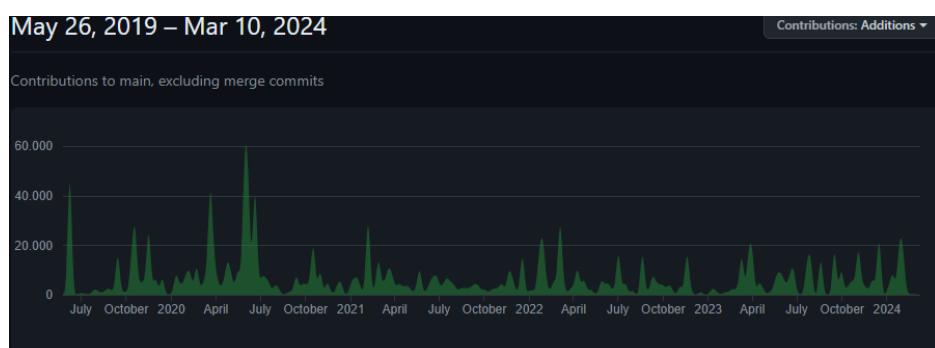


Abbildung 2.41: Stackgres - Contributors Additions

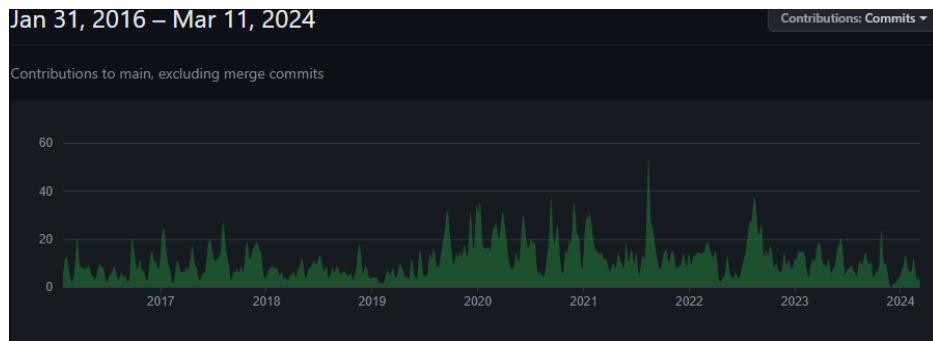


Abbildung 2.42: Citus - Contributors Commits

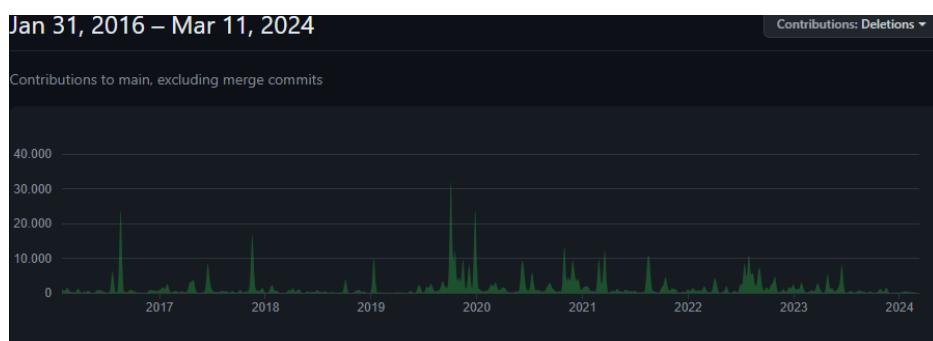


Abbildung 2.43: Citus - Contributors Deletations

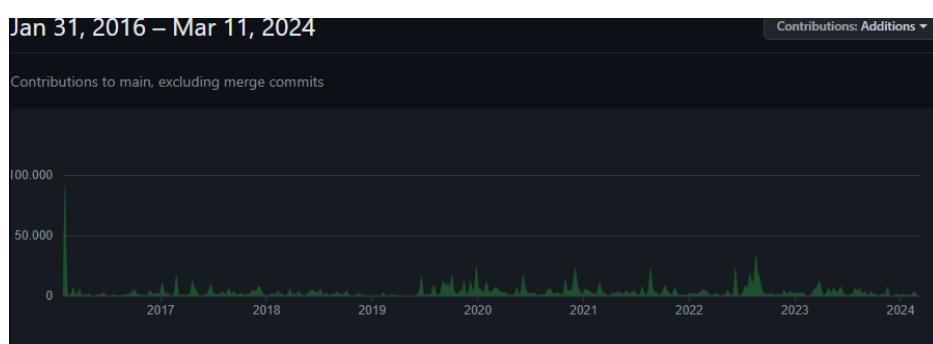


Abbildung 2.44: Citus - Contributors Additions

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:

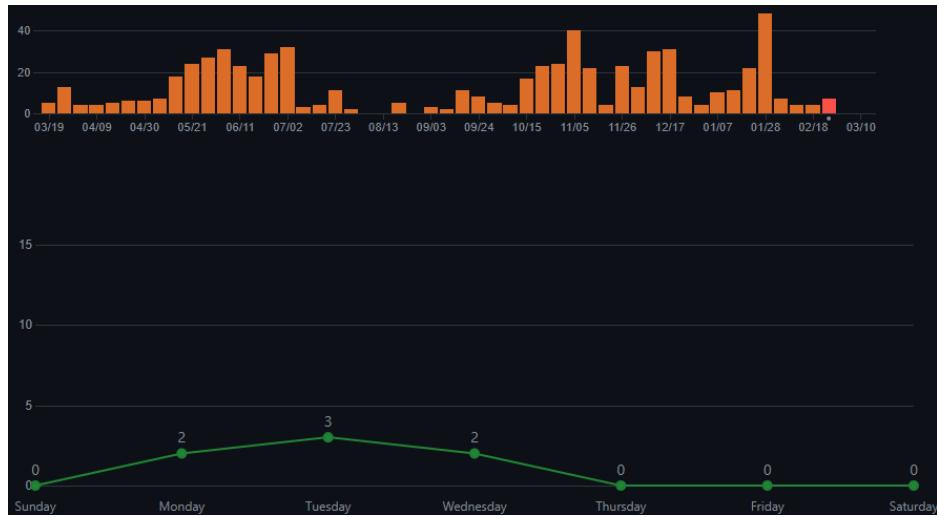


Abbildung 2.45: Stackgres - Commit Activity



Abbildung 2.46: 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.

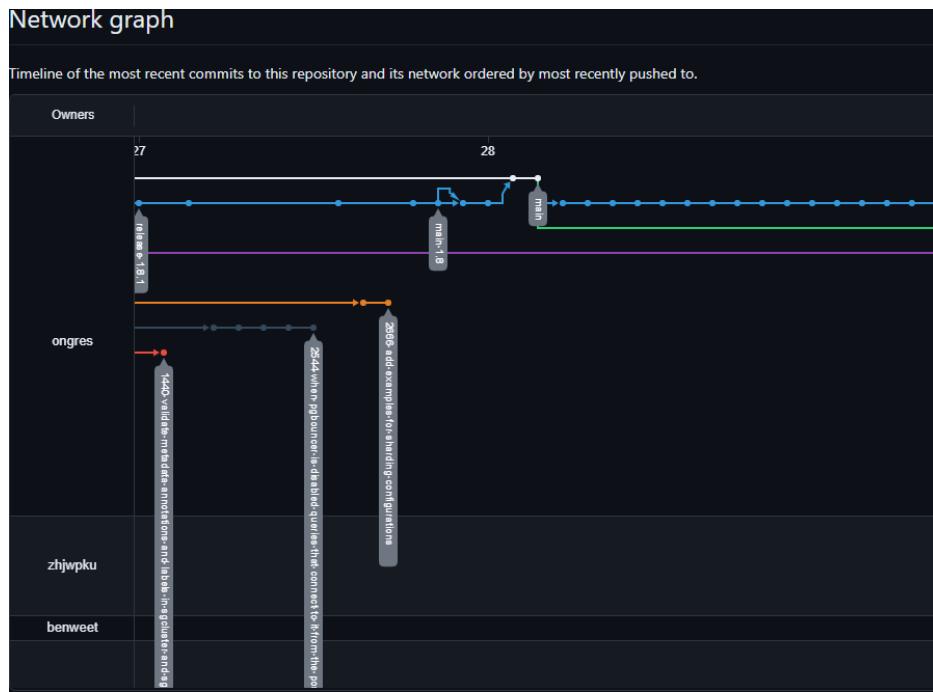


Abbildung 2.47: Stackgres - Network Graph

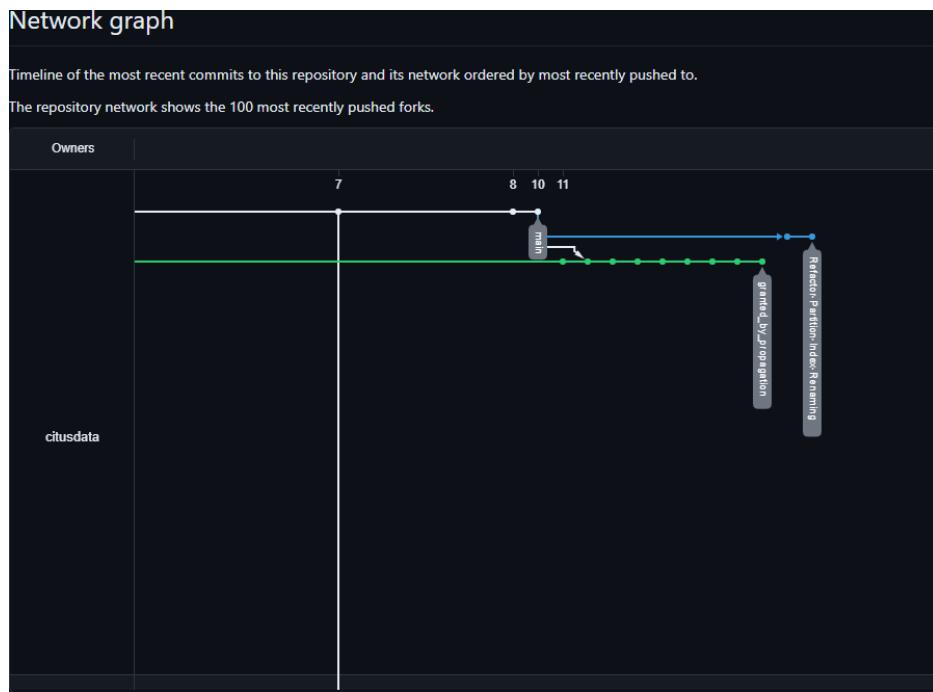


Abbildung 2.48: Citus - Network Graph

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.
3	pg_auto_failover	Vorausgeschieden	pg_auto_failover würde zwar Citus-Support bieten, allerdings gibt es keine gut dokumentierte Implementation für Kubernetes. Erfüllt daher das Kriterium für die Synergien nicht
			CloudNativePG ist keine vollständige Cloud Native Lösung. Mittels Citus könnte sogar eine Distributed SQL Lösung implementiert werden.
4	CloudNativePG	Vorausgeschieden	Die Grundarchitektur bleibt aber Monolithisch mit einem Primary und Repikas. Und da kein Benefit in Form von Synergien vorhanden sind, fällt CloudNativePG raus. Citus row-based-sharding wäre Hocoeffizient wenn es um Ressourcenverteilung geht und zudem echtes Sharding. Allerdings setzt es anpassungen an den Tabellen der Applikationen voraus.
8	Citus row-based-sharding	Vorausgeschieden	Das KSGR ist allerdings kein Softwarehaus und kann keine Forks durchführen, auch weil viele Applikationen zertifiziert sein müssen. Scheitert daher an der Machbarkeit

