TALLINN UNIVERSITY OF TECHNOLOGY
School of Information Technologies

Daniil Mandrikov 206443IAAB

Selection and Implementation of a NewSQL Platform for Pipedrive

Bachelor's thesis

Supervisor: Juri Hudolejev
MSc

Tallinn 2023

TALLINNA TEHNIKAÜLIKOOL
Infotehnoloogia teaduskond

Daniil Mandrikov 206443IAAB

NewSQL platvormi valik ja rakendamine Pipedrive'i jaoks

Bakalaureusetöö

Juhendaja: Juri Hudolejev
MSc

Tallinn 2023

Author's declaration of originality

I, Daniil Mandrikov, hereby declare that this thesis Selection and Implementation of a NewSQL Platform for Pipedrive, submitted in partial fulfilment of the requirements for the degree of Bachelor at Tallinn University of Technology, is my original work.

I confirm that:

1. All the ideas, information, and words of others used in this thesis/dissertation have been properly cited and credited.
2. I have not used any material that is copyrighted or that belongs to someone else without their permission.
3. I have not engaged in any form of academic dishonesty, such as plagiarism or fabrication of data.

I understand that any violation of academic integrity may result in serious consequences, including the cancellation of my degree.

Signed: Daniil Mandrikov
Date: 15.04.2023

Abstract

Pipedrive is a growing sales CRM company that has been experiencing challenges with its current database system in terms of performance, scalability, and availability. This thesis presents a comparative analysis of three NewSQL databases to determine the most suitable replacement for Pipedrive's current database system.

The study begins with an overview of database technologies, including SQL, NoSQL, and NewSQL, and their advantages and disadvantages. A requirements analysis is then conducted to define the requirements for Pipedrive's new database system.

The study evaluates and compares three NewSQL databases, namely CockroachDB, TiDB, and YugabyteDB, based on the defined requirements. A recommendation is made based on the comparison, and the implementation and integration of the chosen database system are described, along with the challenges encountered and how they were overcome. Performance and failover testing were conducted on both the old and new database systems, and the results were analysed and compared.

The study concludes with a discussion of the significance of the findings and potential future work, including possible improvements to the new database system.

This study provides a valuable resource for businesses seeking to replace or upgrade their database systems.

Thesis is written in English and is 49 pages long, including 6 chapters, 12 figures and 2 tables.

Annotatsioon

Pipedrive on kasvav CRM-i müügiettevõte, mis on oma praeguse andmebaasisüsteemiga kokku puutunud väljakutsetega nii jõudluse, mastaapsuse kui ka saadavuse osas. See lõputöö esitab kolme NewSQL-i andmebaasi võrdleva analüüsi, et teha kindlaks Pipedrive'i praegusele andmebaasisüsteemile sobivaim asendus.

Uuring algab andmebaasitehnoloogiate, sealhulgas SQL, NoSQL ja NewSQL, ning nende eeliste ja puuduste kirjanduse ülevaatega. Seejärel viiakse läbi nõuete analüüs, et määratleda nõuded Pipedrive'i uuele andmebaasisüsteemile.

Uuringus hinnatakse ja võrreldakse kolme NewSQL-i andmebaasi, nimelt CockroachDB, TiDB ja YugabyteDB, lähtudes määratletud nõuetest. Võrdluse põhjal koostatakse soovitus ning kirjeldatakse valitud andmebaasisüsteemi juurutamist ja integreerimist, väljakutseid, millega kokku puututi ja kuidas neid ületati. Jõudlus- ja tõrkekontrollitestid viidi läbi nii vanadel kui ka uutel andmebaasisüsteemidel ning tulemusi analüüsiti ja võrreldi.

Uuringu lõpetab arutelu leidude olulisuse ja võimaliku tulevase töö üle, sealhulgas uue andmebaasisüsteemi võimalike täiustuste üle.

See uuring pakub väärtuslikku ressurssi ettevõtetele, kes soovivad oma andmebaasisüsteeme välja vahetada või uuendada.

Lõputöö on kirjutatud inglise keeles ja on 49 lehekülge pikk, sisaldab 6 peatükke, 12 jooniseid ja 2 tabeleid.

List of abbreviations and terms

ACID	Atomicity, Consistency, Isolation, Durability [1].
CDC	Change Data Capture is a technique used in database systems to capture and propagate changes made to data in real-time [35].
Control Plane	Set of components that manage the state of the cluster and make decisions about the scheduling and placement of workloads. These components work together to ensure that the cluster operates in a consistent and reliable manner [37].
CP	Consistency with Partition Tolerance. The concept of Consistency with Partition Tolerance (CP) comes from the CAP theorem, which states that a distributed system can only guarantee two out of the following three properties: consistency, availability, and partition tolerance. In the context of CP, a distributed system prioritizes consistency over availability in the face of network partitions. This means that in the event of a network partition, the system will sacrifice availability to maintain a consistent view of the data across all nodes [33].
CRD	Custom Resource Definition
CRM	Customer Relationship Management [2].
GTID	Global Transaction Identifier

HAProxy	HAProxy is a free, very fast and reliable reverse-proxy offering high availability, load balancing, and proxying for TCP and HTTP-based applications [46].
Helm	Open-source package manager for Kubernetes that simplifies the deployment and management of applications on Kubernetes clusters. It provides a templating engine that enables users to define, install, and upgrade applications and services as a single deployable unit called a chart [38].
Kafka	Distributed streaming platform used for building real-time data pipelines and streaming applications. It allows for the processing of high-throughput, low-latency data streams in a fault-tolerant and scalable manner [34].
KV	Key-Value
LDAP/AD	Lightweight Directory Access Protocol. Directory server.
OLAP	Online Analytical Processing [1].
OLTP	Online Transaction Processing [1].
PD	Placement Driver
ProxySQL	ProxySQL, an open-source MySQL proxy server, acts as a mediator between applications and a MySQL server, enhancing performance through traffic distribution across multiple database servers and ensuring availability through automatic failover to standby servers in the event of database server failures [45].
RDBMS	Relational Database Management System [1].

Serverless Cluster	Type of cloud computing service that enables to run applications on a fully-managed infrastructure without having to worry about the underlying hardware or software [36].
SQL	Structured Query Language [1].
UDF	User-Defined Function [1].

Table of Contents

List of figures

List of tables

1 Introduction

Over the past 30 years, the database research has demonstrated an exceptional productivity, resulting in the database system becoming a crucial development in the field of software engineering. With the database as the underlying framework of the information system, it has significantly transformed the operations of various organizations. Recent advancements in this technology have introduced more powerful and user-friendly systems, making them increasingly accessible to a wider range of users. However, the apparent simplicity of these systems has led to users developing databases and applications without the necessary knowledge and skills, resulting in ineffective and inefficient systems. Unfortunately, this phenomenon perpetuates the ongoing 'software crisis' or 'software depression'[1].

1.1 Background on Pipedrive's sales management system

Pipedrive is a deal-driven customer relationship management (CRM) platform that is widely used by businesses of all sizes to manage their sales pipelines and customer interactions. The platform is known for its intuitive user interface, powerful automation capabilities, and robust reporting features. Pipedrive has become a key player in the CRM industry, with over 100,000 customers in more than 170 countries [3].

In addition to its core CRM features, Pipedrive also offers a range of integrations with other tools and services, such as email marketing platforms, project management software, and accounting software. This makes it a versatile platform that can be customized to suit the needs of individual businesses [3].

The platform has helped countless businesses to streamline their sales processes and improve their bottom line [3].

1.2 SQL functionality in Pipedrive

Pipedrive's current database management system is based on MySQL, which is a popular open-source relational database management system (RDBMS). MySQL is known for its ease of use, scalability, and reliability, which makes it a popular choice for web applications like Pipedrive. All the user data, including activities, contacts, leads, and deals, are kept in Pipedrive's MySQL database [39].

1.2.1 Pipedrive MySQL architecture

The Pipedrive MySQL architecture utilizes Percona MySQL clusters in combination with standalone machines and primary/replica pairs to ensure scalability and high availability. Percona MySQL is an open-source distribution of MySQL that provides enhanced features and functionality. Pipedrive's architecture is geographically distributed across multiple Openstack regional deployments and one AWS management regional deployment to provide worldwide access to its services and flexibility to support a growing customer base. Additionally, Pipedrive operates in five live regions, each with thousands of primary/replica pairs to provide redundancy and high availability to customers. [39] [40].

