ibW Höhere Fachschule Südostschweiz

Diplomarbeit Technik und Wirtschaftsinformatik 2023-2024

Titel der Arbeit:	PostgreSQL HA Cluster - Konzeption und Implementation
Name:	Graber
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 Ausgangslange . . . 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 Mananagement System
k8s	Kubernetes
HPE	Hewlett Packard Enterprise
HP-UX	Hewlett Packard UNIX
SAP	Systemanalyse Programmentwicklung
SQL	Structured Query Language
DBA	Database Administrator / Datenbankadministrator
HA	High Availability
PRTG	Paessler Router Traffic Grapher
SAN	Storage Area Network
SIEM	Security Information and Event Management
CI/CD	Continuous Integration/Continuous Delivery
SWOT-Analyse	Strengths, Weaknesses, Opportunities, Threats
OLAP	Online Analytical Processing
IaC	Infrastructure as Code
IPERKA	Informieren, Planen, Entscheiden, Realisieren, Kontrollieren, Auswerten
BSI	Bundesamt für Sicherheit in der Informationstechnik
VRRP	Virtual Router Redundancy Protocol
PKI	Private Key Infrastructure

DCS	Distributed Configuration Store
DQL	Data Query Language
DML	Data Manipulation Language
ACID	Atomicity, Consistency, Isolation und Durability

1 Einleitung

1.1 Ausgangslage und Problemstellung

1.1.1 Das Kantonsspital Graubünden

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

Abbildung 1.1: Spitalregionen Kanton Graubünden[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[56]

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.

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. Entsprechen 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.

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 Ausgangslange

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						Abschätzung		Behandlung			
								Massnahmen ergreifen?	Zielwert		Massnahme
ID	Schutzziel	Referenz BSI 200-3	Risiko	Beschreibung / Ursache	Auswirkung	WS	SM		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. Hierdurch entsteht das Risiko, das Systeme Ausfallen	Sofern 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	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. Hierdurch entsteht das Risiko, das Systeme Fehlfunktionen erleiden. Allerdings versuchen Datenbanksysteme, die Auswirkungen so gering wie möglich zu halten.	Fehlfunktionen können innerhalb von Datenbanksystemen die Datenkonsistenz verletzen, Daten können verloren gehen oder ungewollt von Dritten und unberechtigten Personen eingesehen werden. Systeme könnten nicht mehr oder nur noch eingeschränkt verfügbar werden. Daher sind sowohl Vertraulichkeit, Integrität und Verfügbarkeit gefährdet.	2	4	Ja	2	2	Systeme zentralisieren Lifecycle etablieren
4	C, I, A	G0.27-1	Ressourcenmangel (personelle Ressourcen)	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. Dadurch bleiben Fehler länger unentdeckt, Hotfixes, Patches, Updates und Upgrades können nicht oder nicht zur richtigen Zeit eingespielt werden. Bei einem akuten Problemfall ist nicht garantiert, das die Leute erreichbar sind, die notwendig sind	Die Auswirkungen können vielfältig sein, abhängig davon welcher Aspekt unter dem Ressourcenmangel leidet. Grundsätzlich wird aber sowohl die Vertraulichkeit, Integrität und Verfügbarkeit gefährdet.	3	3	Ja	2	3	Systeme zentralisieren
5	A	G0.27-2	Ressourcenmangel (technische Ressourcen)	Kann auftreten wenn Ressorucenwachstum 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.	Wenn die CPU- und Memory-Usage über einen gewissen Schwellwert geht, fängt das Betriebssystem an zu Priorisieren. Dies wird primär der Endanwender in form von Performance Einbussen bemerken. Im schlimmsten Fall steht eine Anwendung nicht mehr zur Verfügung. Gefährlicher sind Storage Overflows, besonders wenn die Datenbank nicht mehr alle Informationen schreiben konnte, die sie für einen korrekten Neustart benötigte.	2	2	Ja	1	2	Monitoring verschärfen
6	C, I, A	G0.31	Fehlerhafte Nutzung oder Administration von Geräten und Systemen	Durch die Vielfalt an Datenbankversionen und Betriebssystemen und Plattformen worauf diese betrieben werden, besteht allen voran das Risiko einer Fehlerhafter Administration und Konfiguration.	Doch die folgen bleiben nichtsdesto trotz 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. Daraus erschliesst sich das auch bei diesem Risiko die Vertraulichkeit, Integrität und Verfügbarkeit gefährdet ist.	4	3	Ja	2	3	Systeme zentralisieren
7	C, I, A	G0.32	Missbrauch von Berechtigungen	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. Multipliziert mit der Anzahl der unterschiedlichsten Datenbanken, Betriebssystemen und Applikationen entsteht das Risiko, das Berechtigungen Wissendlich oder Unwissendlich 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
8	A, I	G0.45	Datenverlust	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. Hinzu kommt, das für die meisten Datenbanken hohe Sicherungsintervalle von einmal pro Stunde oder gar nur einmal am Tag gewählt wurde. Ein weiterer Aspekt des Risikos besteht in der tatsache, das aufgrund der grossen Anzahl Datenbanken und deren Heterogenität nur wenige Backups auch wirklich regelmässig geprüft werden.	Aus dem Risiko ergeben sich zwei Auswirkungen, die aber beide ein hohes Mass an Schaden verursachen können. 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

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, 46] 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.	Hoch
2	Failover Restore	Nach einem Failover muss der DB-Node wieder vom Standby-Node auf den Primären Node zurückgesetzt werden. 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	Hoch
3	Filesystem Management	Die Datenmenge von Datenbanken wachsen in der Regel beständig. Die Belegung von Tablespaces 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 Tablespaces genügend Platz haben. Dazu gehört auch das Überwachen des Logs und entsprechende Schritte im Fehlerfall.	Mittel
5	Statistiken / Cleanup Jobs justieren	PostgreSQL sammelt Statistiken um SQL Queries optimaler ausführen zu können. Zudem wird im Rahmen des gleichen Sheduled 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 inperfomante 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 aufgebläht 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 Bedürfnis 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.	1	Muss	
3	Filesystem Management	Hoch	Das Filesystem muss beim erreichen von 95% Usage automatisiert vergrössert werden. 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	2	Muss	
6	SQL optimierungen	Tief	Es gibt SQL Abfragen, mit dem fehlende Indizes ermittelt werden können. Diese Indizes sollen automatisiert erstellt werden. 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].	2	Kann	
7	Health Checks und Aktionen (Maintenance)	Hoch	Tabellen und Indizes können sich aufblähen (bloaded table / bloaded index) 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 evaluirt 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 evaluirt 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, bloaded 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 bloaded 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	Hoch
13	Testziele	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

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	
Sekundäre Linux Distributionen	Rocky Linux Oracle Linux RedHat Enterprise Linux (RedHat Enterpise Linux (RHEL))	RedHat Enterprise Linux (RedHat Enterpise Linux (RHEL)), Rocky Linux oder Oracle Linux wird nur eingesetzt, wenn es nicht anders möglich ist
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. Produkt zurzeit nicht definiert
Usermanagement	Microsoft Active Directory	

Tabelle 1.8: Gegebene Systeme

Daraus ergeben sich nach nach Züst, Troxler 2002[70] 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 dass 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 Teils Sicherungen mehr von Konfigurationsfiles, Scripts usw.
6	PKI	Key Management	Es braucht einen PKI um Keys und Zertifikate handeln zu können	Bestehender PKI wird abgelöst. Ablösungsprojekt in der Initialisierungsphase. Bestehender PKI nicht für Zertifikate im Einsatz	Es steht kein moderner PKI im Einsatz.	Zertifikate können aus Zeitgründen nicht in der Evaluationsphase eingesetzt werden. Für die Testphase müssen Zertifikate manuell ausgestellt werden.

Tabelle 1.9: Abhängigkeiten

1.5 Risikomanagement

Aus den Abhängigkeiten heraus wurden folgende Risiken identifiziert:

Identifikation				Abschätzung		Behandlung			
						Massnahmen ergreifen?	Zielwert		Massnahme
ID	Risiko	Beschreibung / Ursache	Auswirkung	WS	SM		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 Selbstorganisation	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.	Logs müssen länger auf dem System selber vorgehalten werden. Zudem müssen ggf. eigene Massnahmen zum Auslesen von Logs getroffen werden	4	1	Nein			
8	Foreman nicht betriebsbereit	Die Foreman Provisioning- und Lifecycle Plattform befindet sich aktuell erst in der Proof of Concept Phase. Dadurch besteht das Risiko, dass sie nicht betriebsbereit zum Start der Diplomarbeit ist	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 GANTT-Diagramm