Tabelle 2.4: Vorauswahl - Ausgeschieden

Entsprechend werden nur noch nachfolgende Lösungen genauer betrachtet:

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

Tabelle 2.5: 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

Continued on next page

Tabelle 2.6: Evaluationssyssteme

Server	Typ	Funktion	Full Qualified Device Name	IP
vks0032	Distributed SQL	Virteulle IP	vks0032.ksgr.ch	10.0.20.106
vks0040	Monolith	Virteulle IP	vks0040.ksgr.ch	10.0.20.113

Tabelle 2.6: Evaluationssyssteme

2.1.7.1 rke2 - Evaluationsplattform

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

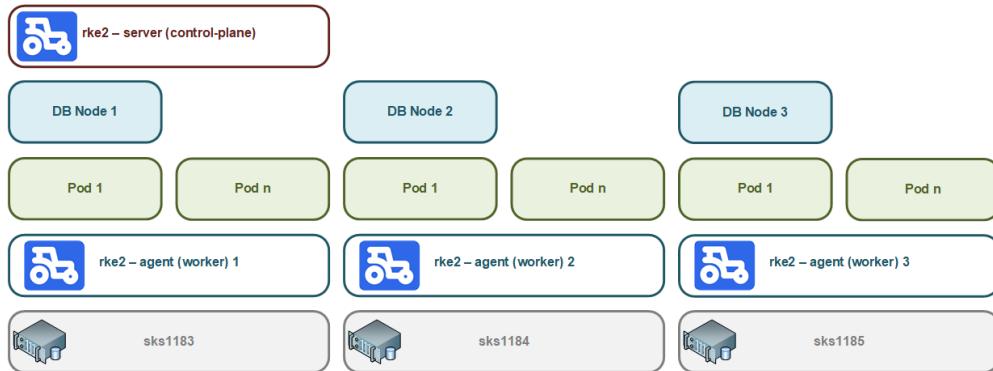


Abbildung 2.49: Evaluationssystem - Distributed SQL / Shards

Die Konfiguration der rke2-Nodes sieht folgendermassen aus:

Kubernetes Runtime	rke2
Container-Enviroment	containerd
Container Network Interface (CNI)	cilium
loud Native Storage (CNS)	local-path-provisioner