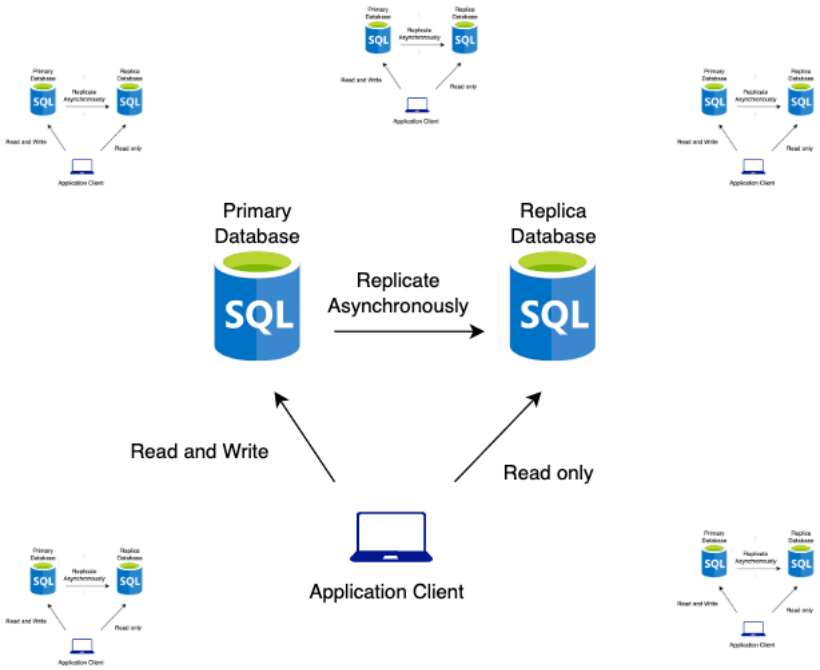

Figure 1. Pipedrive MySQL architecture.

1.2.2 Understanding data storage limits in Pipedrive

To effectively design and operate the Pipedrive infrastructure and application, it is important to understand the data storage limits associated with the platform. In particular, it is essential to ensure that the system can handle the load of the largest amount of data that may be generated over the course of a year, taking into account any anticipated growth or expansion. When it comes to Pipedrive, two key data storage limits to consider are activities and persons. Over the course of a year, the amount of data associated with activities is expected to grow from 1.2 billion to 1.47 billion records, while the amount of data associated with persons is expected to grow from 600 million to 765 million records. (It is likely that the large spikes observed in the data were caused by troubles in processing the data, such as processing errors or data anomalies.)

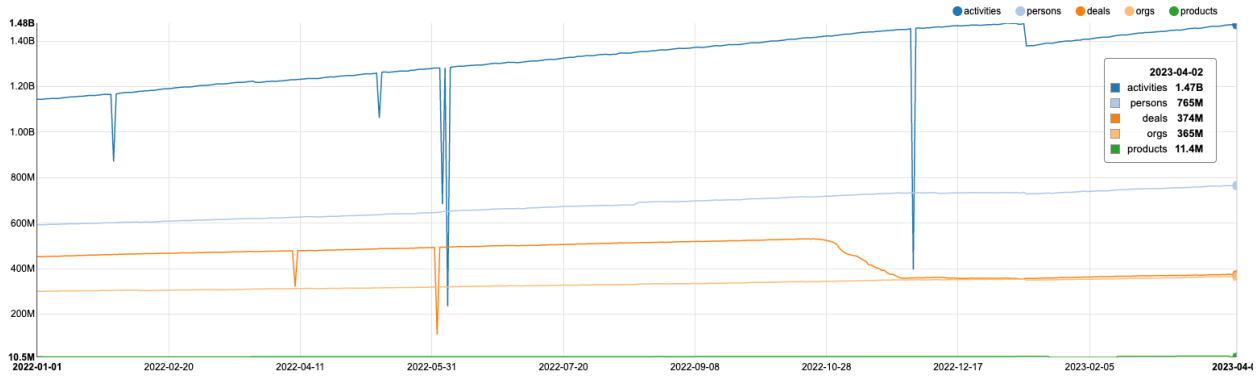

Figure 2. Actual Data Growth for Pipedrive Activities and Persons.

1.3 Limitations of Pipedrive's MySQL database management

While MySQL is a reliable and scalable database management system, Pipedrive's increasing volume of data and traffic is putting a strain on the current system. Some of the limitations of Pipedrive's current database management system include:

1. Performance issues: As the volume of data and traffic on the platform increases, the current MySQL database may struggle to process and retrieve data quickly, resulting in slower load times and performance issues. This can lead to frustration for users who rely on the platform for their daily operations, as well as decreased productivity and potentially lost revenue.

2. Limited scalability: While MySQL is a scalable database management system, there may come a point where it cannot handle the growing volume of data and traffic on Pipedrive without some additional optimizations. This may result in reduced system capacity, increased latency, and overall slower system performance, which can limit the platform's ability to grow and meet the needs of its users.

3. Potential downtime: If the database system is unable to handle the increased load, it may lead to potential downtimes, which can be detrimental to the user experience and customer satisfaction. Downtimes can also result in lost productivity and revenue, as users are unable to access the platform and complete their tasks. Additionally, repeated downtimes can erode trust in the platform and lead to users seeking out alternative solutions.

1.4 Problem identification in Pipedrive's database management

Improving Pipedrive's system performance is a crucial issue that needs to be addressed in order to maintain and enhance the user experience. One possible solution is to implement a more efficient and scalable database management system that can handle the increasing volume of data and traffic on the platform.

NewSQL platforms are designed to meet the demands of high-performance, distributed systems, making them a suitable choice for Pipedrive. By exploring and evaluating NewSQL platforms, Pipedrive can find a platform that is best suited to its needs and can address its performance issues [7].

Implementing a suitable NewSQL platform can significantly improve the platform's performance and scalability, leading to faster load times, better response times, and overall improvement in the user experience. This can have a positive impact on customer satisfaction, retention, and acquisition. Additionally, by mitigating performance issues, potential downtimes can be avoided, ensuring that users can access the platform without interruptions.

Overall, by selecting and implementing a suitable NewSQL platform, Pipedrive can improve its system performance, scalability, and reliability, leading to a better user experience and increased customer satisfaction.

This study will focus on the selection and implementation of a suitable NewSQL platform for Pipedrive's database management system. The study will not cover the development of the CRM application or the evaluation of other aspects of the system architecture. The limitations of this study include the availability of data and resources for evaluating and implementing the selected NewSQL platform.

2 Technology Review

Database Management Systems (DBMS) are essential software applications used to manage and organize data in various forms. Different types of DBMS platforms have emerged over the years, each with its own strengths and limitations [1].

2.1 SQL overview

SQL databases have been widely used in various industries, including finance, healthcare, logistics and e-commerce due to their reliability and consistency in managing data. They provide a structured and standardized way of storing and retrieving data, making them ideal for applications that require consistency and accuracy.

However, SQL databases may have limitations in terms of scalability and flexibility, as their rigid table structure and predefined schema can make it difficult to modify or scale out the database as the application grows. Additionally, SQL databases may not be suitable for applications that require frequent updates or real-time data processing, as they may struggle to keep up with the volume and velocity of incoming data [4].
Despite these limitations, SQL databases continue to be a popular choice for many applications and businesses. With proper design and optimization, SQL databases can offer high performance and reliability, making them a reliable option for managing critical data [4].

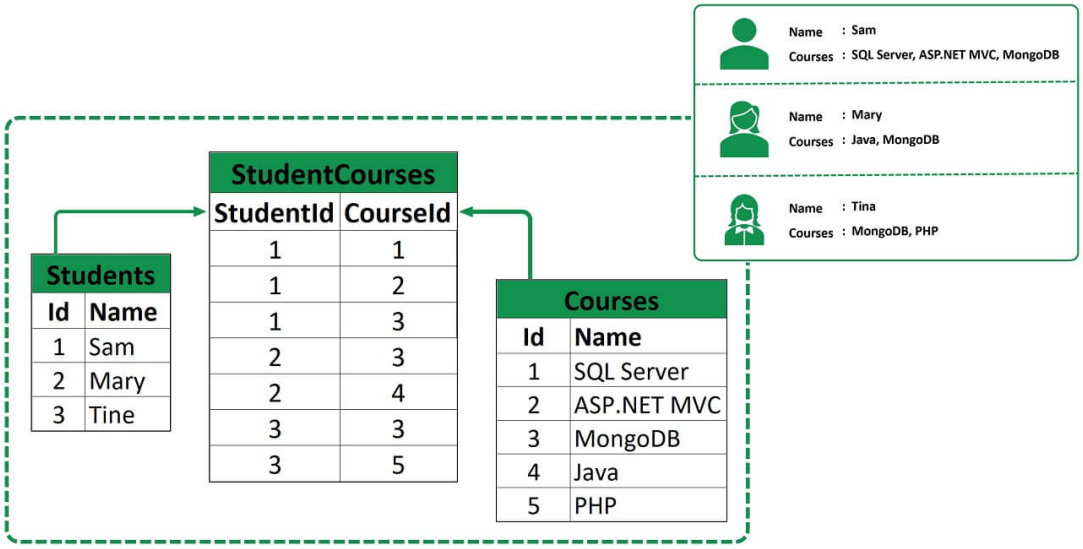

Figure 3. Example of SQL Database Entity Relationship Diagram.