Nr.		Vorgangsm	Vorgangsname	Dauer	Vorgänger	Meilenstein	Anfang	Ende	Ist-Anfang	Ist-Ende	Geplante Arbeit	Geleistete Arbeit
1			**Phase I: Themenerarbeitung- und eingabe**	**81 Tage**		**Nein**	**Don 17.08.23**	**Don 07.12.23**	**Don 17.08.23**	**NV**	**48 Std.**	
2			Einführung Diplomarbeit	81 Tage		Nein	Don 17.08.23	Don 07.12.23	Don 17.08.23	NV	20 Std.	
3			Ideensammlung	30 Tage		Nein	Don 17.08.23	Mit 27.09.23	NV	NV	8 Std.	
4			Abgabe Ideensammlung	1 Tag		Ja	Mit 27.09.23	Mit 27.09.23	NV	NV	0 Std.	
5			Disposition erstellen	2.5 Tage		Nein	Mit 15.11.23	Son 19.11.23	NV	NV	20 Std.	
6			Abgabe Disposition	1 Tag		Ja	Mon 20.11.23	Mon 20.11.23	NV	NV	0 Std.	
7			Bewilligung Disposition durch FV	1 Tag	6	Ja	Don 07.12.23	Don 07.12.23	NV	NV	8 Std.	
8			**Phase II: Vorarbeitungsarbeiten**	**48 Tage**	**7**	**Nein**	**Fre 08.12.23**	**Die 13.02.24**	**NV**	**NV**	**25 Std.**	
9			Vorebereitungsmassnahmen	43 Tage	7	Nein	Fre 08.12.23	Die 06.02.24	NV	NV	20 Std.	
10			Abgabe Statusbericht	1 Tag	9	Ja	Mit 07.02.24	Mit 07.02.24	NV	NV	1 Std.	
11			Vernissage	1 Tag?	10	Ja	Die 13.02.24	Die 13.02.24	NV	NV	4 Std.	
12			**Phase III: Hauptarbeiten**	**64.75 Tage?**	**11**	**Nein**	**Mit 14.02.24**	**Don 06.06.24**	**NV**	**NV**	**0 Std.**	
13			**Haupteil Diplomarbeit**	**55.75 Tage?**		**Nein**	**Mit 14.02.24**	**Fre 24.05.24**	**NV**	**NV**	**200 Std.**	
14			**Umsetzung**	**25 Tage?**		**Nein**	**Mit 14.02.24**	**Mit 27.03.24**	**NV**	**NV**	**0 Std.**	
15			**Evaluation**	**13 Tage?**		**Nein**	**Mit 14.02.24**	**Mit 06.03.24**	**NV**	**NV**	**0 Std.**	
16	✓		Anforderungskatalog	10 Tage		Nein	Mit 21.02.24	Fre 08.03.24	Mit 21.02.24	Fre 08.03.24	16 Std.	4.5
17			Vorbereitung Benchmarking	1 Tag	16	Nein	Mon 26.02.24	Mon 26.02.24	NV	NV	4 Std.	
18	✓		Analyse PostgreSQL HA Cluster Lösungen	4 Tage	16	Nein	Mon 11.03.24	Don 14.03.24	Mon 11.03.24	Don 14.03.24	16 Std.	7.5
19			Gegenüberstellung	2 Tage	16;17;18	Nein	Mon 04.03.24	Die 05.03.24	NV	NV	8 Std.	
20			Variantenentscheid	1 Tag?	19	Ja	Mit 06.03.24	Mit 06.03.24	NV	NV	4 Std.	
21			**Aufbau und Implementation Testsystem**	**8 Tage?**	**20**	**Nein**	**Fre 08.03.24**	**Mit 20.03.24**	**NV**	**NV**	**27 Std.**	
22			Basisinfrastruktur	2 Tage		Nein	Fre 08.03.24	Mon 11.03.24	NV	NV	4 Std.	
23			Installation und Konfiguration PostgreSQL HA Cluster	5 Tage	22	Nein	Die 12.03.24	Die 19.03.24	NV	NV	20 Std.	
24			Technical Review	1 Tag?	23	Ja	Mit 20.03.24	Mon 16.09.24	NV	NV	3 Std.	
25			**Testing**	**4 Tage?**	**24**	**Nein**	**Fre 22.03.24**	**Mit 27.03.24**	**NV**	**NV**	**14 Std.**	
26			Testing Testsystem	1 Tag	24	Nein	Fre 22.03.24	Fre 22.03.24	NV	NV	8 Std.	
27			Protokollierung	2 Tage	26	Nein	Mon 25.03.24	Die 26.03.24	NV	NV	4 Std.	
28			Review und Auswertung	1 Tag?	27	Ja	Mit 27.03.24	Mit 27.03.24	NV	NV	2 Std.	
29			Troubleshooting und Lösungsfindung	1 Tag?	25	Nein	Die 02.04.24	Die 02.04.24	NV	NV	8 Std.	
30			**Resultate**	**6 Tage?**	**25**	**Nein**	**Die 02.04.24**	**Mit 10.04.24**	**NV**	**NV**	**7 Std.**	
31			Zielüberprüfung	4 Tage		Nein	Die 02.04.24	Mon 08.04.24	NV	NV	2 Std.	
32			Schlussfolgerung	1 Tag?	31	Nein	Die 09.04.24	Die 09.04.24	NV	NV	2 Std.	
33			Weiteres Vorgehen / offene Arbeiten	1 Tag?		Nein	Die 02.04.24	Die 02.04.24	NV	NV	1 Std.	
34			Persönliches Fazit	1 Tag	31;32	Nein	Mit 10.04.24	Mit 10.04.24	NV	NV	2 Std.	
35			1. Expertengespräch	1 Tag?		Ja	Mit 14.02.24	Mit 14.02.24	NV	NV	0 Std.	
36			2. Expertengespräch	1 Tag?		Ja	Fre 15.03.24	Fre 15.03.24	NV	NV	0 Std.	
37			Letztes Expertengespräch	1 Tag?		Ja	Mit 15.05.24	Mit 15.05.24	NV	NV	0 Std.	
38			Dokumentation	55.75 Tage		Nein	Mit 14.02.24	Fre 24.05.24	NV	NV	80 Std.	11.2
39			Puffer	9 Tage		Nein	Die 16.04.24	Die 30.04.24	NV	NV	16 Std.	
40			Abgabe Diplomarbeit	1 Tag?	13;30;37	Ja	Fre 24.05.24	Fre 24.05.24	NV	NV	0 Std.	
41			Vorbereitung Präsentation	5 Tage	40	Nein	Die 28.05.24	Mon 03.06.24	NV	NV	20 Std.	
42			Präsentation und Fachgespräch	1 Tag?	40;41	Nein	Die 04.06.24	Die 04.06.24	NV	NV	1 Std.	
43			Diplomausstellung	1 Tag?	42	Nein	Mit 05.06.24	Mit 05.06.24	NV	NV	0 Std.	
44			Diplomfeier	1 Tag?	42;43	Nein	Don 06.06.24	Don 06.06.24	NV	NV	0 Std.	

Projekt: Diplomarbeit - Postgre
Datum: Sam 16.03.24

Vorgang		Projektsammelvorgang		Manueller Vorgang		Nur Anfang		Kritisch		Geplanter Meilenstein		Puffer			
Unterbrechung		Inaktiver Vorgang		Nur Dauer		Nur Ende		Kritische Unterbrechung		Geplanter Sammelvorgang					
Meilenstein		Inaktiver Meilenstein		Manueller Sammelrollup		Externe Vorgänge		Verspätet		In Arbeit					
Sammelvorgang		Inaktiver Sammelvorgang		Manueller Sammelvorgang		Stichtag		Geplant		Manueller Fortschritt					

1.8 Status-Reports

1.8.1 Initialer Statusbericht

PostgreSQL HA Cluster - Konzeption und Implementation		ICT Projektstatusberich	13.02.2024
Projektbeschreibung	Evaluation und Implementation PostgreSQL HA Cluster	Priorität	
ICT veratw. Person	Michael Graber	PMA	-

Status	Ampel	Tendenz	Begründung
Gesamtprojekt			
Zeitplanung	◆	↓	Projekt ist Umfangreich und hat viele Teilaspekte, die es zu planen und berücksichtigen gilt
Ressourcen	▲	↓	Parallel läuft das Grossprojket 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
- Dokumentenstruktur erstellt	
- Projektplanung erstellt	- Anforderungskatalog erarbeiten
- Versnissage vorbereitet	- Vorbereitung Benchmarking
- Statusbericht erstellt	

# nächste Lieferobjekte (inkl. allfällige Links)		Status	Erledigungsgrad	Soll Datum
LO-001	Anforderungskatalog	in arbeit	60%	23.02.2024
LO-002	Vorbereiten Benchmarking	in arbeit	0%	26.02.2024
LO-003				
LO-004				
LO-005				

# Risiken		Auswirkungsgrad	Massnahmen	Verantw.
R-001	Fehlende Ressourcen	●	Organisation und Selbstmanagement	
R-002	HP-UX Ablöseprojekt	●	Ressourcen reservieren	
R-003	Alte Infrastruktur kann ungeplant sämtliche Ressourcen binden	●	Monitoring vorgängig ausbauen und Massnahmen definieren	
R-004	Schwächen beim Selbstmanagement und in der Selbstorganisation	●	Werkzeuge im Vorfeld definieren und bereitstellen	

Kostenübersicht	Abhängigkeiten zu anderen Projekten	Massnahmen
Verfügbare Finanzen bis Ende Projekt:	Erneuerung HP UX Plattform 60002201	Ressourcen reservieren
$\frac{100\,\text{CHF}}{h} * 200h = 24\,000$ CHF	KSGR Provisioning System (KPS) =>Foreman Umgebung	Massnahmen ergreifen um die manuelle Installation so effizient wie möglich zu gestalten

Bemerkungen / Informationen	Anträge

Eingereicht	Geprüft	Bemerkungen/Auftrag PMO
PL:	PMO:	
Datum:	Datum:	

erledigte Lieferobjekte (inkl. allfällige Links)

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 veratw. Person	Michael Graber	PMA	-

Status	Ampel	Tendenz	Begründung
Gesamtprojekt			
Zeitplanung			In verzug. Grossprojket `Erneuerung HP UX Plattform` nimmt viel Zeit in Anschlag. Hinzu kommt, das die analyse gängiger PostgreSQL HA Lösungen ebenfalls viel Zeit kostet. Dokumentionsaufwand unterschätzt.
Ressourcen			Parallel läuft das Grossprojket `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
- Anforderungskatalog erstellt - Parallel dokumentiert	- Analyse der PostgreSQL HA Clusterlösungen abgeschlossen - Benchmarking abgeschlossen - Variantenentscheid getroffen - Basissystem für Testsystem aufgebaut

# nächste Lieferobjekte (inkl. allfällige Links)		Status	Erledigungsgrad	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 Basisinfrakstruktur Testsystem	offen		

# Risiken		Auswirkungsgrad	Massnahmen	Verantw.
R-001	Fehlende Ressourcen		Organisation und Selbstmanagement	
R-002	HP-UX Ablöseprojekt		Ressourcen reservieren	
R-003	Alte Infrastruktur kann ungeplant sämtliche Ressourcen binden		Monitoring vorgängig ausbauen und Massnahmen definieren	
R-004	Schwächen beim Selbstmanagement und in der Selbstorganisation		Werkzeuge im Vorfeld definieren und bereitstellen	
R-005	Scope verlust während des Projekts		Ziele klar definieren	
R-006	Scope Creep		Ziele SMART definieren	
R-007	SIEM / Log Plattform nicht betriebsbereit			
R-008	Foreman nicht betriebsbereit		Massnahmen ergreifen um die manuelle Installation so effizient wie möglich zu gestalten	

Kostenübersicht	Abhängigkeiten zu anderen Projekten	Massnahmen
Verfügbare Finanzen bis Ende Projekt: $100\,\mathrm{CHF\,h^{-1}} * 200 = 24\,000\,\mathrm{CHF}$	Erneuerung HP UX Plattform 60002201 KSGR Provisioning System (KPS) => Foreman Umgebung	Ressourcen reservieren Massnahmen ergreifen um die manuelle Installation so effizient wie möglich zu gestalten

Bemerkungen / Informationen	Anträge

Eingereicht	Geprüft	Bemerkungen/Auftrag PMO
PL:	PMO:	
Datum:	Datum:	

# erledigte Lieferobjekte (inkl. allfällige Links)	
LO-001	Anforderungskatalog

Tabelle 1.13: 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
Definiertr, 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[44]. Dabei wird zwischen den folgenden beiden Replikationen unterschieden:

Synchrone Replikation
Wenn bei einer Synchronen Replikation eine Transaktion abgesetzt wird, wird der Commit auf der primären Seite erst gesetzt, wenn die Änderung auf der sekundären Seite oder den sekundären Seiten ebenfalls eingetragen und Committed ist. Bis zu diesem Moment ist die Transaktion nicht als Committed.
Dies wird dann zum Problem, wenn keine Verbindung mehr zu mindesten einer sekundären Seite vorhanden ist. Zudem wird die Synchrone Replikation bei hohen Latenzen zum Bottleneck der Datenbank.

Asynchrone Replikation

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

2.1.1.5 Quorum

Ein Quorum-System soll die Integrität und Konsistenz in einem Datenbank-Cluster sicherstellen. Dabei gilt zu beachten, das 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-brainSituation 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 [42, 54, 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, das nur zwei der drei folgenden drei Merkmale von verteilten Systeme gewährleistet werden können[28].

Konsistenz - Consistency
Die Datenbank ist Konsistent, alle Clients seher gleichzeitig die gleichen Daten unabhängig 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 ist.