Tabelle 2.7: Evaluationssysstem - Distributed SQL / Sharding

2.1.7.2 Patroni

2.1.7.2.1 Architektur

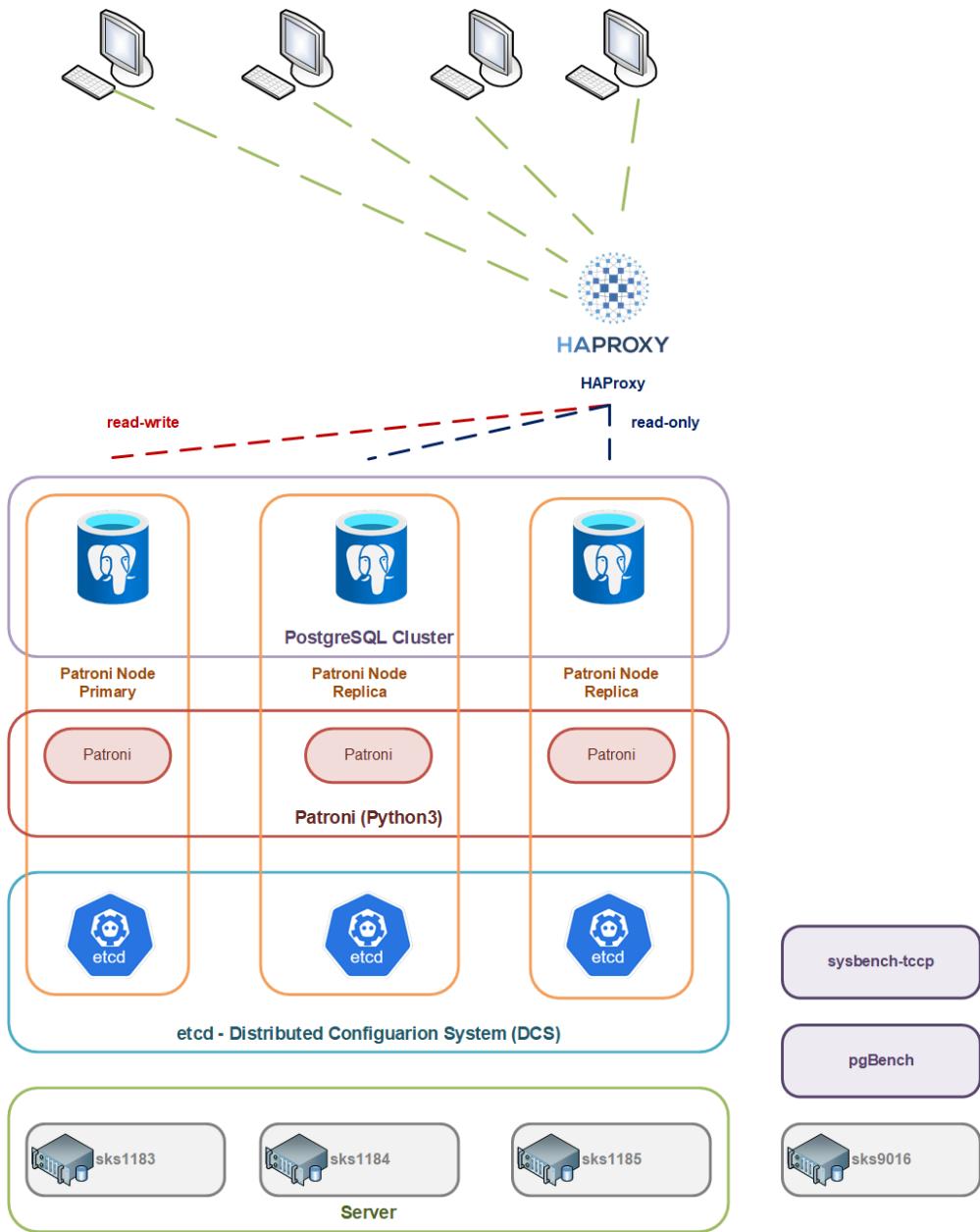


Abbildung 2.50: Patroni - Evaluationsarchitektur

2.1.7.3 StackGres - Citus

2.1.7.3.1 Architektur

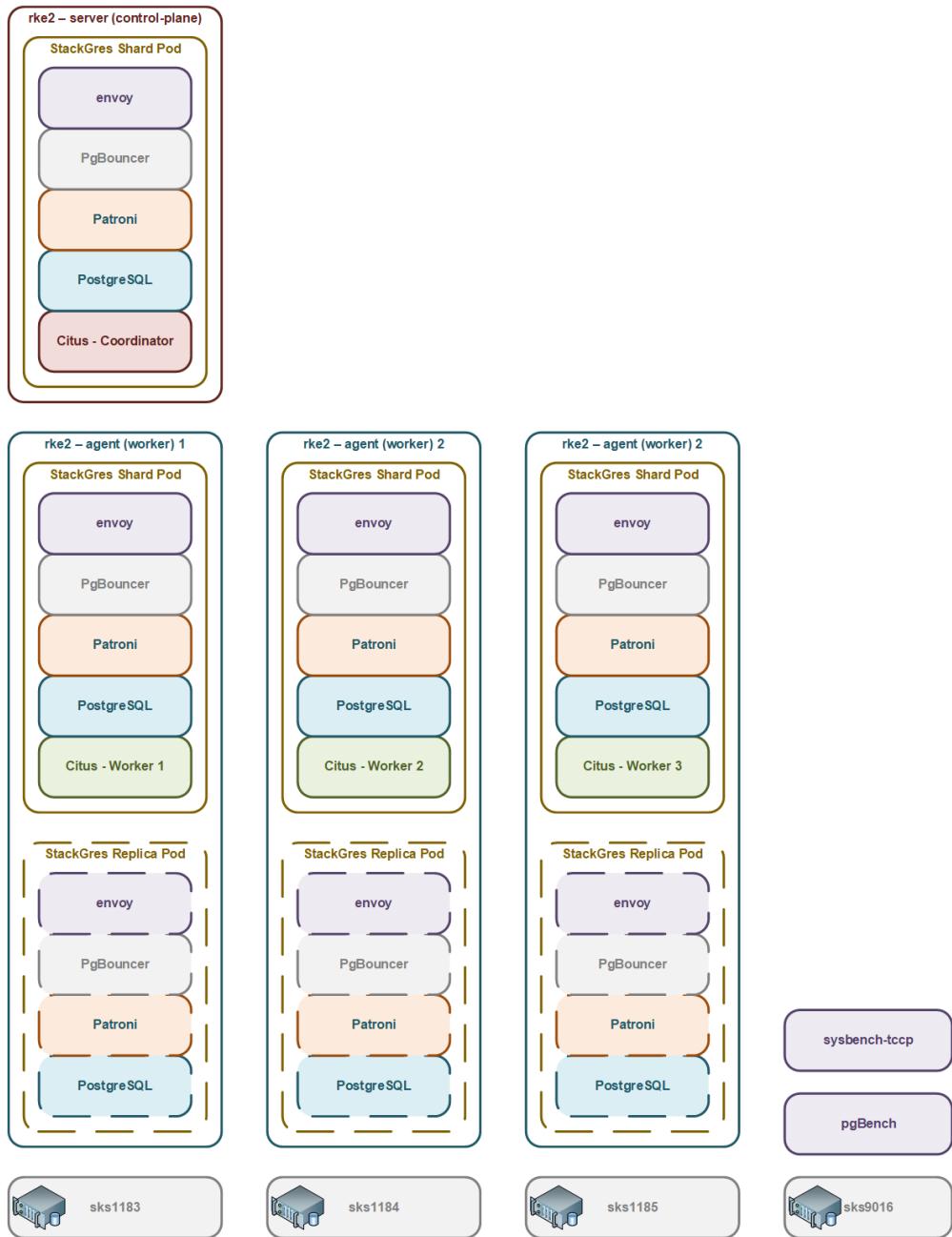


Abbildung 2.51: Stackgres - Citus - Evaluationsarchitektur

2.1.7.4 YugabyteDB

2.1.8 Gegenüberstellung der Lösungen

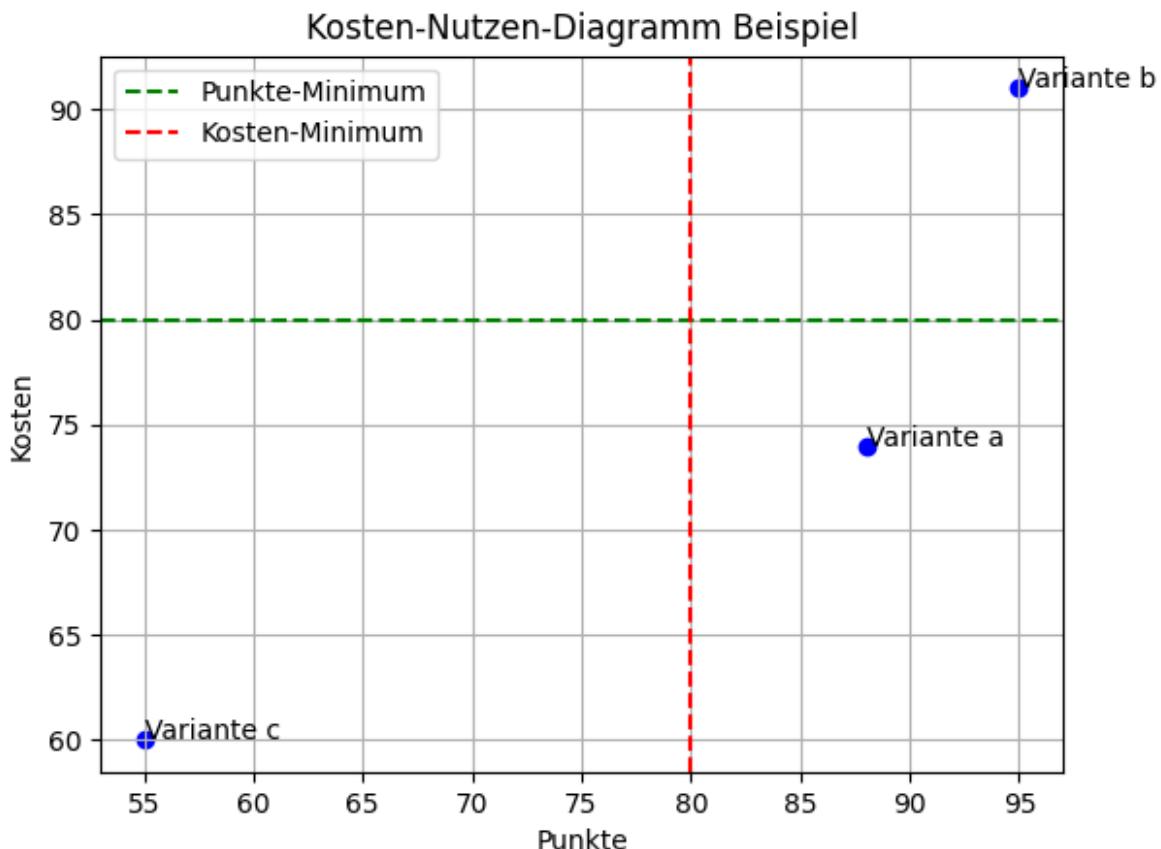


Abbildung 2.52: Kosten-Nutzen-Analyse

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[32]	1
1.2	Wahlkreise Kanton St. Gallen[57]	2
1.3	Spitalregionen / Spitalstrategie Kanton St. Gallen[26]	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	21
1.10	Projektrisiken mit Massnahmen	23
2.1	Monolithische vs. verteilte SQL Systeme	32
2.2	CAP-Theorem	35
2.3	Datenbankskalierung	36
2.4	Präferenzmatrix	40
2.5	Präferenzmatrix - Failover	41
2.6	Präferenzmatrix - Switchover	42
2.7	Präferenzmatrix - Restore	43
2.8	Präferenzmatrix - Replikation	44
2.9	Präferenzmatrix - Sharding	45
2.10	Präferenzmatrix - Quorum	46
2.11	Präferenzmatrix - Management-API	47
2.12	Präferenzmatrix - Backup	48
2.13	Präferenzmatrix - Performance	49
2.14	pg_auto_failover-Architektur - Single Standby	53
2.15	pg_auto_failover-Architektur - Multi-Node Standby	53
2.16	pg_auto_failover-Architektur - Citus	54
2.17	Patroni-Architektur	55
2.18	Patroni - Pulse	56
2.19	Patroni - Code Frequency	56
2.20	Patroni - Community Standards	57
2.21	Patroni - Contributors Commits	57
2.22	Patroni - Contributors Deletations	58
2.23	Patroni - Contributors Additions	58
2.24	Patroni - Commit Activity	58

2.25 Patroni - Network Graph	59
2.26 YugabyteDB - Grundkonzept	59
2.27 YugabyteDB - Architektur	60
2.28 YugabyteDB - Sharding	61
2.29 YugabyteDB - Tablet - Leader und Follower	61
2.30 Citus - Coordinator und Workers	62
2.31 Citus - Row-Based-Sharding	63
2.32 Citus - Schema-Based-Sharding	63
2.33 Stackgres - Pulse	64
2.34 Citus - Pulse	64
2.35 Stackgres - Code Frequency	65
2.36 Citus - Code Frequency	65
2.37 Stackgres - Community Standards	66
2.38 Citus - Community Standards	66
2.39 Stackgres - Contributors Commits	67
2.40 Stackgres - Contributors Deletations	67
2.41 Stackgres - Contributors Additions	67
2.42 Citus - Contributors Commits	68
2.43 Citus - Contributors Deletions	68
2.44 Citus - Contributors Additions	68
2.45 Stackgres - Commit Activity	69
2.46 Citus - Commit Activity	69
2.47 Stackgres - Network Graph	70
2.48 Citus - Network Graph	70
2.49 Evaluationssystem - Distributed SQL / Shards	72
2.50 Patroni - Evaluationsarchitektur	73
2.51 Stackgres - Citus - Evaluationsarchitektur	74
2.52 Kosten-Nutzen-Analyse	75

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	20
1.11	Projektcontrolling	25
1.12	Initialer Statusbericht	28
1.13	Zweiter Statusbericht	29
1.14	Fachgespräche	30
2.1	Quorum Beispiele	33
2.2	Anforderungskatalog	38
2.3	Stakeholder	39
2.4	Vorauswahl - Ausgeschieden	71
2.5	Vorauswahl - Evaluation	71
2.6	Evaluationssystemsste	72
2.7	Evaluationssystem - Distributed SQL / Sharding	72
I	Arbeitsrapport	i
II	Fachgespräche - Protokoll	ii
III	Kommentare - Anmerkung	iii

Listings

1	Python LaTex - zotero.py - Zotero BibLaTex Importer	iv
2	Python LaTex - riskmatrix.py - Risikomatrizen	x
3	Python LaTex - cost_benefit_diagram.py - Kosten-Nutzen-Diagramm	xv
4	Python LaTex - csscatter_plotter_conf.yaml - Konfigurationsdatei - Kosten-Nutzen-Diagramm	xvii
5	Python LaTex - pandas_dataframe_to_latex_table.py CSV - LaTex Tabelle	xviii
6	Python LaTex - csv_to_latex_diplomarbeit.yaml - Konfigurationsdatei - CSV - LaTex-Tabelle	xxvii

Literatur

- [1] *About pgbench-tools*. <https://github.com/gregs1104/pgbench-tools>. original-date: 2010-02-17T13:33:28Z. 2023.
- [2] Satyadeep Ashwathnarayana und Inc. Netdata. *How to monitor and fix Database bloats in PostgreSQL? | Netdata Blog*. <https://blog.netdata.cloud/postgresql-database-bloat/>. 2022.
- [3] unknown author. *Architecture Basics — pg_auto_failover 2.0 documentation*. <https://pg-auto-failover.readthedocs.io/en/main/architecture.html>.
- [4] unknown author. *Choosing Distribution Column - Citus 12.1 documentation*. https://docs.citusdata.com/en/v12.1/sharding/data_modeling.html#distributed-data-modeling.
- [5] unknown author. *Citus Support — pg_auto_failover 2.0 documentation*. <https://pg-auto-failover.readthedocs.io/en/main/citus.html>.
- [6] 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.
- [7] unknown author. *Concepts - Citus 12.1 documentation - row-based-sharding*. https://docs.citusdata.com/en/v12.1/get_started/concepts.html#row-based-sharding.
- [8] unknown author. *etcd*. <https://etcd.io/>.
- [9] unknown author. *HAProxy Documentation Converter*. <https://docs.haproxy.org/>.
- [10] unknown author. *HAProxy version 2.9.6 - Starter Guide*. <https://docs.haproxy.org/2.9/intro.html#3.2>.
- [11] unknown author. *Multi-node Architectures — pg_auto_failover 2.0 documentation*. <https://pg-auto-failover.readthedocs.io/en/main/architecture-multi-standby.html>.
- [12] unknown author. *Tuning PostgreSQL with pgbench*. <https://www.cloudbees.com/blog/tuning-postgresql-with-pgbench>. 2017.
- [13] GitLab B.V. und GitLab Inc. *The DevSecOps Platform | GitLab*. <https://about.gitlab.com/>.
- [14] Fernando Laudares Camargos Avinash Vallarapu. *Tuning PostgreSQL for sysbench-tpcc*. <https://www.percona.com/blog/tuning-postgresql-for-sysbench-tpcc/>. 2018.
- [15] Alexandre Cassen und Read the Docs. *Introduction — Keepalived 1.2.15 documentation*. <https://keepalived.readthedocs.io/en/latest/introduction.html>. 2017.
- [16] Microsoft Corporation. *Azure SQL-Datenbank – ein verwalteter Clouddatenbankdienst | Microsoft Azure*. <https://azure.microsoft.com/de-de/products/azure-sql/database>. 2023.

- [17] Microsoft Corporation. *Datenbank-Software und Datenbankanwendungen / Microsoft Access*. <https://www.microsoft.com/de-de/microsoft-365/access>. 2023.
- [18] Microsoft Corporation. *Microsoft Data Platform / Microsoft*. <https://www.microsoft.com/de-ch/sql-server>.
- [19] ORACLE CORPORATION. „Oracle (Active) Data Guard 19c“. In: (2019), S. 14.
- [20] Varun Dhawan und data-nerd.blog. *PostgreSQL-Diagnostic-Queries – data-nerd.blog*. <https://data-nerd.blog/2018/12/30/postgresql-diagnostic-queries/>.
- [21] EDB: Open-Source, Enterprise Postgres Database Management. <https://www.enterprisedb.com/>.
- [22] Elektronik-Kompendium.de und Schnabel Schnabel. SAN - Storage Area Network. <https://www.elektronik-kompendium.de/sites/net/0906071.htm>. 2023.
- [23] DB-Engines und solidIT consulting & software development gmbh. *DB-Engines Ranking*. <https://db-engines.com/en/ranking>.
- [24] DB-Engines und solidIT consulting & software development gmbh. *relationale Datenbanken - DB-Engines Enzyklopädie*. <https://db-engines.com/de/article/relationale+Datenbanken?ref=RDBMS>.
- [25] The Linux Foundation. *Harbor*. <https://goharbor.io/>. 2023.
- [26] 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>.
- [27] Git. *About - Git*. <https://git-scm.com/about>.
- [28] IBM Deutschland GmbH. *Was ist das CAP-Theorem? / IBM*. <https://www.ibm.com/de-de/topics/cap-theorem>. 2023.
- [29] IBM Deutschland GmbH. *Was ist OLAP? / IBM*. <https://www.ibm.com/de-de/topics/olap>.
- [30] Jedox GmbH. *Was ist OLAP? Online Analytical Processing im Überblick*. <https://www.jedox.com/de/blog/was-ist-olap/>. Section: Knowledge.
- [31] 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>.
- [32] 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/InstitutionenGesundheitswesens/Spitaeler/Dok%20Spitler/Kenndaten%202016%20Spit%C3%A4ler.pdf>.
- [33] The Pgpool Global Development Group. *What is Pgpool-II?* <https://www.pgpool.net/docs/44/en/html/intro-whatis.html>. 2023.