2.2 NoSQL overview

NoSQL databases are non-relational database management systems that allow for more flexible data structures, making them well-suited for handling unstructured data. They can be horizontally scalable, meaning they can handle a high volume of data and traffic, but may have limitations in data consistency and query complexity [5]. Examples of NoSQL databases include MongoDB, Cassandra, and Couchbase.

One of the advantages of NoSQL databases is their ability to handle large volumes of unstructured and semi-structured data, such as social media feeds, clickstream data, and sensor data. NoSQL databases can also be highly scalable and fault-tolerant, with built-in redundancy and replication features. Additionally, NoSQL databases can offer faster performance and lower latency compared to traditional SQL databases, particularly for read-heavy workloads [5].

However, NoSQL databases may have limitations in data consistency, as they typically sacrifice consistency in favor of availability and partition tolerance. This can lead to data inconsistencies and conflicts, particularly in distributed environments. NoSQL databases may also have a steeper learning curve compared to SQL databases and may require more specialized knowledge and expertise to set up and manage [6].

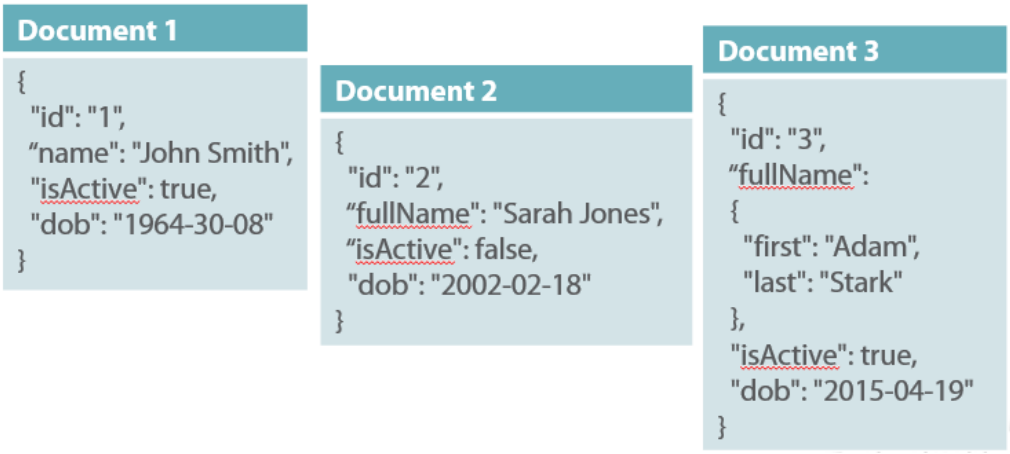

Figure 4. Example of NoSQL Database with JSON Data.

2.3 NewSQL overview

Recently, a third type of DBMS platform has emerged, known as NewSQL databases. NewSQL databases aim to combine the benefits of traditional SQL and NoSQL databases by providing both scalability and consistency. They use distributed architecture to scale horizontally but maintain transactional consistency and ACID compliance [7].

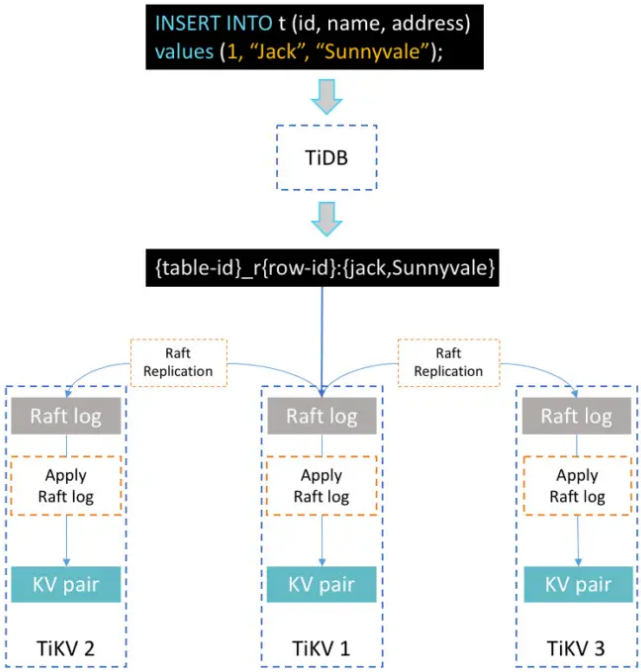

Figure 5. Example of NewSQL Database Architecture with TiDB.

NewSQL databases are typically designed with a shared-nothing architecture, where each node in the cluster has its own set of processors, memory, and storage. This approach helps to

reduce contention and improve scalability, as each node can process transactions independently. Many NewSQL databases rely on in-memory processing to speed up query processing and reduce latency. By storing data in memory rather than on disk, NewSQL systems can achieve extremely fast read and write performance. To ensure consistency across nodes in the cluster, many NewSQL databases use distributed consensus protocols such as Paxos or Raft. These protocols help to ensure that all nodes in the cluster agree on the state of the database, even in the face of node failures or network partitions. NewSQL databases often use automatic sharding to distribute data across nodes in the cluster. This approach helps to improve performance by reducing the amount of data that needs to be processed by each node [7].

NewSQL databases have become increasingly popular due to their ability to combine the benefits of traditional SQL and NoSQL databases. While there are clear advantages to using NewSQL platforms, there are also some disadvantages to consider [7].

Advantages of NewSQL platforms include horizontal scalability: databases can scale horizontally by adding more nodes to a cluster, which allows for increased performance and handling of high volumes of data and traffic. High performance for complex queries and transactions through the use of in-memory processing and distributed architecture, and transactional consistency for maintaining data integrity and reliability [7].

NewSQL databases have several potential disadvantages to consider. Firstly, they are a relatively new technology and may not have the same level of support and tooling as more established databases like MySQL or Oracle. This can make it harder to find skilled developers or third-party tools that integrate with the database. NewSQL databases can be complex to set up and manage, particularly in larger deployments, as they typically require more nodes to achieve high levels of scalability. This not only increases the complexity of the system as a whole, but also makes NewSQL databases more expensive than traditional SQL databases, particularly when it comes to hardware and infrastructure costs.

2.3.1 Summary of NewSQL database research studies

NewSQL databases can be complex to set up and maintain, requiring expertise in distributed systems and database administration.

Furthermore, NewSQL platforms are not as widely adopted as traditional SQL or NoSQL databases and may require expensive hardware and infrastructure to achieve the desired performance and scalability.

Research on NewSQL performance and scalability is still in its early stages, but some studies have been conducted.

These studies suggest that NewSQL databases can provide higher performance and scalability than traditional SQL databases and NoSQL databases for certain workloads. Additionally, NewSQL databases can achieve high transactional consistency while still providing good performance and scalability. However, tuning and optimizing NewSQL databases for the best performance and scalability may be non-trivial task.

3 Methodology

The aim of this research is to select and implement a NewSQL platform for Pipedrive, a cloud-based sales CRM software company. In order to ensure that the selected platform meets Pipedrive's specific needs and requirements, we have identified several key criteria that the platform must satisfy.

3.1 Defining requirements for Pipedrive's database system

The following requirements were set for the new platform:

- Scalability is a crucial factor for Pipedrive, so the platform must be able to handle high volumes of data and traffic. Automatic sharding and online re-sharding are important features for scalability, as they allow for the addition of new nodes to the cluster without disrupting operations. Scale out and scale up capabilities are also necessary for supporting Pipedrive's growth.