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	Systemvielfallt		Es muss mindestens eine Monolitisches 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 geordeten 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 Restoren 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	Monolitisch	MUSS
14	Replikation	Failover / Switchover Garantie	Die Replikation muss sicherstellen, das es bei einem Failover/Switchover zu keinem Fehler kommt	Monolitisch	MUSS
15	Replikation	Throughput	Beschreibt, wie viele Transaktionen pro Zeiteinheit vom Primary an die Replikas gesendet und Commited werden. Dieser Wert ist bei Synchroner 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	Robhustheit	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 damit Nodes hinzugefügt oder entfernt werden können	Beides	MUSS
22	Management-API	Authentifizierung & Autorisierung	Es müssen gängige Standards für Authentifizierung und Autorisierung mitgebracht werden	Beides	MUSS
23	Management-API	Aufwand	Der Aufwand, der benötigt wird um die DB zu verwalten, Nodes hinzuzufügen oder zu entfernen usw. muss gegeneinander verglichen werden.	Beides	MUSS
24	Backup	Backup mit PostgreSQL Standards	Backups müssen mittels PostgreSQL Standards angezogen werden	Beides	MUSS
25	Backup	Restore mit PostgreSQL Standanrds	Backups müssen mittels PostgreSQL Standards restored werden können	Beides	MUSS
26	Housekeeping - Log Rotation		Das Clustersystem muss die möglichkeit zur Log Rotation bieten	Beides	MUSS
27	Self Heahling		Das Clustersystem muss im Fehlerfall Nodes selber wiederherstellen können	Beides	KANN
28	Monitoring - Node Failure		Läuft ein Node auf einen Fehler, muss das Clustersystem dies erkennen und Melden resp. eine Schnittstelle liefern die abgefragt werden kann	Beides	MUSS
29	Maintenance Quality		Da die meisten PostgreSQL HA Lösungen Open-Source sind, muss sichergestellt werden, dass die gewählte Lösung auch aktiv gepflegt wird. Als Basis dienen hier Informationen wie z.B. GitHub Insights.	Beides	MUSS
30	Performance	tps - Read-Only	Die Transaktionsrate (transactions per second / tps) für DQL Transaktionen	Beides	MUSS
31	Performance	tps - Read-Writes	Die Transaktionsrate (transactions per second / tps) für DML Transaktionen	Beides	MUSS
32	Performance	Ø Latenz - Read-Only	Die Latenzzeit bei DQL Transaktionen	Beides	MUSS
33	Performance	Ø Latenz - Read-Write	Die Latenzzeit bei DML Transaktionen	Beides	MUSS

Tabelle 2.2: Anforderungskatalog

2.1.2.2 Stakeholder

Rolle	Funktion	Departement	Bereich	Abteilung
Zabbix Stakeholder	Abteilungsleiter	D10 ICT	Infrastrukturmanagement	ICT Netzwerk, Security und Comm.
Staekholder 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 Systemvielfallt	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 Heahling	14 Monitoring - Node Failure	15 Maintenance Quality
13	15	1	1	Systemvielfallt															
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 Heahling	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	1 Automatismus	2 Connection-Stabilität
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)	Connection - Stabilität (2)
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 Skript / API	2 Datensicherheit	3 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	Synchrone Replikation 1	Failover / Switchover Garant 2
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	Datenkonsistenz- und Integr 1
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 1
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 1	Authentifizierung & Autorisie 2
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	Backup mit PostgreSQL Stan 1
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 tps - Read-Only	2 tps - Read-Writes	3 Ø Latenz - Read-Only
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, 69, 1] sind dies folgende Anforderungen:

Maintenance
Vor und nach dem Initialisieren des Benchmarks muss AUTOVACUUM gestartet 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 [65] 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 [53].

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[55]. 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[51, 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 Monolothisch.

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:

Abbildung 2.20: Patroni - Community Standards

Die Contributors commiten, löschen und erweitern Patroni Regelmässig:

Abbildung 2.21: Patroni - Contributors Commits

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[66].

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

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

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 verhältnissmässig 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-Standars, Stackgres selbst schneidet hier nur Mittelmässig ab:

Abbildung 2.37: Stackgres - Community Standards

Abbildung 2.38: Citus - Community Standards

Die Stackgres Constributors pflegen aktiv Additions ein, löschen Regelmässig und Commiten ebenfalls auf die main-Branch. Citus, dessen Repository länger Commited 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 evaluirt, 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 vernüftigen Zeit.
2	pgpool-II	Vorausgeschieden	pgpool-II hat kein GitHub-Repository und bietet daher keine vergleichswerte mittels Github Insights.
3	pg_auto_failover	Vorausgeschieden	pg_auto_failover würde zwar Citus-Support bieten, allerdings gibt es keine gut dokumentierte Implementation für Kubernetes. Erfüllt daher das Kriterium für die Synergien nicht
4	CloudNativePG	Vorausgeschieden	CloudNativePG ist keine vollständige Cloud Native Lösung. Mittels Citus könnte sogar eine Distributed SQL Lösung implementiert werden. Die Grundarchitektur bleibt aber Monolithisch mit einem Primary und Replikas. Und da kein Benefit in Form von Synergien vorhanden sind, fällt CloudNativePG raus.
8	Citus row-based-sharding	Vorausgeschieden	Citus row-based-sharding wäre Hocheffizient wenn es um Ressourcenverteilung geht und zudem echtes Sharding. Allerdings setzt es anpassungen an den Tabellen der Applikationen voraus. Das KSGR ist allerdings kein Softwarehaus und kann keine Forks durchführen, auch weil viele Applikationen zertifiziert sein müssen. Scheitert daher an der Machbarkeit

Tabelle 2.4: 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. Da Patroni unter der Haube ist, kann die API und sonstige Skripte auch auf einem Monolithischen System eingesetzt werden.
7	Yugabyte-DB	Evaluation	Ist eine reine Distributed SQL Lösung und ist Vollständig Cloud Native.

Tabelle 2.5: 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
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[56] . . . 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 64
2.36 Citus - Code Frequency 65
2.37 Stackgres - Community Standards 65
2.38 Citus - Community Standards 66
2.39 Stackgres - Contributors Commits 66
2.40 Stackgres - Contributors Deletations 67
2.41 Stackgres - Contributors Additions 67
2.42 Citus - Contributors Commits 67
2.43 Citus - Contributors Deletations 68
2.44 Citus - Contributors Additions 68
2.45 Stackgres - Commit Activity 68
2.46 Citus - Commit Activity 69
2.47 Stackgres - Network Graph 69
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 70
2.5 Vorauswahl - Evaluation 71
2.6 Evaluationssyssteme 71
2.7 Evaluationssysstem - 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 - Risxikomatrizen 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 - worder-node-failure*. https://docs.citusdata.com/en/v12.1/admin_guide/cluster_management.html#worker-node-failu

[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-s 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/InstitutionenGesundeitswesens/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 Insausti. *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-d 2020.

[42] Unmesh Joshi. *Quorum*. https://martinfowler.com/articles/patterns-of-distributed-systems/quorum.html. 2020.

[43] Martin Keen und IBM Deutschland GmbH. *IBM Db2*. https://www.ibm.com/de-de/products/db2.

[44] Pasha Kostohrys. *Database replication — an overview*. https://medium.com/@pkostohrys/database-replication-an-overview-f7ade110477. 2020.

[45] Anatoli Kreyman. *Was ist eigentlich Splunk?* https://www.kreyman.de/index.php/splunk/76-was-ist-eigentlich-splunk-big-data-platform-monitoring-security.

[46] Pankaj Kushwaha und Unit 3D North Point House. *POSTGRESQL DATABASE MAINTENANCE. Routine backup of daily database... | by Pankaj kushwaha | Medium*. https://pankajconnect.medium.com/postgresql-database-maintenance-66cd638d25ab.

[47] Red Hat Limited. *Was ist Ansible?* https://www.redhat.com/de/technologies/management/ansible/what-is-ansible.

[48] 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.

[49] Switzerland Linuxfabrik GmbH Zurich. *Keepalived — Open Source Admin-Handbuch der Linuxfabrik*. https://docs.linuxfabrik.ch/software/keepalived.html. 2023.

[50] Nico Litzel, Stefan Luber und Vogel IT-Medien GmbH. *Was ist Elasticsearch?* https://www.bigdata-insider.de/was-ist-elasticsearch-a-939625/. 2020.

[51] SRA OSS LLC. *pgpool Wiki*. https://www.pgpool.net/mediawiki/index.php/Main_Page. 2023.

[52] Hewlett Packard Enterprise Development LP. *Was ist SAN-Speicher? | Glossar*. https://www.hpe.com/ch/de/what-is/san-storage.html.

[53] Julian Markwort. „Benchmarking four Different Replication Solutions“. In: ().

[54] Diego Ongaro. „Consensus: Bridging Theory and Practice“. In: (2014).

[55] Bruno Queirós und LinkedIn Ireland Unlimited Company. *Postgresql replication with automatic failover*. https://www.linkedin.com/pulse/postgresql-replication-automatic-failover-brun C3%B3s. 2020.

[56] 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.

[57] Ed Reckers und SnapLogic Inc. *Was ist die Snowflake-Datenplattform?* https://www.snaplogic.com/de/blog/snowflake-data-platform. 2023.

[58] IONOS SE. *Apache Cassandra: Verteilte Verwaltung großer Datenbanken*. https://www.ionos.de/digitalguide/hosting/hosting-technik/apache-cassandra-vorgestellt/. 2021.

[59] IONOS SE. *Datenbankmanagementsystem (DBMS) erklärt*. https://www.ionos.de/digitalguide/hosting/hosting-technik/datenbankmanagementsystem-dbms-erklaert/. 2020.

[60] IONOS SE. *MongoDB – die flexible und skalierbare NoSQL-Datenbank*. https://www.ionos.de/digitalguide/websites/web-entwicklung/mongodb-vorstellung-und-vergleich-mit-mysc 2019.

[61] IONOS SE. *SQLite: Die bekannte Programmbibliothek im Detail vorgestellt*. https://www.ionos.de/digitalguide/websites/web-entwicklung/sqlite/. 2023.

[62] IONOS SE. *Terraform*. https://www.ionos.de/digitalguide/server/tools/was-ist-terraform/. 2020.

[63] IONOS SE. *Was ist Redis? Die Datenbank vorgestellt*. https://www.ionos.de/digitalguide/hosting/hosting-technik/was-ist-redis/. 2020.

[64] IONOS SE. *Was ist SIEM (Security Information and Event Management)?* https://www.ionos.de/digitalguide/server/sicherheit/was-ist-siem/. 2020.

[65] Naveed Shaikh. *It's All About Replication Lag in PostgreSQL*. https://www.percona.com/blog/replication-lag-in-postgresql/. 2023.

[66] Sami Ahmed Siddiqui. *Distributed SQL 101*. https://www.yugabyte.com/distributed-sql/.

[67] Inc. Snowflake. *Datenbanken, Tabellen und Ansichten – Überblick | Snowflake Documentation*. https://docs.snowflake.com/de/guides-overview-db.

[68] Thomas-Krenn.AG. *Git Grundlagen – Thomas-Krenn-Wiki*. https://www.thomas-krenn.com/de/wiki/Git_Grundlagen.