- [34] The PostgreSQL Global Development Group. *25.1. Routine Vacuuming*. <https://www.postgresql.org/docs/16/routine-vacuuming.html>. 2023.
- [35] The PostgreSQL Global Development Group. *27.1. Comparison of Different Solutions*. <https://www.postgresql.org/docs/16/different-replication-solutions.html>. 2023.
- [36] The PostgreSQL Global Development Group. *pgbench*. <https://www.postgresql.org/docs/16/pgbench.html>. 2023.
- [37] Inc. HashiCorp. *Terraform by HashiCorp*. <https://www.terraform.io/>.
- [38] Patrick Hunt, Mahadev Konar, Flavio P Junqueira und Benjamin Reed. „ZooKeeper: Wait-free coordination for Internet-scale systems“. In: (2010).
- [39] Splunk Inc. *Splunk / Der Schlüssel zu einem resilienten Unternehmen*. https://www.splunk.com/de_de. 2023.
- [40] Sebastian Insauti. *Scaling PostgreSQL for Large Amounts of Data*. <https://severalnines.com/blog/scaling-postgresql-large-amounts-data/>. 2019.
- [41] Shiv Iyer und MinervaDB. *PostgreSQL DBA Daily Checklist*. <https://minervadb.xyz/postgresql-dba-daily-checklist/>. 2020.
- [42] jobinau/pg_gather. https://github.com/jobinau/pg_gather. original-date: 2021-01-19T08:12:07Z. 2024.
- [43] Unmesh Joshi. *Quorum*. <https://martinfowler.com/articles/patterns-of-distributed-systems/quorum.html>. 2020.
- [44] Martin Keen und IBM Deutschland GmbH. *IBM Db2*. <https://www.ibm.com/de-de/products/db2>.
- [45] Pasha Kostohrys. *Database replication — an overview*. <https://medium.com/@pkostohrys/database-replication-an-overview-f7ade110477>. 2020.
- [46] Anatoli Kreyman. *Was ist eigentlich Splunk?* <https://www.kreyman.de/index.php/splunk/76-was-ist-eigentlich-splunk-big-data-platform-monitoring-security>.
- [47] Pankaj Kushwaha und Unit 3D North Point House. *POSTGRES DATABASE MAINTENANCE. Routine backup of daily database... | by Pankaj kushwaha | Medium*. <https://pankajconnect.medium.com/postgresql-database-maintenance-66cd638d25ab>.
- [48] Red Hat Limited. *Was ist Ansible?* <https://www.redhat.com/de/technologies/management/ansible/what-is-ansible>.
- [49] 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>.
- [50] Switzerland Linuxfabrik GmbH Zurich. *Keepalived — Open Source Admin-Handbuch der Linuxfabrik*. <https://docs.linuxfabrik.ch/software/keepalived.html>. 2023.

- [51] Nico Litzel, Stefan Luber und Vogel IT-Medien GmbH. *Was ist Elasticsearch?* <https://www.bigdata-insider.de/was-ist-elasticsearch-a-939625/>. 2020.
- [52] SRA OSS LLC. *pgpool Wiki*. https://www.pgpool.net/mediawiki/index.php/Main_Page. 2023.
- [53] Hewlett Packard Enterprise Development LP. *Was ist SAN-Speicher? / Glossar*. <https://www.hpe.com/ch/de/what-is/san-storage.html>.
- [54] Julian Markwort. „Benchmarking four Different Replication Solutions“. In: () .
- [55] Diego Ongaro. „Consensus: Bridging Theory and Practice“. In: (2014) .
- [56] Bruno Queirós und LinkedIn Ireland Unlimited Company. *Postgresql replication with automatic failover*. <https://www.linkedin.com/pulse/postgresql-replication-automatic-failover-bruno-queiros-c3%b3s>. 2020.
- [57] 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>.
- [58] Ed Reckers und SnapLogic Inc. *Was ist die Snowflake-Datenplattform?* <https://www.snaplogic.com/de/blog/snowflake-data-platform>. 2023.
- [59] IONOS SE. *Apache Cassandra: Verteilte Verwaltung großer Datenbanken*. <https://www.ionos.de/digitalguide/hosting/hosting-technik/apache-cassandra-vorgestellt/>. 2021.
- [60] IONOS SE. *Datenbankmanagementsystem (DBMS) erklärt*. <https://www.ionos.de/digitalguide/hosting/hosting-technik/datenbankmanagementsystem-dbms-erklaert/>. 2020.
- [61] IONOS SE. *MongoDB – die flexible und skalierbare NoSQL-Datenbank*. <https://www.ionos.de/digitalguide/websites/web-entwicklung/mongodb-vorstellung-und-vergleich-mit-mysql>. 2019.
- [62] IONOS SE. *SQLite: Die bekannte Programmzbibliothek im Detail vorgestellt*. <https://www.ionos.de/digitalguide/websites/web-entwicklung/sqlite/>. 2023.
- [63] IONOS SE. *Terraform*. <https://www.ionos.de/digitalguide/server/tools/was-ist-terraform/>. 2020.
- [64] IONOS SE. *Was ist Redis? Die Datenbank vorgestellt*. <https://www.ionos.de/digitalguide/hosting/hosting-technik/was-ist-redis/>. 2020.
- [65] IONOS SE. *Was ist SIEM (Security Information and Event Management)?* <https://www.ionos.de/digitalguide/server/sicherheit/was-ist-siem/>. 2020.
- [66] Naveed Shaikh. *It's All About Replication Lag in PostgreSQL*. <https://www.percona.com/blogreplication-lag-in-postgresql/>. 2023.
- [67] Sami Ahmed Siddiqui. *Distributed SQL 101*. <https://www.yugabyte.com/distributed-sql/>.

- [68] Inc. Snowflake. *Datenbanken, Tabellen und Ansichten – Überblick | Snowflake Documentation*. <https://docs.snowflake.com/de/guides-overview-db>.
- [69] Thomas-Krenn.AG. *Git Grundlagen – Thomas-Krenn-Wiki*. https://www.thomas-krenn.com/de/wiki/Git_Grundlagen.
- [70] Mahmut Can Uçanefe. *Pgbench Load Test*. <https://medium.com/@c.ucanefe/pgbench-load-test-166b2023>.
- [71] 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[48].. 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[34].. 14, 15, 50

Cassandra Cassandra ist eine Spaltenorganisierte NoSQL-Datenbank die 2008 veröffentlicht[59] wurde.. 7, 59

CI/CD Continuous Integration/Continuous Delivery bedeutet, dass Anpassungen kontinuierlich in die Entwicklungsumgebungen integriert und auf die Zielplattformen verteilt werden[49].. 4

DBMS Ein Database Management System regelt und organisiert die Datenbasis einer Datenbank[60].. 4

Debian Debian gehört neben 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[51].. 7

etcd etcd ist [8]. 54

Failover In einem Fehlerfall wird in einem HA-System meist ein Primary Node auf den Secondary ungeplant geswitched.. 15, 32, 33, 51, 88

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, 20

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[27, 69].. 88

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[13].. 15, 51

HAProxy HAProxy [10]. 52, 54

Harbor Harbor ist ein Open-Source-Tool zur Registrierung von Richtlinien rollenbasierten Zugriffssteuerung[25]. Harbor wird beim KSGR zur Verwaltung der Kubernetes-Plattform verwendet.. 15, 51

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, 20

IBM DB2 IBM DB2 ist eine Relationale Datenbank[44] deren Vorläufer System-R von IBM zwischen 1975 und 1979 entwickelt wurde. DB2 selber wurde 1983 von IBM veröffentlicht.. 7, 35

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[50, 15].. 51

Key-Value-orientierte Siehe Key-Value-Datenbank. 91

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. 88, 89

Key-Value-Store Siehe Key-Value-Datenbank. 59

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, 88

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, 89

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[16].. 7

Microsoft Access Access wurde 1992 veröffentlicht und ist Entwicklungsumgebung, Front- und Backend-Software und Relationale Datenbank in einem[17].. 7

Microsoft SQL Server MS SQL Server ist das RDBMS von Microsoft[18]. 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, 89

MongoDB MongoDB ist eine dokumentenorientierte NoSQL-Datenbank, die zum ersten Mal 2007 veröffentlicht wurde[61].. 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 Spaltenorientierte Datenbanken. Viele der grossen Datenbanklösungen wie Oracle Database oder Microsoft SQL Server sind NoSQL Datenbanken resp. bieten diese Option an.. 7, 87, 88, 89, 91

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[30, 29]. 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, 35, 89

PKI . 4

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, 35, 50, 51, 52, 59

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, 51, 90

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. . 51, 88

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[24].. 4, 59

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[64]. Redis wurde 2009 zum ersten Mal veröffentlicht.. 7, 88

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[22, 53, 31]. 4, 5, 16, 20

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[65].. 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[68, 58]. 7

Split-brain Im Kapitel ?? werden die ursachen und folgenden eines Split-brains genauer besprochen. . 33, 90

Splunk Splunk ist Big Data Plattform, Monitoring- und Security-Tool in einem[39, 46]. . 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[62].. 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[63, 37].. 16

Transaktion Eine Transaktion ist beinhaltet Schreib-, Lese-, Mutatations- oder Löschoperationen auf Daten.. 31, 50

UNIX Die erste Version von UNIX wurde im Jahr 1969 in den Bell Labs entwickelt und übernahm viele Komponenten aus dem gescheiterten Multics-Projekt. Aus dem Ursprünglichen UNIX entstanden im Laufe der Zeit viele offene und Proprietäre Derivate deren Einfluss weit über die Welt der Informatik reicht.. 4

VRRP VRRP . 4, 88

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	Anorderungskatalog	Anorderungskatalog erarbeiten			
22.02.2024	16:00	17:30	1.5	Evaluation	Anorderungskatalog	Anorderungskatalog 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	Anorderungskatalog	Anorderungskatalog 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.
11.03.2024	12:00	13:30	1.5	Dokumentation	-	Dokumentation erweitern			pgpool II fällt somit direkt aus der Betrachtung raus, da kein Vergleich möglich ist.
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		
17.03.2024	20:15	21:00	0.8	Dokumentation	-	Dokumentation erweitern	Listings sauber machen.		
18.03.2024	14:00	16:00	2.0	Dokumentation	-	Dokumentation erweitern	Neue Listing-Sprache für yaml-Files erstellt, da noch einige folgen werden.		
18.03.2024	20:20	21:50	1.5	Evaluation	Vorbereitung Benchmarking	Dokumentation erweitern	pgbench analysieren	Statusbericht 2 fertig Schreiben und Mail an Norman Süssstrunk senden	
19.03.2024	08:00	10:00	2.0	Evaluation	Analyse PostgreSQL HA Cluster Lösungen	yugabyteDB	Percona ist Dein Freund		
19.03.2024	14:00	16:00	2.0	Dokumentation	-	Dokumentation erweitern	yugabyteDB		

TABLE 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	<ul style="list-style-type: none"> - Darf eine Vorauswahl stattfinden, um den Aufwand zur reduzieren? 	<ul style="list-style-type: none"> - Eine Vorauswahl ist Sinnvoll und in diesem Rahmen fast zwingend Notwendig, da sonst viel zuviel Zeit investiert werden müsste 	<ul style="list-style-type: none"> - 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 	<ul style="list-style-type: none"> - 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		Norman Süssstrunk	-	Michael Graber	<ul style="list-style-type: none"> - 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 	- Protokoll genehmigen		

TABLE II: Fachgespräche - Protokoll

==

III Kommentare / Anmerkungen

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

Woche	Beschreibung / Event / Problem
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.