- High availability is another critical requirement for Pipedrive, as downtime or data loss could have severe consequences for the business. The selected platform must be able to support a minimum number of replicas online and have a high commit level for data consistency.

- Maintenance and failover procedures must also be in place to ensure continuous availability.

- Backup and restore capabilities are necessary for disaster recovery and business continuity. The selected platform must support schema dump and point-in-time backup and restore procedures.

- Monitoring is essential for maintaining performance and identifying potential issues. Grafana with alerting features is preferred for monitoring the selected platform.

- The platform must also have certain features, such as the ability to be pluggable to Kafka via binlog, be relational, support transactions (ACID compliant), indices, and online altering.

- Cloud native design is preferred, i.e. selected platform development and deployment that is optimized for cloud infrastructure. This involves using containerization, microservices, and distributed systems to create applications that are scalable, flexible, and portable. Cloud native design enables organizations to take full advantage of cloud features and benefits, including improved performance, cost savings, and greater agility.

- The license and documentation level, including additional development, market research, success stories, and support level/price, are also important factors for consideration.

3.2 Potential candidates

After considering these criteria, we have identified TiDB, CockroachDB, and YugabyteDB as potential candidates for Pipedrive's NewSQL platform. All three platforms are distributed SQL databases that are designed to be highly available, scalable, and resilient. They are parts of the emerging category of NewSQL databases, which combine the benefits of traditional SQL databases with the advantages of NoSQL databases.

3.2.1 TiDB

TiDB is a distributed SQL database that is designed to handle large-scale data with high availability and scalability. One of the key features of TiDB is its ability to scale both horizontally and vertically. To scale horizontally, TiDB supports automatic sharding and online re-sharding, which makes it easy to split a TiDB cluster into multiple smaller clusters,

each handling a subset of the data. This approach allows TiDB to add more smaller clusters as needed, rather than scaling up a single large cluster. By doing so, the impact of individual node failures or hotspots in the data can be reduced, which can improve performance and availability.

In addition to horizontal scaling, TiDB can also be scaled out by adding more nodes to the cluster. TiDB can also be scaled up by increasing the amount of CPU, memory, or storage resources on each node in the cluster [8].

TiDB ensures high availability by requiring a minimum number of replicas to be online at all times, which can be configured on a per-shard basis and can range from 1 to the total number of replicas for that shard. The default value is 3. TiDB also supports a variety of commit levels, including 1PC, 2PC, and Raft, which allow users to choose the level of fault tolerance they require. 1PC only requires one replica to acknowledge the transaction before it is considered committed, making it the fastest but offering the least durability. 2PC requires a majority of replicas to acknowledge the transaction, offering better durability but may be slower than 1PC. Raft uses the Raft consensus algorithm to determine commit decisions, meaning that a majority of replicas must agree on the commit decision before it is considered committed. This is the most durable commit level but may be slower than 2PC [13].

TiDB also supports rolling maintenance and upgrade, allowing users to perform maintenance operations (such as upgrading software or hardware) on one node at a time without impacting the overall availability of the system [8].

TiDB supports automatic failover, automatically promoting one of the replicas to be the new leader for that shard. The time required for failover depends on the commit level and the number of replicas for that shard. All these features make TiDB a highly available and fault-tolerant distributed database system [9].

TiDB supports backup and restore operations, including schema dump to backup and restore database structure and point-in-time recovery to restore the database to a specific point in time [10].

TiDB provides a monitoring framework that includes a Grafana dashboard for monitoring the database and supports alerting. With Grafana and alerting, users can quickly identify issues and take action to maintain the database's health [11].

TiDB is a relational database that is MySQL protocol compliant and supports most of the standard SQL syntax used by MySQL including data types, and indexing options such as primary keys, secondary indices, and full-text search indices. TiDB can export data changes from binlog to Kafka [12].

TiDB supports ACID-compliant transactions and online altering, making it suitable for mission-critical applications [12].

To replicate schema changes to all the nodes in the cluster, TiDB uses a technique called "Schema Replicant", and for online schema changes, TiDB supports several types of online schema changes, including adding or dropping columns, changing the data type of a column, and changing the column order [12].

However, TiDB lacks support for:

- stored procedures and functions
- triggers
- events
- user-defined functions
- FOREIGN KEY constraints
- FULLTEXT syntax and indices
- SPATIAL (also known as GIS/GEOMETRY) functions
- data types
- character sets other than ascii, latin1, binary, utf8, utf8mb4, and gbk,
- optimizer trace
- XML functions
- column-level privileges
- CREATE TABLE tblName AS SELECT stmt syntax
- CHECK TABLE syntax
- CHECKSUM TABLE syntax

- REPAIR TABLE syntax
- OPTIMIZE TABLE syntax
- HANDLER statement
- CREATE TABLESPACE statement

TiDB is designed to be cloud native, supporting deployment on Kubernetes and other container orchestration platforms [12].

It is open-source and licensed under the Apache License 2.0, and has documentation that covers installation, configuration, and usage of the database.
TiDB is developed and maintained by PingCAP, a company that is dedicated to building an open-source, cloud native database that can handle mission-critical workloads. TiDB has a vibrant developer community, with over 900 contributors on GitHub, and provides various support plans including premium providing 24/7 support and guaranteed response times [14].

3.2.2 CockroachDB

CockroachDB is a scalable and highly available distributed SQL database that supports sharding, automatic range partitioning, and rebalancing of data across multiple nodes in a cluster. It is designed to handle large-scale, high-throughput workloads, making it an ideal choice for applications that require high scalability and availability [15].

One of the key features of CockroachDB is its support for sharding, which enables the distribution of data across multiple nodes in a cluster. CockroachDB automatically shards data based on a configurable range partitioning scheme, and it supports online rebalancing and range splitting, which can be triggered manually or automatically based on performance metrics [15].

CockroachDB is designed to scale out horizontally across multiple nodes in a cluster, with automatic load balancing and rebalancing of data. Additionally, it can also scale up vertically within a node, by using faster hardware or adding more resources [15].

To ensure high availability, CockroachDB supports configurable replication factors, allowing users to specify the minimum number of replicas that must be online to maintain data availability in the face of node failures or network partitions. CockroachDB uses a consensus algorithm based on the Raft protocol to ensure that data is replicated and committed across

nodes in a cluster. The number of replicas that must acknowledge a transaction before it is considered committed can be configured [16].

CockroachDB provides a range of maintenance features, including online schema changes, automatic data compaction, and automatic vacuuming. It also uses automatic failover to ensure that nodes can be replaced in the event of a failure, with automatic promotion of replicas to maintain data availability [17].

For backup and restore, CockroachDB provides a schema dump tool called cockroach dump, which allows users to generate a SQL script that can be used to recreate the schema of a database. It also supports point-in-time backups, which can be used to restore a cluster to a specific point in time in case of data loss or corruption [18].

CockroachDB supports a range of features for enterprise use cases. One of the most notable enterprise features of CockroachDB is its support for exporting events from binlog to Kafka, which enables real-time data streaming to external systems. CockroachDB is also a SQL database that supports the SQL-92 standard, as well as the PostgreSQL wire protocol and a subset of the MySQL protocol. It is not designed to work with the MySQL protocol natively. However, it is possible to use CockroachDB with MySQL in some cases using an open-source tool called Vitess. The database supports distributed transactions with ACID semantics, ensuring consistency and durability across nodes in a cluster. Additionally, CockroachDB supports a range of indexing options, including primary and secondary indices, full-text search, and geospatial indexing [20].

CockroachDB also supports online schema changes, which allows users to modify the schema of a live cluster without downtime. However, there are some important features that CockroachDB does not support:

- stored procedures and functions
- user-defined functions (UDFs)
- foreign keys with ON DELETE CASCADE
- full-text search
- change data capture, and follower reads

Furthermore, multi-region clusters, self-service upgrades, and range management are also not supported in CockroachDB Serverless [20].

CockroachDB is designed to run natively in cloud environments, with support for Kubernetes and other container orchestration platforms. It provides a Grafana dashboard and Prometheus metrics for monitoring cluster performance and health, with configurable alerts and notifications. However, certain features such as the DB Console and audit logs are not supported in CockroachDB Serverless clusters [19][20].