[69] Mahmut Can Uçanefe. *Pgbench Load Test*. https://medium.com/@c.ucanefe/pgbench-load-test-166b 2023.

[70] Rainer Züst. „Einstieg ins Systems Engineering“. In: (2002).

Glossar

Ansible Ansible ist ein Open-Soure Automatisierungstool zur Provisionierung, Konfiguration, Deployment und Orchestrierung. Ansible verbindet sich auf auf die Zielgeräte und führt dort die Hinterlegten Module aus. Oft werden die verschieden Aufgaben in einem Skript, in einem sogenannten Playbook geschrieben werden[47].. 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[58] wurde.. 7, 59

CI/CD Continuous Integration/Continuous Delivery bedeutet, das Anpassungen kontinuirlich in die Entwicklungsumgebungen integriert und auf die Zielplattformen verteilt werden[48].. 4

DBMS Ein Database Management System regelt und organisiert die Datenbasis einer Datenbank[59].. 4

Debian Debian gehört nebst Slackware Linux zu den ältesten Linux Distribution die noch immer gepflegt und eingesetzt werden. Sie wurde im August 1993 gestartet und brachte im Laufe der Zeit einige der beliebstesten 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[50].. 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 organsiert und arbeiten daher mit Working Copies von Repositories[27, 68].. 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[43] 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 realsieren. 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[49, 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 inspiert 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 eder Zeit immer mehr voneinander undf sind nicht mehr in jdem Fall kompatibel oder beliebig austauschbar. Auf den Linux Distributionen tratt 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 zurm ersten Mal 2007 veröffentlicht wurde[60].. 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 Microsyszem durch Oracle gründen zu dürfen. Seit Oracle Sun Microsystem 2010 gekauft hat, wurden immer mehr Funktionalitäten von der Community Edition zu der Enterprise Edition verschoben worden. Aus diesem Grund hat heute der MySQL Fork MariaDB MySQL mehrheitlich aus allen Linux Distributionen als Standard Datenbank verdrängt.. 5, 7, 8

NoSQL NoSQL steht für Not only SQL. Das heisst, Relationale Datenbanken haben komponenten wie Dokumentendatenbanken, Graphendatenbanken, Key-Value-Datenbanken und Spaltenorientiert Datenbanken. Viele der grossen Datenbanklösungen wie Oracle Database oder Microsoft SQL Server sind NoSQL Datenbanken resp. bieten diese option an.. 7, 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 Binärkompatibel. Sie wird primär für den Betrieb von Oracle Datenbanken verwendet und komnt auf den Oracle Eigenenen 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 Vergangheit ermöglichen) konnte Oracle sich stark von der Konkurenz absetzen. Heute gilt die Datenbank als erste Wahl, wenn es um Hochverfügbare Systeme, hohe Perforamce 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 beliebstesten 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 Netzwerktrennung 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 Unterunterabschnitt 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 Tabellenorgansierte 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 Enterpise 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[63]. 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, 52, 31]. 4, 5, 16, 20

SIEM Ein sammelt Daten aus verschieden Netzwerkkomponenten oder Geräten von Agents oder Logs. Diese Daten werden permanent analysiert und mit einem definierten Regelwerk gegengeprüft. Ziel ist es, verdächtige Events zu erkennen und einem Angriff zuvorzukommen oder ihn möglichst früh zu unterbinden[64].. 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[67, 57]. 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, 45]. . 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[61].. 7

Switchover In einem Maintenance-Fall in einem HA-System meist ein Primary Node auf den Secondary geplant geswitched.. 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[62, 37].. 16

Transaktion Eine Transaktion ist beinhaltet Schreib-, Lese-, Mutatations- oder Löschoperatione 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 enstanden im Laufe der Zeit viele offene und Proprioritäre 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. Zabbrix speichert die Metriken und nicht die Auswertungen, das heisst, solange die Daten vorhanden sind können Grafiken zu jedem Zeitpunkt generiert worden. Zabbix ist grundsätzlich Open-Source, man kann allerdings Supportverträge Abschliessen.. 8, 16

Selbstständigkeitserklärung

Ich versichere, dass die vorliegende Arbeit von den Autoren selbständig 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.	Generator mit python pandas gebaut für alle möglichen Tabellen. Inkl. Aggregation und Pivot-Mechaniken
28.02.2024	09:00	11:00	2.0	Dokumentation	-	Dokumentation erweitern		Viele LaTEX Tabellen.	Generator mit python pandas gebaut für alle möglichen Tabellen. Inkl. Aggregation und Pivot-Mechaniken
01.03.2024	07:00	09:00	2.0	Dokumentation	-	Dokumentation 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.	pgpool II fällt somit direkt aus der betrachtung raus, da kein vergleich möglich ist.
11.03.2024	12:00	13:30	1.5	Dokumentation	-	Dokumentation erweitern			
11.03.2024	16:45	17:30	0.5	Dokumentation	-	Dokumentation erweitern	Stakeholder erfassen		
13.03.2024	17:45	19:45	2.0	Evaluation	Analyse PostgreSQL HA Cluster Lösungen	Stackgres und Citus analysieren	Citus row-based-sharding	Citus Dokumentation stark Textlastig. Wenig Abbildungen, vieles muss selber gezeichnet werden.	
14.03.2024	19:45	20:45	1.0	Evaluation	Analyse PostgreSQL HA Cluster Lösungen		Citus row-based-sharding		
14.03.2024	20:45	21:30	0.8	Dokumentation	-	Projektcontrolling Arbeiten	Citus row-based-sharding Dokumentieren		
16.03.2024	17:45	18:30	0.8	Dokumentation	-	Projektcontrolling Arbeiten			
17.03.2024	14:45	16:30	1.8	Dokumentation	-	Dokumentation erweitern	Zweiter Statusbericht verfassen		
17.03.2024	19:30	20:00	0.5	Dokumentation	-	Dokumentation erweitern	ACID Exkurs erfassen		
17.03.2024	20:15	21:00	0.8	Dokumentation	-	Dokumentation erweitern	Listings sauber machen. Neue Listing-Sprache für yaml-Files erstellt, da noch einige folgen werden.		
18.03.2024	14:00	16:00	2.0	Dokumentation	-	Dokumentation erweitern	Statusbericht 2 fertig Schrieben und Mail an Norman Süsstrunk senden		
18.03.2024	20:20	21:50	1.5	Evaluation	Vorbereitung Benchmarking	pgbench analysieren	Percona ist Dein Freund		
19.03.2024	08:00	10:00	2.0	Evaluation	Analyse PostgreSQL HA Cluster Lösungen	yugabytedb			
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 Süsstrunk	-	Michael Graber Curdin Roffler	- Darf eine Vorauswahl stattfinden, um den Aufwand zur reduzieren?	- Eine Vorauswahl ist Sinnvoll und in diesem Rahmen fast zwingend Notwendig, da sonst viel zuviel Zeit investiert werden müsste	- Vorstellung Norman Süsstrunk, Curdin Roffler und Michael Graber - Kontaktdaten shared - Bei Fragen jederzeit an Norman wenden - Norman braucht aber mindestens 1. Woche vorlaufzeit - Norman wird sich spätestens zur Halbzeit melden. - Norman wird sic	- Es wurden zwar für alle Studenten von Norman Süsstrunk Zoom-Räume bereitgestellt, aus effizienzgründen nahmen Curdin Roffler und ich beide am selben Meeting teil
2		Norman Süsstrunk	-	Michael Graber	- 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 Freiutag 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

```
import json
import pybtex
import requests
import os
from pybtex.database import BibliographyData, Entry, Person
from dateutil.parser import parse
import math

def load_configuration():
    zotero_bibtex_config = dict()
    zotero_conf_filename = 'zotero_bibtex_configuration.json'
    zotero_conf_dir = os.path.join(os.path.dirname(os.getcwd()), 'source', '
    configuration')
    # zotero_conf_dir = os.path.join(os.getcwd(), 'src', 'content')
    json_path = os.path.join(zotero_conf_dir, zotero_conf_filename)

    with open(json_path) as json_string:
        zotero_bibtex_config = json.load(json_string)

    return zotero_bibtex_config
def downlaod_zotero_datas(URL, API_KEY):
    zotero_result = list
    response = requests.get(URL, headers={'Zotero-API-Key': API_KEY})
    response = response.json()
    zotero_raw = json.dumps(response, ensure_ascii=False)  # json.loads(response)
    zotero_result = json.loads(zotero_raw)
    return zotero_result

def get_data(zotero_bibtex_config):
    # result_limit = 100
    # access_type = 'groups'
    # zotero_access_id = '5245833'
    # collection_id = 'USSFDCEH'
    result_limit = int(zotero_bibtex_config.get('result_limit'))
    access_type = zotero_bibtex_config.get('access_type')
    zotero_access_id = zotero_bibtex_config.get('zotero_access_id')
    collection_id = zotero_bibtex_config.get('collection_id')
    API_KEY = zotero_bibtex_config.get('api_key')
    zotero_data = list()
    URL = 'https://api.zotero.org/' + str(access_type) + '/' + str(
    zotero_access_id) + '/collections/' + str(
        collection_id) + '/items?limit=1?format=json?sort=dateAdded?direction=asc'

    # API_KEY = '6Xgb3XhGjQXwA8NuZgu3bw3s'
    response = requests.get(URL, headers={'Zotero-API-Key': API_KEY})

```

```
    header_dict = response.headers
    total_elemets = int(header_dict.get('Total-Results'), 0)

    if total_elemets < result_limit:
        URL_ALL_ITEMS = 'https://api.zotero.org/' + str(access_type) + '/' + str(
            zotero_access_id) + '/collections/' + str(collection_id) + '/items?
    limit=' + str(
            result_limit) + '?format=json?sort=dateAdded?direction=asc'
        zotero_result = downlaod_zotero_datas(URL_ALL_ITEMS, API_KEY)

        zotero_data.extend(zotero_result)
    else:
        runs = int(math.ceil(total_elemets / result_limit))
        index = 0
        start_index = 0
        while index < runs:
            URL_Separated = 'https://api.zotero.org/' + str(access_type) + '/' +
    str(
                zotero_access_id) + '/collections/' + str(collection_id) + '/items
    ?limit=' + str(
                result_limit) + '?format=json?sort=dateAdded?direction=asc' + '&
    start=' + str(start_index)
            zotero_result = downlaod_zotero_datas(URL_Separated, API_KEY)

            zotero_data.extend(zotero_result)

            start_index += result_limit
            index += 1

    return zotero_data

def convert_to_datetime(input_str, parserinfo=None):
    return parse(input_str, parserinfo=parserinfo)
def get_dates(date, bibtex_item_type, bibtex_month_attributes):
    dated_date = convert_to_datetime(date)
    return_value = dict()
    if bibtex_item_type in bibtex_month_attributes:
        year = dated_date.year
        month = dated_date.month
        return_value = {'year': year, 'month':month}
    else:
        year = dated_date.year
        return_value = {'year': year}

    return return_value

def split_creators(creators):
    if creators != []:
```

```
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):
```

```
                bibtex_item_key = 'masterthesis'
            # Second PHD Thesis
            elif any(item in bibtex_item_titel for item in phd_list):
                bibtex_item_key = 'phdthesis'
            else:
                bibtex_item_key = 'masterthesis'
        else:
            if zotero_bibtex_keys.get(zotero_item_type).get('key'):
                bibtex_item_key = zotero_bibtex_keys.get(zotero_item_type).get('key')
            else:
                bibtex_item_key = 'misc'

        # get all Keys for the zotero item type
        entryset = '\n'
        entry = ''

        zotero_item_attributes = zotero_bibtex_keys.get(zotero_item_type).get('attributes').keys()
        item_attributes = sorted(zotero_item_attributes, reverse=True)

        for index, item_attribute in enumerate(item_attributes):
            bibtex_item_attribute = zotero_bibtex_keys.get(zotero_item_type).get('attributes').get(item_attribute)
            zotero_item_value = biblio_item.get(item_attribute)
            zotero_item_value_extra = ''
            bibtex_item_attribute_extra = ''

            # Special Cases
            if bibtex_item_attribute == 'SPECIALCHECK' and zotero_item_value not in ['', None]:
                bibtex_special_attribute = zotero_bibtex_attributes_special.get(item_attribute)

                match bibtex_special_attribute:
                    case 'get_dates':
                        zotero_item_value = get_dates(zotero_item_value, bibtex_item_key, bibtex_month_attributes)
                        if zotero_item_value.get('month'):
                            zotero_item_value_extra = zotero_item_value.get('month')
                            bibtex_item_attribute_extra = 'month'

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