TABLE 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
9 def load_configuration():
10     zotero_bibtex_config = dict()
11     zotero_conf_filename = 'zotero_bibtex_configuration.json'
12     zotero_conf_dir = os.path.join(os.path.dirname(os.getcwd()), 'source', 'configuration')
13     # zotero_conf_dir = os.path.join(os.getcwd(), 'src', 'content')
14     json_path = os.path.join(zotero_conf_dir, zotero_conf_filename)
15
16     with open(json_path) as json_string:
17         zotero_bibtex_config = json.load(json_string)
18
19     return zotero_bibtex_config
20 def download_zotero_datas(URL, API_KEY):
21     zotero_result = list()
22     response = requests.get(URL, headers={'Zotero-API-Key': API_KEY})
23     response = response.json()
24     zotero_raw = json.dumps(response, ensure_ascii=False) # json.loads(response)
25     zotero_result = json.loads(zotero_raw)
26     return zotero_result
27
28 def get_data(zotero_bibtex_config):
29     # result_limit = 100
30     # access_type = 'groups'
31     # zotero_access_id = '5245833'
32     # collection_id = 'USSFDCEH'
33     result_limit = int(zotero_bibtex_config.get('result_limit'))
34     access_type = zotero_bibtex_config.get('access_type')
35     zotero_access_id = zotero_bibtex_config.get('zotero_access_id')
36     collection_id = zotero_bibtex_config.get('collection_id')
37     API_KEY = zotero_bibtex_config.get('api_key')
38     zotero_data = list()
39     URL = 'https://api.zotero.org/' + str(access_type) + '/' + str(
40         zotero_access_id) + '/collections/' + str(
41             collection_id) + '/items?limit=1&format=json&sort=dateAdded&direction=asc'
42     # API_KEY = '6Xgb3XhGjQXwA8NuZgu3bw3s'
43     response = requests.get(URL, headers={'Zotero-API-Key': API_KEY})
44

```

```

45     header_dict = response.headers
46     total_elems = int(header_dict.get('Total-Results'), 0)
47
48     if total_elems < result_limit:
49         URL_ALL_ITEMS = 'https://api.zotero.org/' + str(access_type) + '/' + str(
50             zotero_access_id) + '/collections/' + str(collection_id) + '/items?
51             limit=' + str(
52                 result_limit) + '?format=json&sort=dateAdded&direction=asc'
53             zotero_result = downlaod_zotero_datas(URL_ALL_ITEMS, API_KEY)
54
55             zotero_data.extend(zotero_result)
56     else:
57         runs = int(math.ceil(total_elems / result_limit))
58         index = 0
59         start_index = 0
60         while index < runs:
61             URL_Separated = 'https://api.zotero.org/' + str(access_type) + '/' +
62             str(
63                 zotero_access_id) + '/collections/' + str(collection_id) + '/items?
64             limit=' + str(
65                 result_limit) + '?format=json&sort=dateAdded&direction=asc' + '&
66             start=' + str(start_index)
67             zotero_result = downlaod_zotero_datas(URL_Separated, API_KEY)
68
69             zotero_data.extend(zotero_result)
70
71             start_index += result_limit
72             index += 1
73
74
75     return zotero_data
76
77 def convert_to_datetime(input_str, parserinfo=None):
78     return parse(input_str, parserinfo=parserinfo)
79 def get_dates(date, bibtex_item_type, bibtex_month_attributes):
80     dated_date = convert_to_datetime(date)
81     return_value = dict()
82     if bibtex_item_type in bibtex_month_attributes:
83         year = dated_date.year
84         month = dated_date.month
85         return_value = {'year': year, 'month': month}
86     else:
87         year = dated_date.year
88         return_value = {'year': year}
89
90     return return_value
91
92 def split_creators(creators):
93     if creators != []:
94

```

```
89
90     creatorlist = ''
91     for index, creator in enumerate(creators):
92         type = creator.get('creatorType')
93         firstname = creator.get('firstName')
94         lastname = creator.get('lastName')
95         name = creator.get('name')
96         if type == 'author':
97
98             if name and not (firstname or lastname):
99                 creatorlist = creatorlist + name
100                if index != len(creators) - 1:
101                    creatorlist = creatorlist + ', and '
102                else:
103                    creatorlist = creatorlist + lastname + ', ' + firstname
104                    if index != len(creators) - 1:
105                        creatorlist = creatorlist + ', and '
106            else:
107                creatorlist = 'unknown author'
108
109 bib_entry = 'author=' + '\"' + creatorlist + '\"'
110
111 return bib_entry
112
113
114 def write_bibliography(zotero_data, zotero_bibtex_config):
115     # file_json = 'keystore.json'
116     file_json = 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()), 'source', 'configuration')
119     tex_dir = os.path.join(os.path.dirname(os.getcwd()), keystore_path)
120     # tex_dir = os.path.join(os.getcwd(), 'src', 'content')
121     json_path = os.path.join(tex_dir, file_json)
122
123     with open(json_path) as json_string:
124         zotero_bibtex_keys = json.load(json_string)
125
126     zotero_bibtex_keys_specials = {
127         'thesis': {'phdthesis': ['dissertation', 'phd', 'doctorial', 'doctor', 'doktor', 'doktorarbeit'],
128                     'masterthesis': ['ma', 'master', 'masters']},
129     }
130     zotero_bibtex_attributes_special = {
131         'date': 'get_dates',
132         'creators': 'split_creators',
133     }
134     bibtex_month_attributes = ['booklet', 'mastersthesis', 'phdthesis', '
```

```

techreport']

# Bibliography
# tex_dir = os.path.join(os.path.dirname(os.getcwd()), 'source')
bibtex_path = zotero_bibtex_config.get('bibtex_filepath')
tex_dir = os.path.join(os.path.dirname(os.getcwd()), bibtex_path)
# tex_dir = os.path.join(os.getcwd(), 'src', 'content')
# file_name = 'Datenbank_Projektauftrag_Michael_Graber.bib'
file_name = zotero_bibtex_config.get('bibtex_filename')

file_path = os.path.join(tex_dir, file_name)

# bib_datas = BibliographyData()
listKeys = list()
bib_data = ''
for zotero_items in zotero_data:
    biblio_item = zotero_items.get('data')
    itemkeys = biblio_item.keys()
    listKeys.extend(biblio_item.keys())
    zotero_item_key = biblio_item.get('key')
    zotero_item_title = biblio_item.get('title')
    zotero_item_nameofact = biblio_item.get('nameOfAct')
    zotero_item_nameofcase = biblio_item.get('caseName')
    zotero_item_subject = biblio_item.get('subject')
    zotero_item_type = biblio_item.get('itemType')

    # some item types have no titles
    # set the special names instead of the title
    if zotero_item_title:
        bibtex_item_titel = zotero_item_title
    else:
        if zotero_item_type == 'statute':
            biblio_item['title'] = zotero_item_nameofact
            bibtex_item_titel = zotero_item_nameofact
        elif zotero_item_type == 'case':
            biblio_item['title'] = zotero_item_nameofcase
            bibtex_item_titel = zotero_item_nameofcase
        elif zotero_item_type == 'email':
            biblio_item['title'] = zotero_item_subject
            bibtex_item_titel = zotero_item_subject

    if zotero_item_type == 'thesis':
        master_list = zotero_bibtex_keys_specials.get(zotero_item_type).get(
            'masterthesis')
        phd_list = zotero_bibtex_keys_specials.get(zotero_item_type).get(
            'phdthesis')

    # First Master thesis
    if any(item in bibtex_item_titel for item in master_list):

```

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

```

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

```

```

259
260
261 zotero_bibtex_config = load_configuration()
262 zotero_data = get_data(zotero_bibtex_config)
263 write_bibliography(zotero_data, zotero_bibtex_config)

```

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

V riskmatrix.py

```

1 import matplotlib.pyplot as plt
2 import numpy as np
3 import pip as pd
4 import os
5 import csv
6 import pandas as pd
7
8 def riskmatrix(risk, conf, matrix):
9     # get the risk datas
10    risk_conf = conf.get(risk)
11    startpath = risk_conf.get('startpath')
12    destination = risk_conf.get('destination')
13    imagename = risk_conf.get('imagename')
14    datafilename = risk_conf.get('datafilename')
15    itemname = risk_conf.get('itemname')
16    x_axis_title = risk_conf.get('x-axis-title')
17    y_axis_title = risk_conf.get('y-axis-title')
18    title = risk_conf.get('title')
19    bubble_standard_size = int(risk_conf.get('bubble-standard-size'))
20
21    if startpath == 'homendir':
22        directory = os.path.join(os.getcwd(), destination)
23    else: # parentdir
24        directory = os.path.join(os.path.dirname(os.getcwd()), destination)
25
26    print(directory)
27
28    # get the Datas as dict
29    data_path = os.path.join(directory, datafilename)
30    image_path = os.path.join(directory, imagename)
31
32    # load datas from csv into dict
33    with open(data_path) as f:
34        csv_list = [[val.strip() for val in r.split(",")] for r in f.readlines()]
35
36    (_, *header), *data = csv_list
37    datas = {}

```

```
38     for row in data:
39         key, *values = row
40         datas[key] = {key: value for key, value in zip(header, values)}
41
42 # fig_dir = os.path.join(os.path.dirname(os.getcwd()), 'src', 'source')
43 fig = plt.figure()
44 plt.subplots_adjust(wspace=0, hspace=0)
45 plt.xticks([])
46 plt.yticks([])
47 plt.xlim(0, 5)
48 plt.ylim(0, 5)
49 plt.xlabel(x_axis_title)
50 plt.ylabel(y_axis_title)
51 plt.title(title)
52
53 #This example is for a 5 * 5 matrix
54 nrows=5
55 ncols=5
56 axes = [fig.add_subplot(nrows, ncols, r * ncols + c + 1) for r in range(0, nrows) for c in range(0, ncols) ]
57
58 # remove the x and y ticks
59 for ax in axes:
60     ax.set_xticks([])
61     ax.set_yticks([])
62     ax.set_xlim(0,5)
63     ax.set_ylim(0,5)
64
65 #Add background colors
66 #This has been done manually for more fine-grained control
67 #Run the loop below to identify the indice of the axes
68
69 #Identify the index of the axes
70 green = [10, 15, 16, 20, 21] #Green boxes
71 yellow = [0, 5, 6, 11, 17, 22, 23] #yellow boxes
72 orange = [1, 2, 7, 12, 13, 18, 19, 24] # orange boxes
73 red = [3, 4, 8, 9, 14] #red boxes
74
75 for _ in green:
76     axes[_].set_facecolor('green')
77
78 for _ in yellow:
79     axes[_].set_facecolor('yellow')
80
81 for _ in orange:
82     axes[_].set_facecolor('orange')
83
84 for _ in red:
```