CockroachDB is available as an open-source solution under the Apache 2.0 license. Cockroach Labs offers additional features and capabilities for CockroachDB Enterprise, such as advanced security features, performance optimization tools, and integrations with third-party tools and services. Enterprise-level support and services, such as 24/7 technical support, proactive monitoring, and personalized assistance, are available through Cockroach Labs' CockroachCloud platform and CockroachDB Enterprise offering. The free version of CockroachDB is designed to scale out horizontally across multiple nodes in a cluster, while CockroachDB Enterprise includes additional features to optimize scalability and performance. Pricing for CockroachCloud is based on usage, with a pay-as-you-go model that charges for the amount of storage and compute resources used. There are also options for reserved instances and annual contracts that offer discounted rates [21].

3.2.3 YugabyteDB

YugabyteDB is a distributed SQL database that offers various features to support scalability, high availability, and backup and restore. To achieve horizontal scalability, YugabyteDB supports automatic sharding and re-sharding, allowing the database to be partitioned and distributed across nodes in the cluster for optimal performance and availability. Re-sharding can be performed online, allowing scaling without downtime. Additionally, YugabyteDB can be scaled out by adding more nodes to the cluster or scaled up by increasing the resources available to each node [22].

To ensure high availability, YugabyteDB supports configurable replication factors, allowing users to specify the minimum number of replicas that must be online to maintain data availability in the face of node failures or network partitions. YugabyteDB can automatically promote a replica to take the place of a failed replica to ensure that the minimum number of

replicas is always met. YugabyteDB uses a distributed consensus algorithm based on the Raft protocol to ensure data is replicated and committed across nodes in a cluster [23].

Additionally, YugabyteDB provides maintenance features such as online schema changes, backups and restores, and node upgrades, along with automatic failover to ensure nodes can be replaced in the event of a failure [23].

For backup and restore, YugabyteDB provides a schema dump tool called "ysql_dump," which allows users to dump the schema and data of their cluster to a set of files for disaster recovery purposes or to recreate the database on another cluster. The tool supports dumping the schema and data of all or selected databases, tables, and indices in the cluster and can dump data in parallel to improve performance. It supports incremental backups, which reduce the amount of data that needs to be backed up after the initial full backup, and point-in-time recovery, which allows users to restore a database to a specific point in time [24].

YugabyteDB offers observability through a Grafana dashboard and alerting capabilities for proactive monitoring [25].

It also supports CDC through a pluggable architecture that allows exporting data changes from binlog to Kafka [26].

YugabyteDB is a fully relational database that supports the MySQL wire protocol, making it easy to integrate with existing MySQL client applications. It is not designed to work with the MySQL protocol natively. However, it is possible to use YugabyteDB with MySQL in some cases using an open-source tool called Vitess. It is ACID-compliant, supporting distributed transactions and a range of indexing options, including primary, secondary, and unique indices [27].

While YugabyteDB does not yet support certain features such as user-defined functions, stored procedures, or MySQL-specific functions like GROUP_CONCAT, it supports Kubernetes, Docker, and other cloud native technologies [28].

YugabyteDB is an open-source database management system that comes with an Apache 2.0 license. Along with the open-source version, YugabyteDB offers an Enterprise Edition with advanced features for commercial use [29].

YugabyteDB's commercial version provides an array of advanced features such as distributed backup and restore, automatic failover and high availability, online schema changes, and cutting-edge security features including encryption and LDAP/AD integration. Additionally, this license includes priority support from the Yugabyte team [30].

The documentation library of YugabyteDB covers all aspects of the system, including installation, configuration, operation, and management [31].

3.3 Summary of NewSQL candidates and their features

Here is a comparison of three different database management systems: TiDB, CockroachDB and YugabyteDB. The table compares these systems across several categories, including architecture, consistency model, high availability, scalability, sharding, replication, distributed transactions, ACID compliance, indices, backup and restore, monitoring, cloud native, open-source, community support, commercial model, SQL dialect.

Table 1. Comparison of NewSQL Databases.

Feature	TiDB	YugabyteDB	CockroachDB
Consistency Model	CP	CP	CP
High Availability	Yes	Yes	Yes
Scalability	Yes	Yes	Yes
Sharding	Automatic	Automatic	Automatic
Replication	Yes	Yes	Yes
Distributed Transactions	Yes	Yes	Yes
ACID Compliance	Yes	Yes	Yes
Indices	B-tree	B-tree	B-tree, inverted index

Backup and Restore	Yes	Yes	Yes
Monitoring	Yes	Yes	Yes
Cloud Native	Yes	Yes	Yes
Kubernetes Support	Yes	Yes	Yes
Open-source	Yes	Yes	Yes
Community Support	Yes	Yes	Yes
Commercial Model	Yes (Important features are included in the free version)	Yes (Important features are included in the free version)	Yes (Important features are included in the free version)
SQL Dialect	MySQL	PostgreSQL	PostgreSQL

Based on the comparison table, TiDB appears to be a good option to consider for several reasons. TiDB offers good support for MySQL, which can make it easier for organizations that have already invested in MySQL to switch to TiDB without significant retraining. TiDB is open-source and has a large community of developers contributing to its development, which can ensure continuous improvements and updates.

Furthermore, TiDB has a commercial model that can provide enterprise-grade features and support for organizations that require additional functionalities and professional support.

CockroachDB and YugabyteDB are only PostgreSQL compatible and require additional work for MySQL protocol support.

4 Implementation and integration

This chapter describes the process of deploying TiDB, a NewSQL database system, on a Kubernetes cluster using Terraform and Ansible.

4.1 TiDB architecture

The architecture of TiDB consists of several components that work together to provide a high-performance, scalable database solution. These components include:

- TiDB Cluster: The TiDB cluster is the front-end layer of the TiDB system that provides a standard SQL interface for clients to interact with the database. It includes the TiDB server, which parses SQL statements, optimizes queries, and executes them across the cluster [9].
- Storage Cluster: The storage cluster consists of two components: TiKV and TiFlash. TiKV is a distributed, transactional Key-Value store that provides the primary storage engine for TiDB. It stores data on disk and provides high availability through replication and automatic failover. TiFlash is a columnar storage engine that provides fast analytical queries for OLAP workloads. It stores data in memory and on disk and is optimized for read-heavy workloads [9].
- PD Cluster: PD is the cluster manager for TiDB. It is responsible for maintaining the metadata of the TiDB cluster, including information about the location and status of TiKV and TiFlash nodes. PD also manages the distribution of data across the cluster, ensuring that data is evenly distributed and available in the event of node failures [9].
- KV API: The KV API is the storage engine for TiDB. It provides a distributed key-value store that stores data across the TiKV and TiFlash nodes. The KV API is responsible for ensuring data consistency and availability, as well as managing the distribution of data across the cluster [9].

Overall, the architecture of TiDB is designed to provide a highly available, scalable, and performant database solution that can handle both OLTP and OLAP workloads. The use of distributed components such as PD, TiKV, and TiFlash allows for horizontal scaling and automatic failover, ensuring that the database is always available and performing well [9].

4.2 Infrastructure setup

The TiDB cluster was set up in a Kubernetes cluster in the Pipedrive environment via custom Terraform resource, that directly connects to Openstack. Cluster was set up consisting of one control plane with 2 CPU and 2GB RAM, and five nodes with 8 CPU and 32GB RAM each. All the machines were effortlessly built using a Jenkins job. This powerful automation tool executes the deployment of Terraform resources in the correct order, ensuring the seamless assembly of the entire cluster. With just a simple click, the Jenkins job takes care of launching and integrating all the necessary components automatically. This streamlined process eliminates the need for manual intervention, making the construction of the cluster quick and hassle-free. Nodes were configured to use local storage to simplify the test setup. In production environments Pipedrive uses enterprise NAS to guarantee data availability and persistence, however integrating TiDB with this storage solution is outside of the scope of this study [41].

4.3 TiDB deployment

The TiDB cluster was deployed by following the TiDB documentation, without utilizing Helm charts. The deployment was guided by inspecting the manifests of the Helm charts and downloading the CRDs, which were adapted to meet the specific needs and requirements of the environment. The documentation was also used to troubleshoot issues and configure TiDB components. TiDB Kubernetes Operator was chosen to automate the deployment, management, and scaling of TiDB cluster.

Deployment for the Pipedrive environment did not use Helm charts, but instead relied on manifests. Manifests have several advantages over Helm charts, including version control using Git or other version control systems, which makes it easier to track changes, roll back to a previous version, or review code changes before deployment. Manifests also provide

greater flexibility, allowing for more control over the deployment process, customization of each component, and changes can be applied independently using various tools for deployment and management, such as kubectl [42].

Security is another benefit of using manifests, which are self-contained, unlike Helm charts that rely on external repositories that may contain third-party packages introducing security vulnerabilities [42].