```


zotero_bibtex_config = load_configuration()
zotero_data = get_data(zotero_bibtex_config)
write_bibliography(zotero_data, zotero_bibtex_config)
```

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

V riskmatrix.py

```
import matplotlib.pyplot as plt
import numpy as np
import pip as pd
import os
import csv
import pandas as pd

def riskmatrix(risk, conf, matrix):
    # get the risk datas
    risk_conf = conf.get(risk)
    startpath = risk_conf.get('startpath')
    destination = risk_conf.get('destinatination')
    imagename = risk_conf.get('imagename')
    datafilename = risk_conf.get('datafilename')
    itemname = risk_conf.get('itemname')
    x_axis_title = risk_conf.get('x-axis-title')
    y_axis_title = risk_conf.get('y-axis-title')
    title = risk_conf.get('title')
    bubble_standard_size = int(risk_conf.get('bubble-standard-size'))

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

    print(directory)

    # get the Datas as dirct
    data_path = os.path.join(directory, datafilename)
    image_path = os.path.join(directory, imagename)

    # load datas from csv into dict
    with open(data_path) as f:
        csv_list = [[val.strip() for val in r.split(",")] for r in f.readlines()]

    (_, *header), *data = csv_list
    datas = {}
```

```
    for row in data:
        key, *values = row
        datas[key] = {key: value for key, value in zip(header, values)}

    # fig_dir = os.path.join(os.path.dirname(os.getcwd()), 'src', 'source')
    fig = plt.figure()
    plt.subplots_adjust(wspace=0, hspace=0)
    plt.xticks([])
    plt.yticks([])
    plt.xlim(0, 5)
    plt.ylim(0, 5)
    plt.xlabel(x_axis_title)
    plt.ylabel(y_axis_title)
    plt.title(title)

    #This example is for a 5 * 5 matrix
    nrows=5
    ncols=5
    axes = [fig.add_subplot(nrows, ncols, r * ncols + c + 1) for r in range(0,
   nrows) for c in range(0, ncols) ]

    # remove the x and y ticks
    for ax in axes:
        ax.set_xticks([])
        ax.set_yticks([])
        ax.set_xlim(0,5)
        ax.set_ylim(0,5)

    #Add background colors
    #This has been done manually for more fine-grained control
    #Run the loop below to identify the indice of the axes

    #Identify the index of the axes
    green = [10, 15, 16, 20 , 21] #Green boxes
    yellow = [0, 5, 6, 11, 17, 22, 23] #yellow boxes
    orange = [1 , 2, 7, 12, 13, 18, 19, 24] # orange boxes
    red = [3, 4, 8, 9, 14] #red boxes

    for _ in green:
        axes[_].set_facecolor('green')

    for _ in yellow:
        axes[_].set_facecolor('yellow')

    for _ in orange:
        axes[_].set_facecolor('orange')

    for _ in red:
```

```
        axes[_].set_facecolor('red')


    #Add labels to the Green boxes
    # axes[10].text(0.1,0.8, '4')
    # axes[15].text(0.1,0.8, '2')
    # axes[20].text(0.1,0.8, '1')
    # axes[16].text(0.1,0.8, '5')
    # axes[21].text(0.1,0.8, '3')

    #Add labels to the Yellow boxes
    # axes[0].text(0.1,0.8, '11')
    # axes[5].text(0.1,0.8, '7')
    # axes[6].text(0.1,0.8, '12')
    # axes[11].text(0.1,0.8, '8')
    # axes[17].text(0.1,0.8, '9')
    # axes[22].text(0.1,0.8, '6')
    # axes[23].text(0.1,0.8, '10')

    #Add lables to the Orange boxes
    # axes[1].text(0.1,0.8, '16')
    # axes[2].text(0.1,0.8, '20')
    # axes[7].text(0.1,0.8, '17')
    # axes[12].text(0.1,0.8, '13')
    # axes[13].text(0.1,0.8, '18')
    # axes[18].text(0.1,0.8, '14')
    # axes[19].text(0.1,0.8, '19')
    # axes[24].text(0.1,0.8, '15')

    #Add lables to the Red Boxes
    # axes[3].text(0.1,0.8, '23')
    # axes[8].text(0.1,0.8, '21')
    # axes[4].text(0.1,0.8, '25')
    # axes[9].text(0.1,0.8, '24')
    # axes[14].text(0.1,0.8, '22')

    # run throuh datas and generate axis datas
    dict_bubble_axis = dict()
    bubble_axis = list()
    for datasets in datas:
        # get the datas
        riskid = datas.get(datasets).get('risk-id')
        x_axis = int(datas.get(datasets).get('x-axis'))
        y_axis = int(datas.get(datasets).get('y-axis'))
        axis_point = matrix.get((x_axis, y_axis))
        x_axis_text = float(datas.get(datasets).get('x-axis-text'))
        y_axis_text = float(datas.get(datasets).get('y-axis-text'))
        x_axis_bubble = float(datas.get(datasets).get('x-axis-bubble'))
```

```
        y_axis_bubble = float(datas.get(datasets).get('y-axis-bubble'))
        bubble_axis.append(axis_point)

        # merge riks if two or more risks share the same axispoint
        if dict_bubble_axis.get(axis_point):
            risktag = dict_bubble_axis.get(axis_point).get('risk')
            risktag = risktag + '/' + riskid
            x_axis_text = x_axis_text + 0.25
            y_axis_text = y_axis_text - 0.5
            bubble_size = bubble_standard_size * 2
        else:
            risktag = itemname + riskid
            bubble_size = bubble_standard_size
        dict_axis_value = dict()

        dict_axis_value['risk'] = risktag
        dict_axis_value['x-axis-text'] = x_axis_text
        dict_axis_value['y-axis-text'] = y_axis_text
        dict_axis_value['x-axis-bubble'] = x_axis_bubble
        dict_axis_value['y-axis-bubble'] = y_axis_bubble
        dict_axis_value['size'] = bubble_size
        dict_bubble_axis[axis_point] = dict_axis_value

    # cleanup the list, remove duplicated entries
    bubble_axis = set(bubble_axis)

    # plot the bubbles and texts in the bubbles
    for axispoint in bubble_axis:
        axes[axispoint].scatter(dict_bubble_axis[axispoint]['x-axis-bubble'],
    dict_bubble_axis[axispoint]['y-axis-bubble'], dict_bubble_axis[axispoint]['
    size'], alpha=1)
        axes[axispoint].text(dict_bubble_axis[axispoint]['x-axis-text'],
    dict_bubble_axis[axispoint]['y-axis-text'], s=dict_bubble_axis[axispoint]['
    risk'], va='bottom', ha='center')

    # save the plot as image
    plt.savefig(image_path)

"""
Config File:
    1.  Name
    2.  Startpoint Directory
    3.  Destination Dir
    4.  Alternate Path
    5.  Data File Name
Data File:
    1.  Spalte:  Nummer
    2.  x-achse
```

```
    3.  x-achse
"""

"""
    Matrix
    This Matrix translate the x/y axis from a given risk matrix csv to the
    axispoint.

    The key of each axispoint is an integer tupel (x, y)
    So, you can access the axis point this way:
    <axispoint> = matrix.get((<x_axis>, <y_axis>))
"""
matrix = {
     # first column
     (1, 1):20,
     (1, 2):15,
     (1, 3):10,
     (1, 4):5,
     (1, 5):0,
     # second column
     (2, 1):21,
     (2, 2):16,
     (2, 3):11,
     (2, 4):6,
     (2, 5):1,
     # third column
     (3, 1): 22,
     (3, 2): 17,
     (3, 3): 12,
     (3, 4): 7,
     (3, 5): 2,
     # fourth column
     (4, 1): 23,
     (4, 2): 18,
     (4, 3): 13,
     (4, 4): 8,
     (4, 5): 3,
     # fifth column
     (5, 1): 24,
     (5, 2): 19,
     (5, 3): 14,
     (5, 4): 9,
     (5, 5): 4
}

# load the configuration file
riskmatrix_conf_filename = 'conf.csv'
riskmatrix_conf_dir = 'source/configuration/'
```

```
conf_riskmatrix_path = os.path.join(os.path.dirname(os.getcwd()),
    riskmatrix_conf_dir)
conf_csv_path = os.path.join(conf_riskmatrix_path, riskmatrix_conf_filename)
with open(conf_csv_path) as f:
    csv_list = [[val.strip() for val in r.split(",")] for r in f.readlines()]

(_, *header), *data = csv_list
conf = {}
for row in data:
    key, *values = row
    conf[key] = {key: value for key, value in zip(header, values)}

for risks in conf:
    riskmatrix(risks, conf, matrix)
# 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 - Risxikomatrizen