```
85     axes[_].set_facecolor('red')
86
87
88 #Add labels to the Green boxes
89 # axes[10].text(0.1,0.8, '4')
90 # axes[15].text(0.1,0.8, '2')
91 # axes[20].text(0.1,0.8, '1')
92 # axes[16].text(0.1,0.8, '5')
93 # axes[21].text(0.1,0.8, '3')
94
95 #Add labels to the Yellow boxes
96 # axes[0].text(0.1,0.8, '11')
97 # axes[5].text(0.1,0.8, '7')
98 # axes[6].text(0.1,0.8, '12')
99 # axes[11].text(0.1,0.8, '8')
100 # axes[17].text(0.1,0.8, '9')
101 # axes[22].text(0.1,0.8, '6')
102 # axes[23].text(0.1,0.8, '10')
103
104 #Add lables to the Orange boxes
105 # axes[1].text(0.1,0.8, '16')
106 # axes[2].text(0.1,0.8, '20')
107 # axes[7].text(0.1,0.8, '17')
108 # axes[12].text(0.1,0.8, '13')
109 # axes[13].text(0.1,0.8, '18')
110 # axes[18].text(0.1,0.8, '14')
111 # axes[19].text(0.1,0.8, '19')
112 # axes[24].text(0.1,0.8, '15')
113
114 #Add lables to the Red Boxes
115 # axes[3].text(0.1,0.8, '23')
116 # axes[8].text(0.1,0.8, '21')
117 # axes[4].text(0.1,0.8, '25')
118 # axes[9].text(0.1,0.8, '24')
119 # axes[14].text(0.1,0.8, '22')
120
121 # run throuh datas and generate axis datas
122 dict_bubble_axis = dict()
123 bubble_axis = list()
124 for datasets in datas:
125     # get the datas
126     riskid = datas.get(datasets).get('risk-id')
127     x_axis = int(datas.get(datasets).get('x-axis'))
128     y_axis = int(datas.get(datasets).get('y-axis'))
129     axis_point = matrix.get((x_axis, y_axis))
130     x_axis_text = float(datas.get(datasets).get('x-axis-text'))
131     y_axis_text = float(datas.get(datasets).get('y-axis-text'))
132     x_axis_bubble = float(datas.get(datasets).get('x-axis-bubble'))
```

```

133     y_axis_bubble = float(datas.get(datasets).get('y-axis-bubble'))
134     bubble_axis.append(axis_point)
135
136     # merge riks if two or more risks share the same axispoint
137     if dict_bubble_axis.get(axis_point):
138         risktag = dict_bubble_axis.get(axis_point).get('risk')
139         risktag = risktag + '/' + riskid
140         x_axis_text = x_axis_text + 0.25
141         y_axis_text = y_axis_text - 0.5
142         bubble_size = bubble_standard_size * 2
143     else:
144         risktag = itemname + riskid
145         bubble_size = bubble_standard_size
146     dict_axis_value = dict()
147
148     dict_axis_value['risk'] = risktag
149     dict_axis_value['x-axis-text'] = x_axis_text
150     dict_axis_value['y-axis-text'] = y_axis_text
151     dict_axis_value['x-axis-bubble'] = x_axis_bubble
152     dict_axis_value['y-axis-bubble'] = y_axis_bubble
153     dict_axis_value['size'] = bubble_size
154     dict_bubble_axis[axis_point] = dict_axis_value
155
156     # cleanup the list, remove duplicated entries
157     bubble_axis = set(bubble_axis)
158
159     # plot the bubbles and texts in the bubbles
160     for axispoint in bubble_axis:
161         axes[axispoint].scatter(dict_bubble_axis[axispoint]['x-axis-bubble'],
162         dict_bubble_axis[axispoint]['y-axis-bubble'], dict_bubble_axis[axispoint][
163         'size'], alpha=1)
164         axes[axispoint].text(dict_bubble_axis[axispoint]['x-axis-text'],
165         dict_bubble_axis[axispoint]['y-axis-text'], s=dict_bubble_axis[axispoint][
166         'risk'], va='bottom', ha='center')
167
168     # save the plot as image
169     plt.savefig(image_path)
170
171 """
172
173 Config File:
174     1. Name
175     2. Startpoint Directory
176     3. Destination Dir
177     4. Alternate Path
178     5. Data File Name
179
180 Data File:
181     1. Spalte: Nummer
182     2. x-achse

```

```
177     3. x-achse
178 """
179
180 """
181     Matrix
182     This Matrix translate the x/y axis from a given risk matrix csv to the
183     axispoint.
184
185     The key of each axispoint is an integer tupel (x, y)
186     So, you can access the axis point this way:
187     <axispoint> = matrix.get((<x_axis>, <y_axis>))
188 """
189 matrix = {
190     # first column
191     (1, 1):20,
192     (1, 2):15,
193     (1, 3):10,
194     (1, 4):5,
195     (1, 5):0,
196     # second column
197     (2, 1):21,
198     (2, 2):16,
199     (2, 3):11,
200     (2, 4):6,
201     (2, 5):1,
202     # third column
203     (3, 1): 22,
204     (3, 2): 17,
205     (3, 3): 12,
206     (3, 4): 7,
207     (3, 5): 2,
208     # fourth column
209     (4, 1): 23,
210     (4, 2): 18,
211     (4, 3): 13,
212     (4, 4): 8,
213     (4, 5): 3,
214     # fifth column
215     (5, 1): 24,
216     (5, 2): 19,
217     (5, 3): 14,
218     (5, 4): 9,
219     (5, 5): 4
220 }
221 # load the configuration file
222 riskmatrix_conf_filename = 'conf.csv'
223 riskmatrix_conf_dir = 'source/configuration/'
```

```

224 conf_riskmatrix_path = os.path.join(os.path.dirname(os.getcwd()),  
    riskmatrix_conf_dir)  
225 conf_csv_path = os.path.join(conf_riskmatrix_path, riskmatrix_conf_filename)  
226 with open(conf_csv_path) as f:  
227     csv_list = [[val.strip() for val in r.split(",")] for r in f.readlines()]  
228  
229 (_, *header), *data = csv_list  
230 conf = {}  
231 for row in data:  
232     key, *values = row  
233     conf[key] = {key: value for key, value in zip(header, values)}  
234  
235 for risks in conf:  
236     riskmatrix(risks, conf, matrix)  
237 # data = pd.read_csv('/home/itgramic/LaTeX/riskmatrix/src/source/riskmatrixproblem  
    .csv', header=None, dtype={0: str}).set_index(0).squeeze().to_dict()

```

Listing 2: Python LaTex - riskmatrix.py - Risikomatrizen

VI cost_benefit_diagram.py

```

1 import matplotlib.pyplot as plt  
2 import os  
3 import csv  
4 import pandas as pd  
5 import yaml  
6  
7 # Get the Configuration  
8 def load_configuration():  
9     cost_benefit_config = dict()  
10    cbd_conf_filename = 'scatter_plotter_conf.yaml'  
11    cbd_conf_dir = os.path.join(os.path.dirname(os.getcwd()), 'source', '  
        configuration')  
12    yaml_path = os.path.join(cbd_conf_dir, cbd_conf_filename)  
13  
14    with open(yaml_path, "r") as file:  
15        cost_benefit_config = yaml.load(file, Loader=yaml.FullLoader)  
16  
17    return cost_benefit_config  
18 # Get the Datas  
19 def get_data(cost_benefit_config):  
20    # Config Variables  
21    startpath = cost_benefit_config.get('startpath')  
22    destination = cost_benefit_config.get('desitination_path')  
23    datafilename = cost_benefit_config.get('datafile')  
24  
25    if startpath == 'homedir':

```

```

26     directory = os.path.join(os.getcwd(), destination)
27 else: # parentdir
28     directory = os.path.join(os.path.dirname(os.getcwd()), destination)
29
30 # get the Datas as dict
31 data_path = os.path.join(directory, datafilename)
32
33 # load datas from csv into dict
34 with open(data_path) as f:
35     csv_list = [[val.strip() for val in r.split(",")] for r in f.readlines()]
36
37 (_, *header), *data = csv_list
38 datas = {}
39 for row in data:
40     key, *values = row
41     datas[key] = {key: value for key, value in zip(header, values)}
42
43 cost_benefit_data = {}
44 for key, value in datas.items():
45     variant_name = value['variant_name']
46     x_axis = int(value['x-axis'])
47     y_axis = int(value['y-axis'])
48     cost_benefit_data[variant_name] = (x_axis, y_axis)
49
50 return cost_benefit_data
51
52 # Plot the Datas
53 def cost_benefit_diagram(cost_benefit_config, cost_benefit_data):
54     # Config Variables
55     startpath = cost_benefit_config.get('startpath')
56     destination = cost_benefit_config.get('destination_path')
57     imagename = cost_benefit_config.get('imagename')
58
59     if startpath == 'homedir':
60         directory = os.path.join(os.getcwd(), destination)
61     else: # parentdir
62         directory = os.path.join(os.path.dirname(os.getcwd()), destination)
63
64 # get the Datas as dict
65 data_path = os.path.join(directory, imagename)
66
67 # Extract the Datas
68 labels, values = zip(*cost_benefit_data.items())
69 x, y = zip(*values)
70
71 # Create Scatter-Diagram
72 plt.scatter(x, y, color=cost_benefit_config.get('scatter-point-color'))
73

```

```

74     # X-Lines
75     plt.axhline(y=cost_benefit_config.get('y-axis-line-pos'), color=
76     cost_benefit_config.get('y-axis-line-color'), linestyle=cost_benefit_config.
77     get('y-axis-line-type'), label=cost_benefit_config.get('y-axis-line-label'))
78
79     # Y-Lines
80     plt.axvline(x=cost_benefit_config.get('x-axis-line-pos'), color=
81     cost_benefit_config.get('x-axis-line-color'), linestyle=cost_benefit_config.
82     get('x-axis-line-type'), label=cost_benefit_config.get('x-axis-line-label'))
83
84     # Add Labels
85     plt.xlabel(cost_benefit_config.get('x-axis-title'))
86     plt.ylabel(cost_benefit_config.get('y-axis-title'))
87     plt.title(cost_benefit_config.get('title'))
88
89     # Labling Data Points
90     for label, x_point, y_point in zip(labels, x, y):
91         plt.text(x_point, y_point, label)
92
93     # Show Grid
94     plt.grid(True)
95
96     # Save Diagram as PNG
97     plt.savefig(data_path)
98
99     cost_benefit_config = load_configuration()
100    cost_benefit_data = get_data(cost_benefit_config)
101    cost_benefit_diagram(cost_benefit_config, cost_benefit_data)

```

Listing 3: Python LaTex - cost_benefit_diagram.py - Kosten-Nutzen-Diagramm

VII cssscatter_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 4: Python LaTex - csscatter_plotter_conf.yaml - Konfigurationsdatei - Kosten-Nutzen-Diagramm

VIII pandas_dataframe_to_latex_table.py