Finally, manifests are faster to deploy and manage than Helm charts, which require additional processing and templating that can slow down the deployment process. Using Ansible to manage files, templates, and manifests is a suitable approach in the Pipedrive case, aligning with Pipedrive preferences and requirements. Overall, manifests offer more control, flexibility, security, and simplicity, and can be preferred over Helm charts in certain situations.

Ansible was used to deploy the TiDB cluster through a playbook, which defined roles, tasks, and variables. The playbook was tailored to the environment, including defining the node IP addresses and network settings.

Local storage class was used for our persistent volumes, which allows us to use the disks attached to each node as storage for our TiKV nodes.

The TiDB cluster was customized by making modifications to the CRDs and manifests from the Helm charts. The number of replicas and replication factor for the TiKV component were changed, and the storage engine for the TiDB component was configured. Additionally, environment variables were defined for the components, including the cluster name and log level. These changes were necessary to tailor the TiDB cluster to our specific requirements and environment.

The TiDB Operator provides a set of Kubernetes CRDs to define the components of the TiDB cluster. Manifests for the TiDB components, such as TiDB, TiKV, and PD, were also created to specify their configuration and interconnection. These manifests were applied to the Kubernetes cluster using Ansible. It should be noted that the original deployment documentation was researched thoroughly. However, as Helm was not being used, certain

setup steps were performed differently. Instead, the necessary manifests from the Helm charts were examined to determine the appropriate deployment configurations.

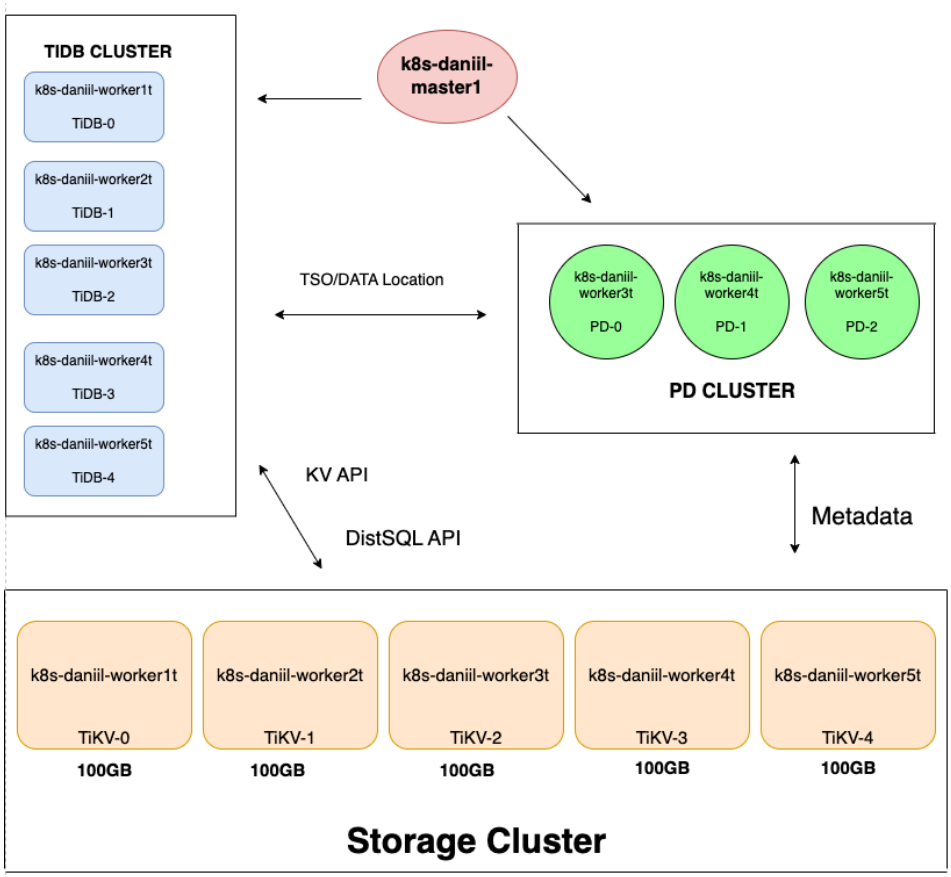

Figure 6. Pipedrive TiDB Kubernetes Deployment Model.

4.3.1 Terraform

Terraform is an infrastructure-as-code tool that enables defining infrastructure in code and applying it consistently across environments [43]. Terraform was utilized to define the Kubernetes cluster, including nodes, networking, and storage.

Using Terraform, a set of modules was defined to describe the Kubernetes cluster and its associated resources, such as:

- Kubernetes nodes with required software and settings
- Networking components like VPCs, subnets, and load balancers
- Storage classes for persistent volumes
- And more

Defining the infrastructure in code allows for easy recreation of the Kubernetes cluster in various environments and ensures consistency across all of them.

4.3.2 Ansible

Ansible, a robust open-source automation tool, was employed to automate the deployment and configuration of TiDB. The tool utilizes a simple declarative language to describe system configurations and can manage both infrastructure and application deployments [44].
A set of playbooks was defined using Ansible to install and configure all the essential components for the TiDB deployment, including

- Kubernetes and related tools like kubeadm, kubectl, and kubelet
- TiDB Operator and TiDB Cluster CRDs
- TiDB components like TiDB, TiKV, and PD
- Prometheus and Grafana for monitoring
- And more

Executing these playbooks enabled us to set up our TiDB deployment quickly and effortlessly, without the need for manual configuration of each component.

4.3.3 Local storage class

Nodes were configured to use local storage class volumes for this purpose.
First, a local-storage class was created using the hostPath provisioner. This class was then used to create persistent volumes that were used by the TiDB cluster. The local-storage class was defined in a YAML file, and the corresponding manifest was then applied to the Kubernetes cluster using Ansible.

Utilizing local storage for persistent volumes in the TiDB cluster bolstered the system's resilience against node failures and mitigated the potential for data loss. It is important to acknowledge that data protection in such scenarios primarily relies on replication across multiple nodes, rather than solely depending on local storage. Replication ensures data redundancy and availability, playing a vital role in safeguarding against the loss of critical information.

5 Performance and failover testing

Performance and failover testing were conducted on two database platforms, MySQL and TiDB. MySQL is a popular relational database management system, while TiDB is a distributed NewSQL database that combines the advantages of both SQL and NoSQL.

5.1 Configuration and environment

The test environment was set up with identical hardware and software configurations for both MySQL and TiDB. Sysbench was used to generate workload on the databases, including read-only and read-write transactions:

- Select random ranges
- Select
- Select random points
- Update index
- Insert
- OLTP (50% reads and 50% writes)

The number of concurrent connections was gradually increased, and the response time and throughput of the databases were monitored.

5.2 Performance comparison between MySQL and TiDB

Performance tests were conducted on two database platforms, MySQL and TiDB, using Sysbench to generate read-only and read-write transactions. The MySQL setup was optimized over time, while the TiDB tests were performed with four different setups and configurations.

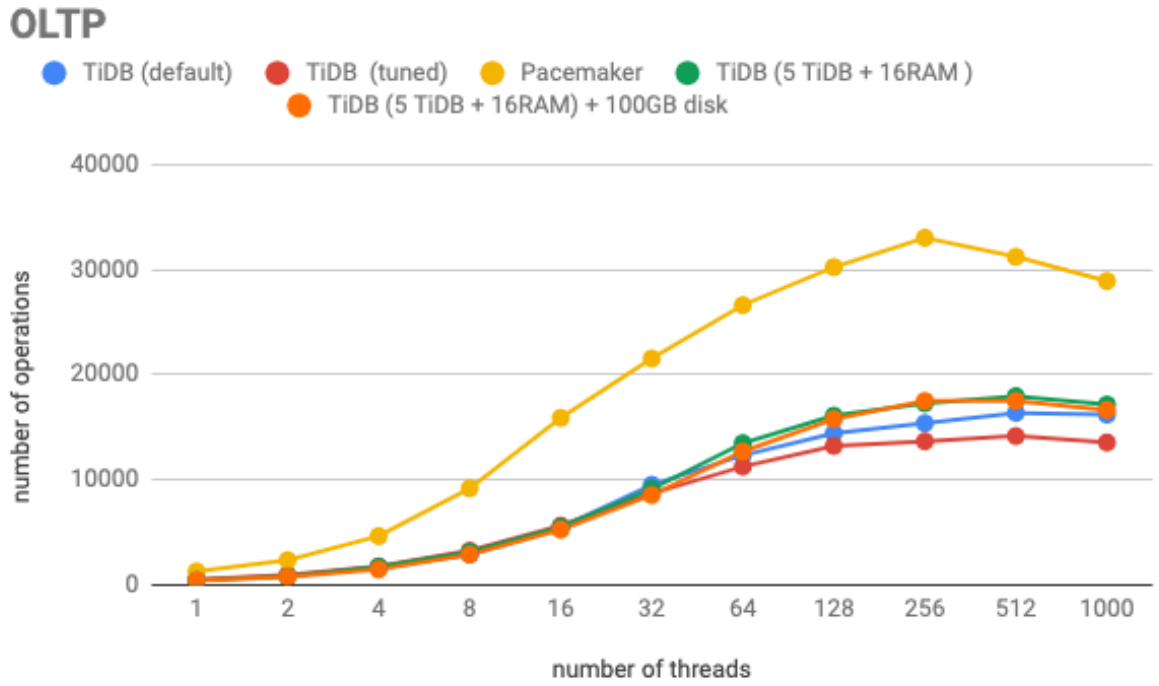

Figure 7. MySQL vs TiDB OLTP Performance Test Results.