VI cost_benefit_diagram.py

```
import matplotlib.pyplot as plt
import os
import csv
import pandas as pd
import yaml

# Get the Configuration
def load_configuration():
    cost_benefit_config = dict()
    cbd_conf_filename = 'scatter_plotter_conf.yaml'
    cbd_conf_dir = os.path.join(os.path.dirname(os.getcwd()), 'source', '
    configuration')
    yaml_path = os.path.join(cbd_conf_dir, cbd_conf_filename)

    with open(yaml_path, "r") as file:
        cost_benefit_config = yaml.load(file, Loader=yaml.FullLoader)

    return cost_benefit_config
# Get the Datas
def get_data(cost_benefit_config):
    # Config Variables
    startpath = cost_benefit_config.get('startpath')
    destination = cost_benefit_config.get('desitination_path')
    datafilename = cost_benefit_config.get('datafile')

    if startpath == 'homedir':
```

```
        directory = os.path.join(os.getcwd(), destination)
    else:   # parentdir
        directory = os.path.join(os.path.dirname(os.getcwd()), destination)

    # get the Datas as dirct
    data_path = os.path.join(directory, datafilename)

    # load datas from csv into dict
    with open(data_path) as f:
        csv_list = [[val.strip() for val in r.split(",")] for r in f.readlines()]

    (_, *header), *data = csv_list
    datas = {}
    for row in data:
        key, *values = row
        datas[key] = {key: value for key, value in zip(header, values)}

    cost_benefit_data = {}
    for key, value in datas.items():
        variant_name = value['variant_name']
        x_axis = int(value['x-axis'])
        y_axis = int(value['y-axis'])
        cost_benefit_data[variant_name] = (x_axis, y_axis)

    return cost_benefit_data

# Plot the Datas
def cost_benefit_diagram (cost_benefit_config, cost_benefit_data):
    # Config Variables
    startpath = cost_benefit_config.get('startpath')
    destination = cost_benefit_config.get('desitination_path')
    imagename = cost_benefit_config.get('imagename')

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

    # get the Datas as dirct
    data_path = os.path.join(directory, imagename)

    # Extract the Datas
    labels, values = zip(*cost_benefit_data.items())
    x, y = zip(*values)

    # Create Scatter-Diagram
    plt.scatter(x, y, color=cost_benefit_config.get('scatter-point-color'))

```

```
    # X-Lines
    plt.axhline(y=cost_benefit_config.get('y-axis-line-pos'), color=
    cost_benefit_config.get('y-axis-line-color'), linestyle=cost_benefit_config.
    get('y-axis-line-type'), label=cost_benefit_config.get('y-axis-line-label'))

    # Y-Lines
    plt.axvline(x=cost_benefit_config.get('x-axis-line-pos'), color=
    cost_benefit_config.get('x-axis-line-color'), linestyle=cost_benefit_config.
    get('x-axis-line-type'), label=cost_benefit_config.get('x-axis-line-label'))

    # Add Labels
    plt.xlabel(cost_benefit_config.get('x-axis-title'))
    plt.ylabel(cost_benefit_config.get('y-axis-title'))
    plt.title(cost_benefit_config.get('title'))

    # Labling Data Points
    for label, x_point, y_point in zip(labels, x, y):
        plt.text(x_point, y_point, label)

    # Show Legends
    plt.legend()

    # Show Grid
    plt.grid(True)

    # Save Diagram as PNG
    plt.savefig(data_path)

cost_benefit_config = load_configuration()
cost_benefit_data = get_data(cost_benefit_config)
cost_benefit_diagram(cost_benefit_config, cost_benefit_data)
```

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

VII csscatter_plotter_conf.yaml

```yaml
startpath: "parentdir"
desitination_path: "source/cost_benefit_diagram"
datafile: "cost_benefit_diagram.csv"
imagename: "cost_benefit_diagram.png"
scatter-point-color: "blue"
x-axis-title: "Punkte"
x-axis-line-pos: 80
x-axis-line-label: "Kosten-Minimum"
x-axis-line-type: "--"
x-axis-line-color: "red"
y-axis-title: "Kosten"
```

```yaml
y-axis-line-pos: 80
y-axis-line-label: "Punkte-Minimum"
y-axis-line-type: "--"
y-axis-line-color: "green"
title: "Kosten-Nutzen-Diagramm Beispiel"
```

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

VIII pandas_dataframe_to_latex_table.py

```python
import os
import pandas as pd
import yaml
from pathlib import Path
import chardet

import csv

# Get the Configuration
def load_configuration(plt_conf_filename):
    panda_latex_tables_config = dict()
    plt_conf_dir = os.path.join(os.path.dirname(os.getcwd()), 'source', 'configuration')
    yaml_path = os.path.join(plt_conf_dir, plt_conf_filename)

    with open(yaml_path, "r") as file:
        panda_latex_tables_config = yaml.load(file, Loader=yaml.FullLoader)

    return panda_latex_tables_config


def get_data(startpath, destination, tablefilename, datafile_path, datafile, alternative_cvs_load, separator, decimal):
    # Config Variables
    if startpath == 'homedir':
        directory = os.path.join(os.getcwd(), datafile_path)
    else:  # parentdir
        directory = os.path.join(os.path.dirname(os.getcwd()), datafile_path)

    # get the Datas as dirct
    data_path = os.path.join(directory, datafile)

    # load datas from csv into dict
    detected = chardet.detect(Path(data_path).read_bytes())
    encoding = detected.get("encoding")

```

```
35     # if alternative_cvs_load:
36     #     with open(data_path, 'r', encoding=encoding) as file:
37     #         reader = csv.reader(file)
38     #         data = list(reader)
39     #
40     #     # panda_table_data = pd.DataFrame(data, columns=data[0])
41     #     # panda_table_data = pd.read_csv(data_path, sep=separator, decimal=
       decimal, encoding=encoding, lineterminator='\n', engine='python')
42     #     panda_table_data = pd.read_csv(data_path, sep=separator, decimal=decimal
       , encoding=encoding, lineterminator='\n')
43     #     df_dtype = {
44     #         "Nr.": int,
45     #         "Anforderung": str,
46     #         "Beschreibung": str,
47     #         "System": str,
48     #         "Muss / Kann": str
49     #     }
50     #     # panda_table_data = pd.read_csv(data_path, sep=",", decimal=".",
       encoding=encoding, lineterminator='\n', dtype=df_dtype)
51     #     # panda_table_data = pd.read_csv(data_path, sep=",", decimal=".",
       encoding=encoding)
52     # else:
53     #     panda_table_data = pd.read_csv(data_path, sep=separator, decimal=decimal
       , encoding=encoding)
54     # panda_table_data = pd.read_csv(data_path, sep=",", decimal=".", encoding=
       encoding, low_memory=False, engine='python')
55     # panda_table_data = pd.read_csv(data_path, sep=",", decimal=".", encoding=
       encoding, engine='python', dtype='unicode')
56     # readed = open(data_path, 'r', encoding=encoding)
57     # panda_table_data = pd.read_csv(open(data_path, 'r', encoding=encoding), sep
       =",", decimal=".", encoding=encoding)
58     # panda_table_data = pd.read_csv(data_path, sep=",", decimal=".", encoding = "
       ISO-8859-1")
59     # panda_table_data = pd.read_csv(data_path, sep=",", decimal=".", encoding=
       encoding, chunksize=10)
60
61     # for chunk in pd.read_csv(data_path, sep=",", decimal=".", encoding=encoding,
        chunksize=5):
62     #     print(chunk)
63     # panda_table_data = pd.DataFrame()
64     # temp = pd.read_csv(data_path, iterator=True, sep=",", decimal=".", encoding=
       encoding, chunksize=1000)
65     # panda_table_data = pd.concat(temp, ignore_index=True)
66
67     df_dtype = {
68         "Nr.": int,
69         "Anforderung": str,
70         "Beschreibung": str,
```

```
71          "System": str,
72          "Muss / Kann": str
73      }
74      # panda_table_data = pd.read_csv(data_path, sep=",", decimal=".", encoding=
    encoding, engine='python', dtype=df_dtype)
75      # panda_table_data = pd.read_csv(data_path, sep=",", decimal=".", encoding=
    encoding, dtype=df_dtype)
76
77      # import dask.dataframe as dd
78      # df = dd.read_csv(data_path, sep=",", decimal=".", encoding=encoding)
79      # panda_table_data = df
80      print(encoding)
81      panda_table_data = pd.read_csv(data_path, sep=separator, decimal=decimal,
    encoding=encoding)
82      # return data
83      return panda_table_data
84
85
86  def create_latex_tables(panda_latex_tables_config):
87      plt_tables = panda_latex_tables_config.get('tables_inventory')
88      for table_item in plt_tables:
89          # id and filesystem informations
90          table_id = panda_latex_tables_config.get('tables').get(table_item).get('id
    ')
91          isbigfile = panda_latex_tables_config.get('tables').get(table_item).get('
    isbigfile')
92          has_longtexts = panda_latex_tables_config.get('tables').get(table_item).
    get('has_longtexts')
93          if isbigfile or has_longtexts:
94              alternative_cvs_load = True
95          else:
96              alternative_cvs_load = False
97          startpath = panda_latex_tables_config.get('tables').get(table_item).get('
    startpath')
98          destination = panda_latex_tables_config.get('tables').get(table_item).get(
    'destination_path')
99          tablefilename = panda_latex_tables_config.get('tables').get(table_item).
    get('tablefilename')
100         datafile_path = panda_latex_tables_config.get('tables').get(table_item).
    get('datafile_path')
101         datafile = panda_latex_tables_config.get('tables').get(table_item).get('
    datafile')
102         if startpath == 'homedir':
103             directory = os.path.join(os.getcwd(), destination)
104         else:  # parentdir
105             directory = os.path.join(os.path.dirname(os.getcwd()), destination)
106         tablefile = os.path.join(directory, tablefilename)
107         separator = panda_latex_tables_config.get('tables').get(table_item).get('
```

```
separator')
108         decimal = panda_latex_tables_config.get('tables').get(table_item).get('
decimal')
109
110         # column operations
111         column_operations = panda_latex_tables_config.get('tables').get(table_item
).get('column_operations').get('datas')
112
113         # group by / aggregation
114         groupby_values = panda_latex_tables_config.get('tables').get(table_item).
get('group_by')
115         group_by_function = panda_latex_tables_config.get('tables').get(table_item
).get('group_by_function')
116         # selected_rows = panda_latex_tables_config.get('tables').get(table_item).
get('selected_rows')
117         agg_funtion = panda_latex_tables_config.get('tables').get(table_item).get(
'agg_funtion')
118         agg_colums = panda_latex_tables_config.get('tables').get(table_item).get('
agg_colums')
119         # dropping and renaming columns
120         drop_columns = panda_latex_tables_config.get('tables').get(table_item).get
('drop_columns')
121         rename_columns = panda_latex_tables_config.get('tables').get(table_item).
get('rename_columns')
122
123         # table filtering and sorting
124         where_clausel = panda_latex_tables_config.get('tables').get(table_item).
get('where_clausel')
125         order_by = panda_latex_tables_config.get('tables').get(table_item).get('
sorting').get('order_by')
126         sort_acending = panda_latex_tables_config.get('tables').get(table_item).
get('sorting').get('sort_acending')
127         sort_inplace = panda_latex_tables_config.get('tables').get(table_item).get
('sorting').get('sort_inplace')
128
129         # pivot settings
130         pivot = panda_latex_tables_config.get('tables').get(table_item).get('pivot
')
131         pivot_column = panda_latex_tables_config.get('tables').get(table_item).get
('pivot_columns')
132         pivot_value = panda_latex_tables_config.get('tables').get(table_item).get(
'pivot_values')
133
134         # pivot_table settings
135         pivot_table = panda_latex_tables_config.get('tables').get(table_item).get(
'pivot_table')
136         pivot_table_column = panda_latex_tables_config.get('tables').get(
table_item).get('pivot_table').get(
```

```
            'pivot_columns')
        pivot_table_value = panda_latex_tables_config.get('tables').get(table_item).get('pivot_table').get(
            'pivot_values')
        pivot_table_agg_function = panda_latex_tables_config.get('tables').get(table_item).get('pivot_table').get(
            'pivot_agg_func')
        pivot_table_indizes = panda_latex_tables_config.get('tables').get(table_item).get('pivot_table').get(
            'pivot_index').get('pivot_indizes')
        pivot_table_indizes_visible = panda_latex_tables_config.get('tables').get(table_item).get('pivot_table').get(
            'pivot_index').get('pivot_indizes_visible')
        pivot_table_rename_indizes = panda_latex_tables_config.get('tables').get(table_item).get('pivot_table').get(
            'pivot_index').get('pivot_rename_indizes')

        # margins (subtotals)
        margin = panda_latex_tables_config.get('tables').get(table_item).get('margins').get('margin')
        margin_name = panda_latex_tables_config.get('tables').get(table_item).get('margins').get('margin_name')

        # table settings
        table_caption = panda_latex_tables_config.get('tables').get(table_item).get('caption')
        table_label = panda_latex_tables_config.get('tables').get(table_item).get('label')
        table_style = panda_latex_tables_config.get('tables').get(table_item).get('table_styles')
        sparse_columns = panda_latex_tables_config.get('tables').get(table_item).get('table_styles').get(
            'sparse_columns')
        table_caption_position = panda_latex_tables_config.get('tables').get(table_item).get('table_styles').get(
            'props').get('caption-side')
        table_position = panda_latex_tables_config.get('tables').get(table_item).get('table_styles').get('props').get(
            'position')
        longtable = panda_latex_tables_config.get('tables').get(table_item).get('table_styles').get('props').get(
            'longtable')
        linebreak_columns = panda_latex_tables_config.get('tables').get(table_item).get('table_styles').get('props').get(
            'linebreak_columns')
        resize_textwidth = panda_latex_tables_config.get('tables').get(table_item).get('table_styles').get('props').get(
            'resize_textwidth')
```

```
169
170         # get the pandas (panda data)
171         panda_table_data = get_data(startpath, destination, tablefilename,
    datafile_path, datafile, alternative_cvs_load, separator, decimal)
172
173         # filter by where clausel
174         if where_clausel:
175             panda_table_data = panda_table_data.query(where_clausel)
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
    pivot_table_column:
183         if groupby_values and not (pivot_column or (pivot_table_column or
    pivot_table_value or pivot_table_indizes)):
184             match group_by_function:
185                 case 'max':
186                     panda_table_data = panda_table_data.groupby(groupby_values,
    as_index=False).max()
187                 case 'min':
188                     panda_table_data = panda_table_data.groupby(groupby_values,
    as_index=False).min()
189                 case 'head':
190                     panda_table_data = panda_table_data.groupby(groupby_values,
    as_index=False).head()
191                 case 'sum':
192                     panda_table_data = panda_table_data.groupby(groupby_values,
    as_index=False).sum()
193                 case 'mean':
194                     panda_table_data = panda_table_data.groupby(groupby_values,
    as_index=False).mean()
195         else:
196             panda_table_data = panda_table_data
197
198         # pivot if pivot is selected
199         if pivot_table_column or pivot_table_value or pivot_table_indizes:
200             if type(pivot_table_agg_function) is list:
201                 agg_tuple = tuple(pivot_table_agg_function)
202                 panda_table_data = pd.pivot_table(panda_table_data, index=
    pivot_table_indizes,
203                                                   columns=pivot_table_column,
    values=pivot_table_value,
204                                                   aggfunc=agg_tuple, margins=
    margin, margins_name=margin_name)
205             elif type(pivot_table_agg_function) is dict:
```