```

1 import os
2 import pandas as pd
3 import yaml
4 from pathlib import Path
5 import chardet
6
7 import csv
8
9 # Get the Configuration
10 def load_configuration(plt_conf_filename):
11     panda_latex_tables_config = dict()
12     plt_conf_dir = os.path.join(os.path.dirname(os.getcwd()), 'source', 'configuration')
13     yaml_path = os.path.join(plt_conf_dir, plt_conf_filename)
14
15     with open(yaml_path, "r") as file:
16         panda_latex_tables_config = yaml.load(file, Loader=yaml.FullLoader)
17
18     return panda_latex_tables_config
19
20
21 def get_data(startpath, destination, tablefilename, datafile_path, datafile,
22             alternative_csv_load, separator, decimal):
23     # Config Variables
24     if startpath == 'homedir':
25         directory = os.path.join(os.getcwd(), datafile_path)
26     else: # parentdir
27         directory = os.path.join(os.path.dirname(os.getcwd()), datafile_path)
28
29     # get the Datas as dict
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     # if alternative_csv_load:
36     #     with open(data_path, 'r', encoding=encoding) as file:
37     #         reader = csv.reader(file)
38     #         data = list(reader)
39     #
40     #         # panda_table_data = pd.DataFrame(data, columns=[0])
41     #         # panda_table_data = pd.read_csv(data_path, sep=separator, decimal=
42     #         decimal, encoding=encoding, lineterminator='\n', engine='python')
43     #         panda_table_data = pd.read_csv(data_path, sep=separator, decimal=decimal
44     , encoding=encoding, lineterminator='\n')
45     #         df_dtype = {
46     #             "Nr.": int,
47     #             "Anforderung": str,
48     #             "Beschreibung": str,
49     #             "System": str,
50     #             "Muss / Kann": str
51     #         }
52     #         # panda_table_data = pd.read_csv(data_path, sep=",", decimal=".",
53     encoding=encoding, lineterminator='\n', dtype=df_dtype)
54     #         # panda_table_data = pd.read_csv(data_path, sep=",", decimal=".",
55     encoding=encoding)
56     # else:
57     #     panda_table_data = pd.read_csv(data_path, sep=separator, decimal=decimal
58     , encoding=encoding)
59     # panda_table_data = pd.read_csv(data_path, sep=",", decimal=".",
60     encoding=encoding, low_memory=False, engine='python')
61     # panda_table_data = pd.read_csv(data_path, sep=",", decimal=".",
62     encoding=encoding, engine='python', dtype='unicode')
63     # readed = open(data_path, 'r', encoding=encoding)
64     # panda_table_data = pd.read_csv(open(data_path, 'r', encoding=encoding), sep
65     =",", decimal=".",
66     encoding=encoding)
67     # panda_table_data = pd.read_csv(data_path, sep=",", decimal=".",
68     encoding=encoding, encoding="ISO-8859-1")
69     # panda_table_data = pd.read_csv(data_path, sep=",", decimal=".",
70     encoding=encoding, chunkszie=10)

71     # for chunk in pd.read_csv(data_path, sep=",", decimal=".",
72     encoding=encoding, chunkszie=5):
73     #     print(chunk)
74     # panda_table_data = pd.DataFrame()
75     # temp = pd.read_csv(data_path, iterator=True, sep=",",
76     decimal=".",
77     encoding=encoding, chunkszie=1000)
78     # panda_table_data = pd.concat(temp, ignore_index=True)

79     df_dtype = {
80         "Nr.": int,
81         "Anforderung": str,
82         "Beschreibung": str,
```

```
71     "System": str,
72     "Muss / Kann": str
73 }
74 # panda_table_data = pd.read_csv(data_path, sep=",", decimal=".",
75 # encoding, engine='python', dtype=df_dtype)
76 # panda_table_data = pd.read_csv(data_path, sep=",", decimal=".",
77 # encoding, engine=df_dtype)
78
79 # import dask.dataframe as dd
80 # df = dd.read_csv(data_path, sep=",", decimal=".",
81 # encoding=encoding)
82 # panda_table_data = df
83 print(encoding)
84 panda_table_data = pd.read_csv(data_path, sep=separator, decimal=decimal,
85 encoding=encoding)
86 # return data
87
88 return panda_table_data
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
```

def create_latex_tables(panda_latex_tables_config):
 plt_tables = panda_latex_tables_config.get('tables_inventory')
 for table_item in plt_tables:
 # id and filesystem informations
 table_id = panda_latex_tables_config.get('tables').get(table_item).get('id')
 isbigfile = panda_latex_tables_config.get('tables').get(table_item).get('isbigfile')
 has_longtexts = panda_latex_tables_config.get('tables').get(table_item).get('has_longtexts')
 if isbigfile or has_longtexts:
 alternative_cvs_load = True
 else:
 alternative_cvs_load = False
 startpath = panda_latex_tables_config.get('tables').get(table_item).get('startpath')
 destination = panda_latex_tables_config.get('tables').get(table_item).get('destination_path')
 tablefilename = panda_latex_tables_config.get('tables').get(table_item).get('tablefilename')
 datafile_path = panda_latex_tables_config.get('tables').get(table_item).get('datafile_path')
 datafile = panda_latex_tables_config.get('tables').get(table_item).get('datafile')
 if startpath == 'homedir':
 directory = os.path.join(os.getcwd(), destination)
 else: # parentdir
 directory = os.path.join(os.path.dirname(os.getcwd()), destination)
 tablefile = os.path.join(directory, tablefilename)
 separator = panda_latex_tables_config.get('tables').get(table_item).get('

```
separator')
108     decimal = panda_latex_tables_config.get('tables').get(table_item).get(
109         'decimal')
110
111     # column operations
112     column_operations = panda_latex_tables_config.get('tables').get(table_item).
113         get('column_operations').get('datas')
114
115     # group by / aggregation
116     groupby_values = panda_latex_tables_config.get('tables').get(table_item).
117         get('group_by')
118     group_by_function = panda_latex_tables_config.get('tables').get(table_item).
119         get('group_by_function')
120     # selected_rows = panda_latex_tables_config.get('tables').get(table_item).
121     #     get('selected_rows')
122     agg_funtion = panda_latex_tables_config.get('tables').get(table_item).get(
123         'agg_funtion')
124     agg_columns = panda_latex_tables_config.get('tables').get(table_item).get(
125         'agg_columns')
126     # dropping and renaming columns
127     drop_columns = panda_latex_tables_config.get('tables').get(table_item).get(
128         'drop_columns')
129     rename_columns = panda_latex_tables_config.get('tables').get(table_item).
130         get('rename_columns')
131
132     # table filtering and sorting
133     where_clausel = panda_latex_tables_config.get('tables').get(table_item).
134         get('where_clausel')
135     order_by = panda_latex_tables_config.get('tables').get(table_item).get(
136         'sorting').get('order_by')
137     sort_acending = panda_latex_tables_config.get('tables').get(table_item).
138         get('sorting').get('sort_acending')
139     sort_inplace = panda_latex_tables_config.get('tables').get(table_item).get(
140         'sorting').get('sort_inplace')
141
142     # pivot settings
143     pivot = panda_latex_tables_config.get('tables').get(table_item).get(
144         'pivot')
145     pivot_column = panda_latex_tables_config.get('tables').get(table_item).
146         get('pivot_columns')
147     pivot_value = panda_latex_tables_config.get('tables').get(table_item).get(
148         'pivot_values')
149
150     # pivot_table settings
151     pivot_table = panda_latex_tables_config.get('tables').get(table_item).get(
152         'pivot_table')
153     pivot_table_column = panda_latex_tables_config.get('tables').get(
154         table_item).get('pivot_table').get(
```

```
137     'pivot_columns')
138     pivot_table_value = panda_latex_tables_config.get('tables').get(table_item).
139     .get('pivot_table').get(
140         'pivot_values')
141     pivot_table_agg_function = panda_latex_tables_config.get('tables').get(
142         table_item).get('pivot_table').get(
143             'pivot_agg_func')
144     pivot_table_indexes = panda_latex_tables_config.get('tables').get(
145         table_item).get('pivot_table').get(
146             'pivot_index').get('pivot_indexes')
147     pivot_table_indexes_visible = panda_latex_tables_config.get('tables').get(
148         table_item).get('pivot_table').get(
149             'pivot_index').get('pivot_indexes_visible')
150     pivot_table_rename_indexes = panda_latex_tables_config.get('tables').get(
151         table_item).get('pivot_table').get(
152             'pivot_index').get('pivot_rename_indexes')
153
154     # margins (subtotals)
155     margin = panda_latex_tables_config.get('tables').get(table_item).get(
156         'margins').get('margin')
157     margin_name = panda_latex_tables_config.get('tables').get(table_item).get(
158         'margins').get('margin_name')
159
160     # table settings
161     table_caption = panda_latex_tables_config.get('tables').get(table_item).
162     get('caption')
163     table_label = panda_latex_tables_config.get('tables').get(table_item).get(
164         'label')
165     table_style = panda_latex_tables_config.get('tables').get(table_item).get(
166         'table_styles')
167     sparse_columns = panda_latex_tables_config.get('tables').get(table_item).
168     get('table_styles').get(
169         'sparse_columns')
170     table_caption_position = panda_latex_tables_config.get('tables').get(
171         table_item).get('table_styles').get(
172             'props').get('caption-side')
173     table_position = panda_latex_tables_config.get('tables').get(table_item).
174     get('table_styles').get('props').get(
175         'position')
176     longtable = panda_latex_tables_config.get('tables').get(table_item).get(
177         'table_styles').get('props').get(
178             'longtable')
179     linebreak_columns = panda_latex_tables_config.get('tables').get(table_item).
180     get('table_styles').get('props').get(
181         'linebreak_columns')
182     resize_textwidth = panda_latex_tables_config.get('tables').get(table_item).
183     get('table_styles').get('props').get(
184         'resize_textwidth')
```

```
169     # get the pandas (panda data)
170     panda_table_data = get_data(startpath, destination, tablefilename,
171     datafile_path, datafile, alternative_cvs_load, separator, decimal)
172
173     # filter by where clause
174     if where_clause:
175         panda_table_data = panda_table_data.query(where_clause)
176
177     # Drop unused columns
178     if drop_columns:
179         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_indexes)):
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             else:
203                 panda_table_data = panda_table_data
204
205     # pivot if pivot is selected
206     if pivot_table_column or pivot_table_value or pivot_table_indexes:
207         if type(pivot_table_agg_function) is list:
208             agg_tuple = tuple(pivot_table_agg_function)
209             panda_table_data = pd.pivot_table(panda_table_data, index=
210             pivot_table_indexes,
211                                         columns=pivot_table_column,
212                                         values=pivot_table_value,
213                                         aggfunc=agg_tuple, margins=
214                                         margin, margins_name=margin_name)
215         elif type(pivot_table_agg_function) is dict:
```

```
206     panda_table_data = pd.pivot_table(panda_table_data, index=
207         pivot_table_index,
208         columns=pivot_table_column,
209         values=pivot_table_value,
210         margins=margin, margins_name=
211         margin_name)
212     else:
213         panda_table_data = pd.pivot_table(panda_table_data, index=
214             pivot_table_index,
215             columns=pivot_table_column,
216             values=pivot_table_value,
217             aggfunc=pivot_table_agg_function
218             , margins=margin,
219             margins_name=margin_name)
220
221     # set column operations
222     if column_operations:
223         for column_ops in column_operations:
224             operation_function = panda_latex_tables_config.get('tables').get(
225                 table_item).get('column_operations').get('operations').get(column_ops).get(
226                 'operation_function')
227             operation_columns = panda_latex_tables_config.get('tables').get(
228                 table_item).get('column_operations').get('operations').get(column_ops).get(
229                 'columns')
230             operation_axis = panda_latex_tables_config.get('tables').get(
231                 table_item).get('column_operations').get('operations').get(column_ops).get(
232                 'axis_number')
233             match operation_function:
234                 case 'max':
235                     panda_table_data[column_ops] = panda_table_data[
236                         operation_columns].max()
237                 case 'min':
238                     panda_table_data[column_ops] = panda_table_data[
239                         operation_columns].min()
240                 case 'head':
241                     panda_table_data[column_ops] = panda_table_data[
242                         operation_columns].head()
243                 case 'sum':
244                     panda_table_data[column_ops] = panda_table_data[
245                         operation_columns].sum(axis=operation_axis)
246                 case 'mean':
247                     panda_table_data[column_ops] = panda_table_data[
248                         operation_columns].mean()
249                 case 'diff':
250                     panda_table_data[column_ops] = panda_table_data[
251                         operation_columns[1]] - panda_table_data[operation_columns[0]]
```

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