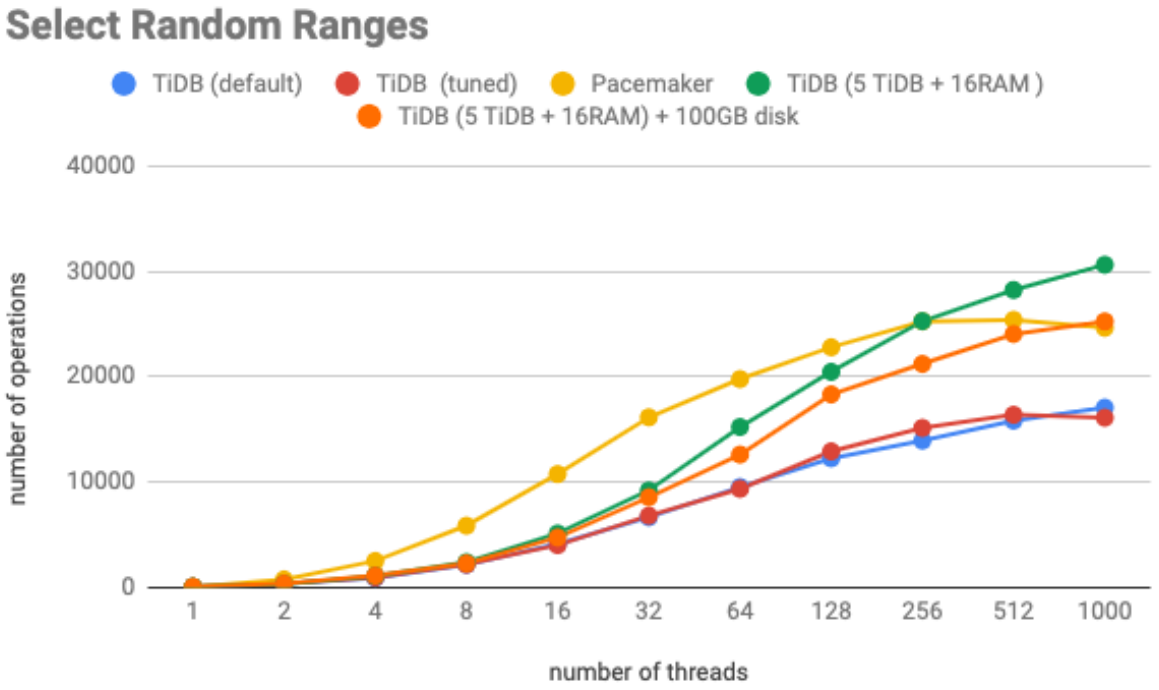

Figure 8. MySQL vs TiDB Select Random Ranges Performance Test Results.

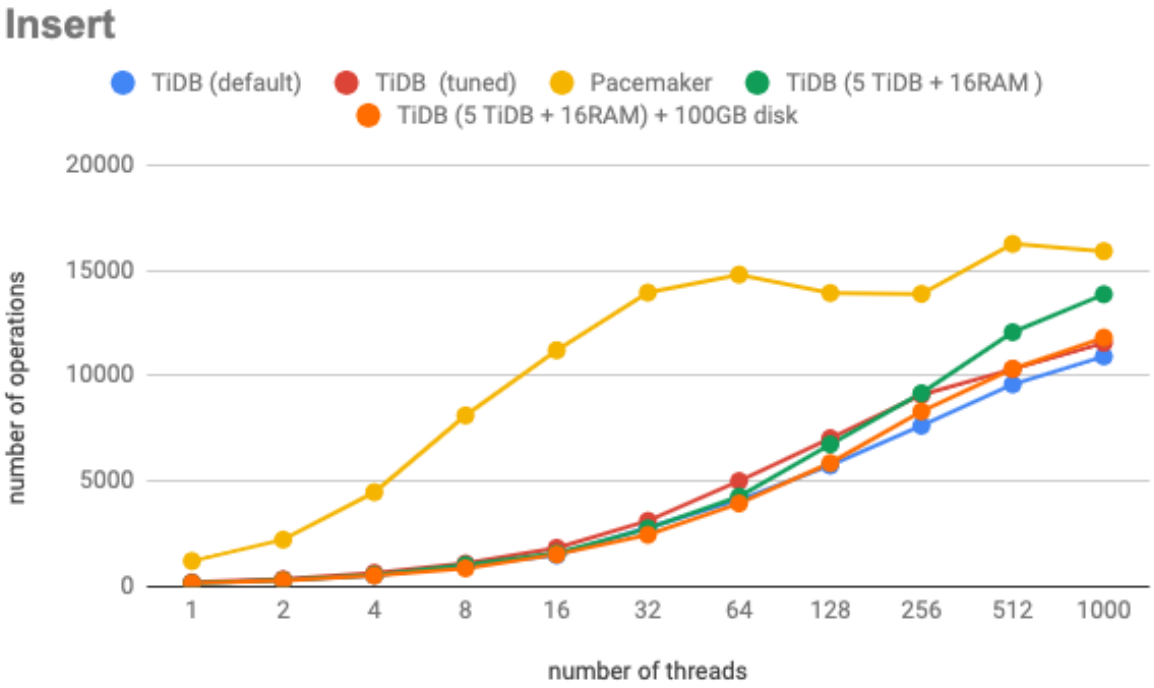

Figure 9. MySQL vs TiDB Insert Performance Test Results.

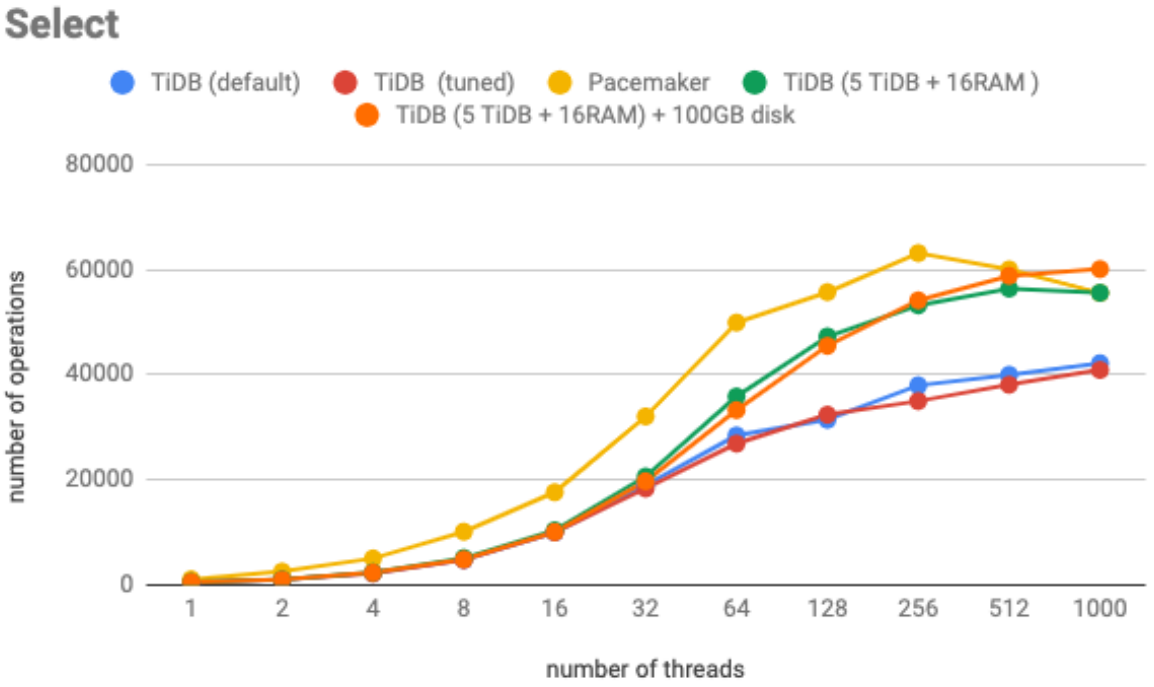

Figure 10. MySQL vs TiDB Select Performance Test Results.