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

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

```
            # for number, line in lines.split('\n'):

                # Condition true if the key exists in the line
                # If true then display the line number
                if table_type in line:
                    # print(f'{key} is at line {number}')
                    resize_line_nr = number
                    resize_line = line

            line_table_resize = resize_line + "\n" + "\\resizebox{\\columnwidth}{!}{%"
            latex_table = latex_table.replace(resize_line, line_table_resize)

            # replace table end with bracket
            resize_line_nr = 0
            resize_line = ""
            if longtable:
                table_type = '\\end{longtable}'
            else:
                table_type = '\\end{table}'

            for number, line in enumerate(lines, 1):

                # Condition true if the key exists in the line
                # If true then display the line number
                if table_type in line:
                    # print(f'{key} is at line {number}')
                    resize_line_nr = number
                    resize_line = line

            line_table_resize =  "}" + "\n" + resize_line
            latex_table = latex_table.replace(resize_line, line_table_resize)

        # caption below is not supported yet (pandas 2.2)
        # replace caption and replace table end with the caption line and table end
        if table_caption_position == 'below':
            caption_label = "\\caption{" + table_caption + "} \\label{" + table_label + "} \\\\"
            caption_label_nbr = "\\caption{" + table_caption + "} \\label{" + table_label + "}"
            caption_only = "\\caption{" + table_caption + "} \\\\"
            caption_only_nbr = "\\caption{" + table_caption + "}"
            label_only = "\\label{" + table_label + "} \\\\"
            label_only_nbr = "\\label{" + table_label + "}"
            latex_table = latex_table.replace(caption_label, '')
            latex_table = latex_table.replace(caption_only, '')
            latex_table = latex_table.replace(label_only, '')
```

```
            latex_table = latex_table.replace(caption_label_nbr, '')
            latex_table = latex_table.replace(caption_only_nbr, '')
            latex_table = latex_table.replace(label_only_nbr, '')

            if longtable:
                table_string = '\\end{longtable}'
                new_caption = caption_label_nbr + "\n" + table_string
                latex_table = latex_table.replace(table_string, new_caption)
            else:
                table_string = '\\end{table}'
                new_caption = caption_label_nbr + "\n" + table_string
                latex_table = latex_table.replace(table_string, new_caption)


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


# run the methods / functions
panda_latex_tables_config = load_configuration('csv_to_latex_diplomarbeit.yaml')
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

```
tables_inventory:
 - "db_inventory"
 - "db_inventory_per_rdbms"
 - "db_inventory_per_os"
 - "anforderungskatalog"
 - "arbeitsrapport"
 - "projektcontrolling"
 - "evaluation_inventory"
 - "dependencis"
 - "predecision_out"
 - "predecision_in"
 - "project_comments"
 - "evaluation_distributed_sql"
 - "expert_discussions_overview"
 - "expert_discussions_full_list"
 - "stakeholder"
tables:
 db_inventory:
  id: "db_inventory"
  isbigfile:
```

```yaml
  has_longtexts: False
  separator: ","
  decimal: "."
  caption: "Datenbankinventar - Roh"
  label: "db_inventory"
  startpath: "parentdir"
  destination_path: "content/latex_tables"
  datafile_path: "source/tables"
  datafile: "inventory.csv"
  tablefilename: "db_inventory.tex"
  decimal_format:
  group_by:
  group_by_function:
  agg_funtion:
  agg_colums:
  drop_columns:
   - "comment"
   - "eol"
   - "eol_since"
   - "releasedate"
  column_operations:
   datas:
   operations:
    dauer_summe:
     operation_function:
     axis_number:
     columns:
  pivot:
   pivot_columns:
   pivot_values:
  pivot_table:
   pivot_index:
    pivot_indizes_visible:
    pivot_rename_indizes:
   pivot_columns:
   pivot_values:
   pivot_agg_func:
  rename_columns:
   server: "Server - Hostname"
   os: "OS"
   rdbms: "RDBMS"
   instance: "Instanz"
   databases: "Datenbanken"
   appliance: "Appliance"
   comment: "Kommentar"
   version: "Version"
   releasedate: "Version - Releasedatum"
   eol: "EoL"
```

```yaml
    age: "Version - Alter"
    eol_since: "EoL seit"
  where_clausel:
  sorting:
    order_by:
      - "server"
      - "rdbms"
    sort_acending: True
    sort_inplace: True
  margins:
    margin: False
    margin_name:
  table_styles:
    selector: "caption"
    props:
      caption-side: "below"
      position: "H"
      sparse_columns: True
      longtable: True
      resize_textwidth: False
      linebreak_columns:
      table_header: True
db_inventory_per_rdbms:
  id: "db_inventory_per_rdbms"
  isbigfile:
  has_longtexts: False
  separator: ","
  decimal: "."
  caption: "Datenbankinventar"
  label: "db_inventory_per_rdbms"
  startpath: "parentdir"
  destination_path: "content/latex_tables"
  datafile_path: "source/tables"
  datafile: "inventory.csv"
  tablefilename: "db_inventory_per_rdbms.tex"
  decimal_format:
  group_by:
    - "rdbms"
  group_by_function: "sum"
  agg_funtion:
  agg_colums:
    - "rdbms"
  drop_columns:
    - "server"
    - "os"
    - "version"
    - "releasedate"
    - "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 üaufgeschlsselt"
168   label: "db_inventory_per_os"
169   startpath: "parentdir"
170   destination_path: "content/latex_tables"
171   datafile_path: "source/tables"
172   datafile: "inventory.csv"
173   tablefilename: "db_inventory_per_os.tex"
174   decimal_format:
175   group_by:
176    - "rdbms"
177    - "os"
178   group_by_function: "sum"
179   agg_funtion:
180   agg_colums:
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:
```

```
    instance: "sum"
    databases: "sum"
    appliance: "sum"
   transpose: True
  rename_columns:
   rdbms: "RDBMS"
   instance : "Instanz"
   databases : "Datenbanken"
   os : "OS"
   appliance: "Appliance"
  where_clausel:
  sorting:
   order_by:
   sort_acending: False
   sort_inplace: True
  margins:
   margin: True
   margin_name: "Gesamtergebnis"
  table_styles:
   selector: "caption"
   props:
    caption-side: "below"
    position: "H"
    sparse_columns: True
    longtable: True
    resize_textwidth: False
    linebreak_columns:
    table_header: True
 anforderungskatalog:
  id: "anforderungskatalog"
  isbigfile:
  has_longtexts: True
  separator: ";"
  decimal: "."
  caption: "Anforderungskatalog"
  label: "anforderungskatalog"
  startpath: "parentdir"
  destination_path: "content/latex_tables"
  datafile_path: "source/tables"
  datafile: "anforderungskatalog.CSV"
  tablefilename: "anforderungskatalog.tex"
  decimal_format:
  group_by:
  group_by_function:
  agg_funtion:
  agg_colums:
  drop_columns:
  column_operations:
```

```
    datas:
    operations:
     dauer_summe:
      operation_function:
      axis_number:
      columns:
  pivot:
   pivot_columns:
   pivot_values:
  pivot_table:
   pivot_index:
    pivot_indizes_visible: False
    pivot_rename_indizes:
   pivot_columns:
   pivot_values:
   pivot_agg_func:
  rename_columns:
  where_clausel:
  sorting:
   order_by:
    - "Nr."
   sort_acending: True
   sort_inplace: True
  margins:
   margin: False
   margin_name:
  table_styles:
   selector: "caption"
   props:
    caption-side: "below"
    position: "H"
    sparse_columns: False
    longtable: False
    resize_textwidth: True
    linebreak_columns:
     - "Beschreibung"
    table_header: True
 arbeitsrapport:
  id: "arbeitsrapport"
  isbigfile:
  has_longtexts: False
  separator: ";"
  decimal: "."
  caption: "Arbeitsrapport"
  label: "arbeitsrapport"
  startpath: "parentdir"
  destination_path: "content/latex_tables"
  datafile_path: "source/tables"
```