```

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

```

```

322         latex_table = latex_table.replace(caption_label_nbr, '')
323         latex_table = latex_table.replace(caption_only_nbr, '')
324         latex_table = latex_table.replace(label_only_nbr, '')
325
326     if longtable:
327         table_string = '\\end{longtable}'
328         new_caption = caption_label_nbr + "\n" + table_string
329         latex_table = latex_table.replace(table_string, new_caption)
330     else:
331         table_string = '\\end{table}'
332         new_caption = caption_label_nbr + "\n" + table_string
333         latex_table = latex_table.replace(table_string, new_caption)
334
335
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 5: Python LaTex - pandas_dataframe_to_latex_table.py CSV - LaTex Tabelle

IX 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 tablename: "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_indexes_visible:
54 pivot_rename_indexes:
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_clause:
72     sorting:
73       order_by:
74         - "server"
75         - "rdbms"
76     sort_acending: True
77     sort_inplace: True
78     margins:
79       margin: False
80       margin_name:
81     table_styles:
82       selector: "caption"
83     props:
84       caption-side: "below"
85       position: "H"
86       sparse_columns: True
87       longtable: True
88       resize_textwidth: False
89       linebreak_columns:
90       table_header: True
91   db_inventory_per_rdbms:
92     id: "db_inventory_per_rdbms"
93     isbigfile:
94     has_longtexts: False
95     separator: ","
96     decimal: "."
97     caption: "Datenbankinventar"
98     label: "db_inventory_per_rdbms"
99     startpath: "parentdir"
100    destination_path: "content/latex_tables"
101    datafile_path: "source/tables"
102    datafile: "inventory.csv"
103    tablefilename: "db_inventory_per_rdbms.tex"
104    decimal_format:
105    group_by:
106      - "rdbms"
107    group_by_function: "sum"
108    agg_funtion:
109    agg_columns:
110      - "rdbms"
111    drop_columns:
112      - "server"
113      - "os"
114      - "version"
115      - "releasedate"
116      - "eol"
```

```
117 - "age"
118 - "eol_since"
119 - "comment"
120 column_operations:
121   datas:
122     operations:
123       dauer_summe:
124         operation_function:
125           axis_number:
126             columns:
127             pivot:
128               pivot_columns:
129               pivot_values:
130             pivot_table:
131               pivot_index:
132                 pivot_indizes_visible:
133                 pivot_rename_indizes:
134               pivot_columns:
135               pivot_values:
136               pivot_agg_func:
137             rename_columns:
138             rdbms: "RDBMS"
139             instance : "Instanz"
140             databases : "Datenbanken"
141             appliance: "Appliance"
142             where_clausel:
143             sorting:
144               order_by:
145                 - "rdbms"
146               sort_acending: True
147               sort_inplace: True
148             margins:
149               margin: True
150               margin_name: "Gesamtergebnis"
151             table_styles:
152               selector: "caption"
153             props:
154               caption-side: "below"
155               position: "H"
156               sparse_columns: True
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 tablename: "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_clauses:
224     sorting:
225       order_by:
226         sort_ascending: False
227         sort_inplace: True
228     margins:
229       margin: True
230       margin_name: "Gesamtergebnis"
231     table_styles:
232       selector: "caption"
233     props:
234       caption_side: "below"
235       position: "H"
236       sparse_columns: True
237       longtable: True
238       resize_textwidth: False
239       linebreak_columns:
240       table_header: True
241     anforderungskatalog:
242       id: "anforderungskatalog"
243       isbigfile:
244       has_longtexts: True
245       separator: ";"
246       decimal: "."
247       caption: "Anforderungskatalog"
248       label: "anforderungskatalog"
249       startpath: "parentdir"
250       destination_path: "content/latex_tables"
251       datafile_path: "source/tables"
252       datafile: "anforderungskatalog.CSV"
253       tablefilename: "anforderungskatalog.tex"
254       decimal_format:
255       group_by:
256       group_by_function:
257       agg_funtion:
258       agg_columns:
259       drop_columns:
260       column_operations:
```

```
261     datas:
262     operations:
263     dauer_summe:
264     operation_function:
265     axis_number:
266     columns:
267     pivot:
268     pivot_columns:
269     pivot_values:
270     pivot_table:
271     pivot_index:
272     pivot_indexes_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_function:
315  agg_columns:
316  drop_columns:
317    - "Hide"
318    - "Geplante Dauer [h]"
319    - "dauer_summe"
320  column_operations:
321    datas:
322      operations:
323        dauer_summe:
324          operation_function:
325            axis_number:
326            columns:
327        pivot:
328          pivot_columns:
329          pivot_values:
330        pivot_table:
331          pivot_index:
332            pivot_indexes_visible: False
333            pivot_rename_indexes:
334          pivot_columns:
335          pivot_values:
336          pivot_agg_func:
337        rename_columns:
338        where_clause: "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_funtion:
381   agg_columns:
382     - "Dauer [h]"
383     - "Geplante Dauer [h]"
384     - "dauer_summe"
385   drop_columns:
386     - "Datum"
387     - "Von"
388     - "Bis"
389     - "Hide"
390     - "ÄTtigkeit"
391     - "Bemerkung"
392     - "Schwierigkeit"
393     - "öLsungen"
394   column_operations:
395     datas:
396       - "dauer_summe"
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_indexes_visible: False  
468 pivot_rename_indexes:  
469 pivot_columns:  
470 pivot_values:  
471 pivot_agg_func:  
472 rename_columns:  
473 where_clauses:  
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_funtion:
510 agg_columns:
511 drop_columns:
512 column_operations:
513 datas:
514 operations:
515 dauer_summe:
516 operation_function:
517 axis_number:
518 columns:
519 pivot:
520 pivot_columns:
521 pivot_values:
522 pivot_table:
523 pivot_index:
524 pivot_indexes_visible: False
525 pivot_rename_indexes:
526 pivot_columns:
527 pivot_values:
528 pivot_agg_func:
529 rename_columns:
530 where_clause:
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 - "äAbhngigkeit"
```

```
549     - "Beschreibung"
550     - "Status"
551     - "Risiko"
552     - "Impact"
553     table_header: True
554     predecision_out:
555       id: "predecision_out"
556       isbigfile:
557       has_longtexts: False
558       separator: ";"
559       decimal: "."
560       caption: "Vorauswahl - Ausgeschieden"
561       label: "predecision_out"
562       startpath: "parentdir"
563       destination_path: "content/latex_tables"
564       datafile_path: "source/tables"
565       datafile: "pre-decision.csv"
566       tablefilename: "pre-decision-out.tex"
567       decimal_format:
568       group_by:
569       group_by_function:
570       agg_funtion:
571       agg_columns:
572       drop_columns:
573       - "hide_state"
574       column_operations:
575         datas:
576         operations:
577           dauer_summe:
578             operation_function:
579             axis_number:
580             columns:
581           pivot:
582             pivot_columns:
583             pivot_values:
584             pivot_table:
585             pivot_index:
586             pivot_indizes_visible: False
587             pivot_rename_indizes:
588             pivot_columns:
589             pivot_values:
590             pivot_agg_func:
591             rename_columns:
592             where_clause1: "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_indexizes_visible: False
```

```
645     pivot_rename_indexes:  
646     pivot_columns:  
647     pivot_values:  
648     pivot_agg_func:  
649     rename_columns:  
650     where_clause: "hide_state == 2"  
651     sorting:  
652         order_by:  
653             - "Nr."  
654         sort_ascending: True  
655         sort_inplace: True  
656     margins:  
657         margin: False  
658         margin_name:  
659     table_styles:  
660         selector: "caption"  
661     props:  
662         caption-side: "below"  
663         position: "H"  
664         sparse_columns: True  
665         longtable: False  
666         resize_textwidth: True  
667         linebreak_columns:  
668             - "ÜBegründung"  
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_colums:  
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_indexizes_visible: False
702       pivot_rename_indexizes:
703     pivot_columns:
704       pivot_values:
705       pivot_agg_func:
706     rename_columns:
707     where_clausel:
708     sorting:
709       order_by:
710         - "Woche"
711     sort_acending: True
712     sort_inplace: True
713   margins:
714     margin: False
715     margin_name:
716   table_styles:
717     selector: "caption"
718   props:
719     caption-side: "below"
720     position: "H"
721     sparse_columns: True
722     longtable: False
723     resize_textwidth: True
724     linebreak_columns:
725       - "Beschreibung / Event / Problem"
726     table_header: True
727   evaluation_distributed_sql:
728     id: "evaluation_distributed_sql"
729     isbigfile:
730     has_longtexts: False
731     separator: ";"
732     decimal: "."
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_colums:
745 drop_columns:
746 column_operations:
747 datas:
748 operations:
749 dauer_summe:
750     operation_function:
751     axis_number:
752     columns:
753 pivot:
754     pivot_columns:
755     pivot_values:
756 pivot_table:
757     pivot_index:
758     pivot_indizes_visible: False
759     pivot_rename_indizes:
760 pivot_columns:
761 pivot_values:
762 pivot_agg_func:
763 rename_columns:
764 where_clausel:
765 sorting:
766     order_by:
767     sort_acending: False
768     sort_inplace: False
769 margins:
770     margin: False
771     margin_name:
772 table_styles:
773     selector: "caption"
774 props:
775     caption-side: "below"
776     position: "H"
777     sparse_columns: True
778     longtable: False
779     resize_textwidth: False
780     linebreak_columns:
781     table_header: False
782 expert_discussions_overview:
783     id: "expert_discussions_overview"
784     isbigfile:
785     has_longtexts: False
786     separator: ";"
787     decimal: "."
788     caption: "äFachgespräche"
```

```
789     label: "expert_discussions_overview"
790     startpath: "parentdir"
791     destination_path: "content/latex_tables"
792     datafile_path: "source/tables"
793     datafile: "expert_discussions.csv"
794     tablefilename: "expert_discussions_overview.tex"
795     decimal_format:
796     group_by:
797     group_by_function:
798     agg_funtion:
799     agg_colums:
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_funtion:
860     agg_colums:
861     drop_columns:
862     column_operations:
863     datas:
864     operations:
865     dauer_summe:
866     operation_function:
867     axis_number:
868     columns:
869     pivot:
870     pivot_columns:
871     pivot_values:
872     pivot_table:
873     pivot_index:
874     pivot_indizes_visible: False
875     pivot_rename_indizes:
876     pivot_columns:
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_indexes_visible: False  
936     pivot_rename_indexes:  
937   pivot_columns:  
938   pivot_values:  
939   pivot_agg_func:  
940   rename_columns:  
941   where_clause:  
942   sorting:  
943     order_by:  
944       sort_ascending: 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 6: Python LaTex - csv_to_latex_diplomarbeit.yaml - Konfigurationsdatei - CSV - LaTex-Tabelle