```
309  datafile: "arbeitsrapport.CSV"
310  tablefilename: "arbeitsrapport.tex"
311  decimal_format: "{:0.1f}"
312  group_by:
313  group_by_function:
314  agg_funtion:
315  agg_colums:
316  drop_columns:
317   - "Hide"
318   - "Geplante Dauer [h]"
319   - "dauer_summe"
320  column_operations:
321   datas:
322   operations:
323    dauer_summe:
324     operation_function:
325     axis_number:
326     columns:
327  pivot:
328   pivot_columns:
329   pivot_values:
330  pivot_table:
331   pivot_index:
332    pivot_indizes_visible: False
333    pivot_rename_indizes:
334   pivot_columns:
335   pivot_values:
336   pivot_agg_func:
337  rename_columns:
338  where_clausel: "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:
```

```yaml
      - "äTtigkeit"
      - "Bemerkung"
      - "Schwierigkeit"
      - "öLsungen"
    table_header: True
projektcontrolling:
 id: "projektcontrolling"
 isbigfile:
 has_longtexts: False
 separator: ";"
 decimal: "."
 caption: "Projektcontrolling"
 label: "projektcontrolling"
 startpath: "parentdir"
 destination_path: "content/latex_tables"
 datafile_path: "source/tables"
 datafile: "arbeitsrapport.CSV"
 tablefilename: "projektcontrolling.tex"
 decimal_format: "{:0.1f}"
 group_by:
  - "Phase"
  - "Subphase"
 group_by_function: "sum"
 agg_funtion:
 agg_colums:
     - "Dauer [h]"
     - "Geplante Dauer [h]"
     - "dauer_summe"
 drop_columns:
  - "Datum"
  - "Von"
  - "Bis"
  - "Hide"
  - "äTtigkeit"
  - "Bemerkung"
  - "Schwierigkeit"
  - "öLsungen"
 column_operations:
  datas:
   - "dauer_summe"
  operations:
   dauer_summe:
    operation_function: "diff"
    axis_number: 1
    columns:
     - "Dauer [h]"
     - "Geplante Dauer [h]"
 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: "Evaluationssyssteme"
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_colums:
454   drop_columns:
455   column_operations:
456    datas:
457    operations:
458     dauer_summe:
459      operation_function:
460      axis_number:
461      columns:
462   pivot:
463    pivot_columns:
464    pivot_values:
465   pivot_table:
466    pivot_index:
467     pivot_indizes_visible: False
468     pivot_rename_indizes:
469    pivot_columns:
470    pivot_values:
471    pivot_agg_func:
472   rename_columns:
473   where_clausel:
474   sorting:
475    order_by:
476     - "Server"
477     - "Typ"
478    sort_acending: True
479    sort_inplace: True
480   margins:
481    margin: False
482    margin_name:
483   table_styles:
484    selector: "caption"
485    props:
486     caption-side: "below"
487     position: "H"
488     sparse_columns: True
489     longtable: True
490     resize_textwidth: False
491     linebreak_columns:
492     table_header: True
493 dependencis:
494   id: "dependencis"
495   isbigfile:
496   has_longtexts: False
497   separator: ";"
498   decimal: "."
499   caption: "äAbhngigkeiten"
500   label: "dependencis"
```

```yaml
  startpath: "parentdir"
  destination_path: "content/latex_tables"
  datafile_path: "source/tables"
  datafile: "dependencis.csv"
  tablefilename: "dependencis.tex"
  decimal_format:
  group_by:
  group_by_function:
  agg_funtion:
  agg_colums:
  drop_columns:
  column_operations:
   datas:
   operations:
    dauer_summe:
     operation_function:
     axis_number:
     columns:
  pivot:
   pivot_columns:
   pivot_values:
  pivot_table:
   pivot_index:
    pivot_indizes_visible: False
    pivot_rename_indizes:
   pivot_columns:
   pivot_values:
   pivot_agg_func:
  rename_columns:
  where_clausel:
  sorting:
   order_by:
    - "Nr."
   sort_acending: True
   sort_inplace: True
  margins:
   margin: False
   margin_name:
  table_styles:
   selector: "caption"
   props:
    caption-side: "below"
    position: "H"
    sparse_columns: True
    longtable: False
    resize_textwidth: True
    linebreak_columns:
     - "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_colums:
572   drop_columns:
573    - "hide_state"
574   column_operations:
575    datas:
576    operations:
577     dauer_summe:
578      operation_function:
579      axis_number:
580      columns:
581   pivot:
582    pivot_columns:
583    pivot_values:
584   pivot_table:
585    pivot_index:
586     pivot_indizes_visible: False
587     pivot_rename_indizes:
588    pivot_columns:
589    pivot_values:
590    pivot_agg_func:
591   rename_columns:
592   where_clausel: "hide_state == 1"
593   sorting:
594    order_by:
595     - "Nr."
596    sort_acending: True
```

```
    sort_inplace: True
  margins:
    margin: False
    margin_name:
  table_styles:
    selector: "caption"
    props:
      caption-side: "below"
      position: "H"
      sparse_columns: True
      longtable: False
      resize_textwidth: True
      linebreak_columns:
        - "üBegrndung"
      table_header: True
 predecision_in:
  id: "predecision_in"
  isbigfile:
  has_longtexts: False
  separator: ";"
  decimal: "."
  caption: "Vorauswahl - Evaluation"
  label: "predecision_in"
  startpath: "parentdir"
  destination_path: "content/latex_tables"
  datafile_path: "source/tables"
  datafile: "pre-decision.csv"
  tablefilename: "pre-decision-in.tex"
  decimal_format:
  group_by:
  group_by_function:
  agg_funtion:
  agg_colums:
  drop_columns:
    - "hide_state"
  column_operations:
    datas:
    operations:
      dauer_summe:
        operation_function:
        axis_number:
        columns:
  pivot:
    pivot_columns:
    pivot_values:
  pivot_table:
    pivot_index:
      pivot_indizes_visible: False
```

```
    pivot_rename_indizes:
   pivot_columns:
   pivot_values:
   pivot_agg_func:
  rename_columns:
  where_clausel: "hide_state == 2"
  sorting:
   order_by:
    - "Nr."
   sort_acending: True
   sort_inplace: True
  margins:
   margin: False
   margin_name:
  table_styles:
   selector: "caption"
   props:
    caption-side: "below"
    position: "H"
    sparse_columns: True
    longtable: False
    resize_textwidth: True
    linebreak_columns:
     - "üBegrndung"
    table_header: True
 project_comments:
  id: "project_comments"
  isbigfile:
  has_longtexts: False
  separator: ";"
  decimal: "."
  caption: "Kommentare - Anmerkung"
  label: "project_comments"
  startpath: "parentdir"
  destination_path: "content/latex_tables"
  datafile_path: "source/tables"
  datafile: "pre-fazit.csv"
  tablefilename: "pre-fazit.tex"
  decimal_format:
  group_by:
  group_by_function:
  agg_funtion:
  agg_colums:
  drop_columns:
  column_operations:
   datas:
   operations:
    dauer_summe:
```

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

```
  label: "expert_discussions_overview"
  startpath: "parentdir"
  destination_path: "content/latex_tables"
  datafile_path: "source/tables"
  datafile: "expert_discussions.csv"
  tablefilename: "expert_discussions_overview.tex"
  decimal_format:
  group_by:
  group_by_function:
  agg_funtion:
  agg_colums:
  drop_columns:
     - "Fragen"
     - "Antworten"
     - "Sonstige Themen"
  column_operations:
   datas:
   operations:
    dauer_summe:
     operation_function:
     axis_number:
     columns:
  pivot:
   pivot_columns:
   pivot_values:
  pivot_table:
   pivot_index:
    pivot_indizes_visible: False
    pivot_rename_indizes:
   pivot_columns:
   pivot_values:
   pivot_agg_func:
  rename_columns:
  where_clausel:
  sorting:
   order_by:
    - "äFachgesprch"
   sort_acending: True
   sort_inplace: True
  margins:
   margin: False
   margin_name:
  table_styles:
   selector: "caption"
   props:
    caption-side: "below"
    position: "H"
    sparse_columns: True
```

```yaml
    longtable: False
    resize_textwidth: True
    linebreak_columns:
     - "Studenten"
     - "Bemerkungen"
    table_header: True
expert_discussions_full_list:
  id: "expert_discussions_full_list"
  isbigfile:
  has_longtexts: False
  separator: ";"
  decimal: "."
  caption: "äFachgesprche - Protokoll"
  label: "expert_discussions_full_list"
  startpath: "parentdir"
  destination_path: "content/latex_tables"
  datafile_path: "source/tables"
  datafile: "expert_discussions.csv"
  tablefilename: "expert_discussions_full_list.tex"
  decimal_format:
  group_by:
  group_by_function:
  agg_funtion:
  agg_colums:
  drop_columns:
  column_operations:
   datas:
   operations:
    dauer_summe:
     operation_function:
     axis_number:
     columns:
  pivot:
   pivot_columns:
   pivot_values:
  pivot_table:
   pivot_index:
    pivot_indizes_visible: False
    pivot_rename_indizes:
   pivot_columns:
   pivot_values:
   pivot_agg_func:
  rename_columns:
  where_clausel:
  sorting:
   order_by:
    - "äFachgesprch"
   sort_acending: True
```

```
885     sort_inplace: True
886   margins:
887     margin: False
888     margin_name:
889   table_styles:
890     selector: "caption"
891     props:
892       caption-side: "below"
893       position: "H"
894       sparse_columns: True
895       longtable: False
896       resize_textwidth: True
897       linebreak_columns:
898         - "Studenten"
899         - "Fragen"
900         - "Antworten"
901         - "Sonstige Themen"
902         - "Bemerkungen"
903       table_header: True
904 stakeholder:
905   id: "stakeholder"
906   isbigfile:
907   has_longtexts: False
908   separator: ";"
909   decimal: "."
910   caption: "Stakeholder"
911   label: "stakeholder"
912   startpath: "parentdir"
913   destination_path: "content/latex_tables"
914   datafile_path: "source/tables"
915   datafile: "stakeholder.csv"
916   tablefilename: "stakeholder.tex"
917   decimal_format:
918   group_by:
919   group_by_function:
920   agg_funtion:
921   agg_colums:
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:
```

```yaml
  pivot_table:
   pivot_index:
    pivot_indizes_visible: False
    pivot_rename_indizes:
   pivot_columns:
   pivot_values:
   pivot_agg_func:
  rename_columns:
  where_clausel:
  sorting:
   order_by:
   sort_acending: True
   sort_inplace: True
  margins:
   margin: False
   margin_name:
  table_styles:
   selector: "caption"
   props:
    caption-side: "below"
    position: "H"
    sparse_columns: True
    longtable: False
    resize_textwidth: True
    linebreak_columns:
    table_header: True
```

Listing 6: Python LaTex - csv_to_latex_diplomarbeit.yaml - Konfigurationsdatei - CSV - LaTex-Tabelle