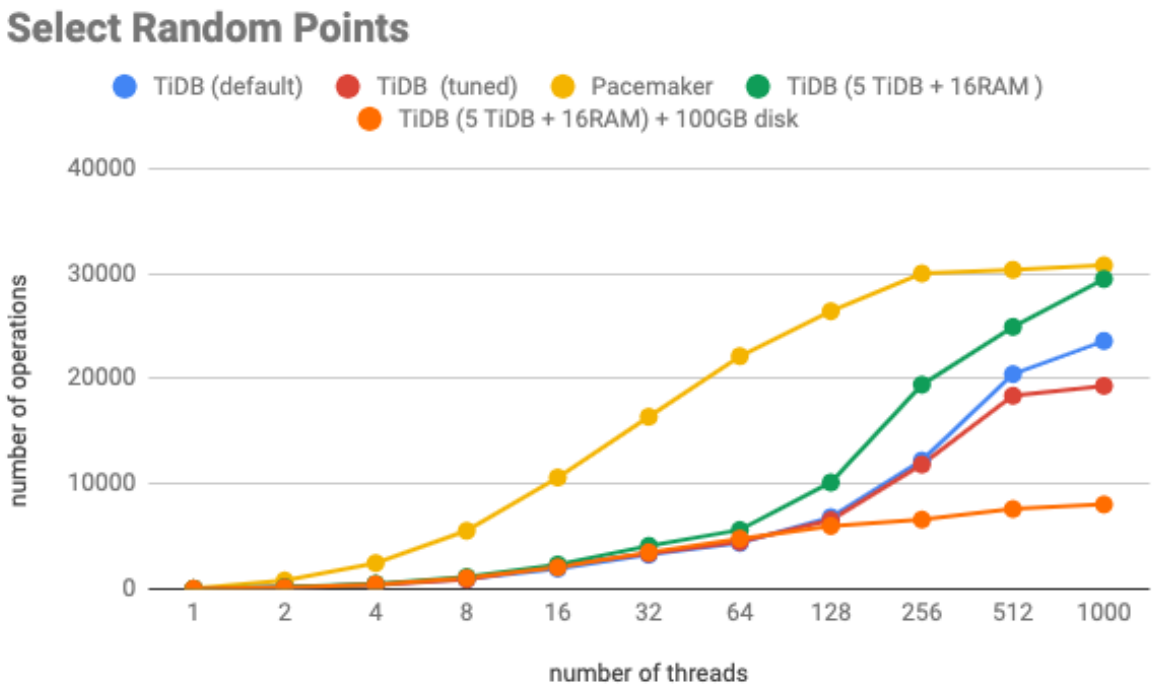

Figure 11. MySQL vs TiDB Select Random Points Performance Test Results.

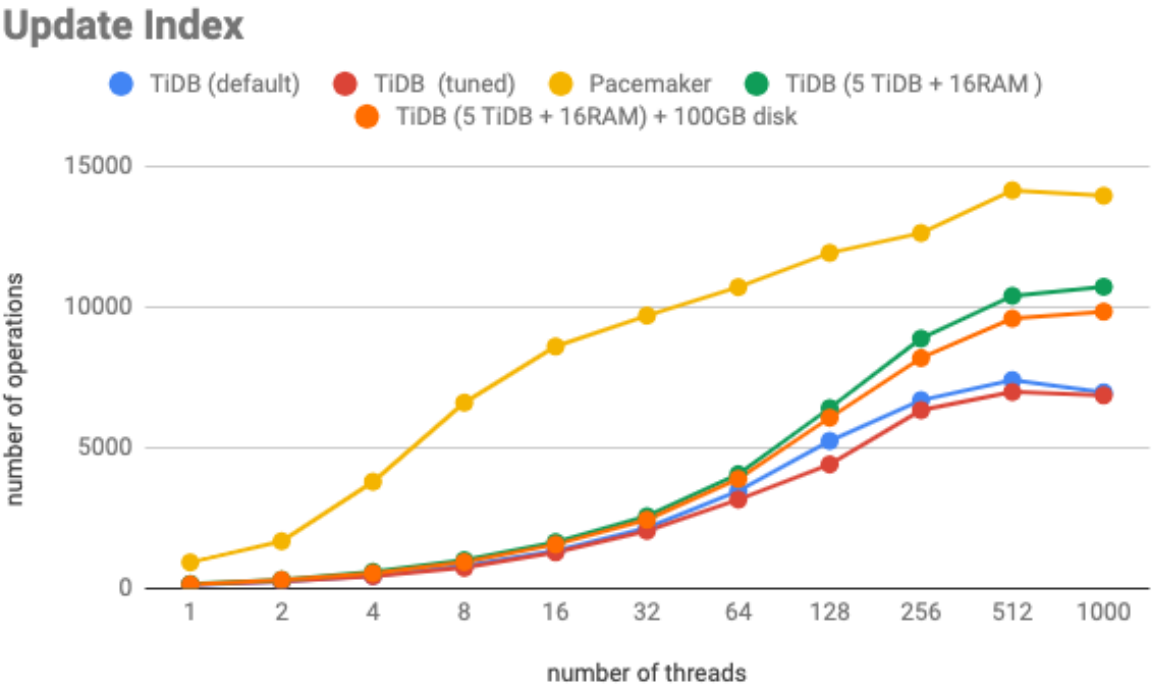

Figure 12. MySQL vs TiDB Update Index Performance Test Results.

The diagrams provided illustrate the performance tests conducted using Sysbench across various scenarios. Each diagram represents a specific test, with the horizontal axis representing the number of threads employed, and the vertical axis representing the number of operations executed. Notably, each thread or step ran for a consistent duration of 240

seconds. Higher counts of performed operations in the diagrams correspond to superior results, signifying enhanced performance and efficiency for the respective tests.

Performance tests were conducted on TiDB using four different setups, each with its own unique configuration. In addition to the provided information, it can be specified that each TiDB setup consisted of 5 nodes, with each node equipped with 8 CPUs and 32GB of RAM. Furthermore, the Pacemaker setup comprised 2 nodes, each configured with 8 CPUs and 32GB of RAM, and supplemented with ProxySQL and HAProxy for enhanced functionality and load balancing capabilities.

The details of each setup are as follows:

1. TiDB (default):
    - Default configuration comprising 2 PD, 2 TiDB, and 5 TiKV pods.
    - Database size approximately 2GB.
2. TiDB (tuned):
    - 2 PD, 2 TiDB, and 5 TiKV pods.
    - Database size approximately 2GB.

    Specific parameter modifications:

    * SET @@tidb_dml_batch_size=67108864
    * SET GLOBAL tidb_server_memory_limit = "32GB"
    * SET tidb_mem_quota_query = 8 << 30
3. TiDB (5 TiDB + 16 RAM):
    - 2 PD, 5 TiDB, and 5 TiKV pods.
    - Database size approximately 2GB.

    Specific parameter modifications:

    * SET @@tidb_dml_batch_size=67108864
    * SET GLOBAL tidb_server_memory_limit = "16GB"
    * SET tidb_mem_quota_query = 8 << 30
4. TiDB (5 TiDB + 16 RAM) + 100 GB disk:
    - 2 PD, 5 TiDB, and 5 TiKV pods.
    - Database size approximately 100GB.

    Specific parameter modifications:

    * SET @@tidb_dml_batch_size=67108864

* SET GLOBAL tidb_server_memory_limit = "16GB"
* SET tidb_mem_quota_query = 8 << 30

The test results showed that MySQL performed better than TiDB in several areas, including insert and update index operations, as well as some OLTP workloads. However, TiDB showed good scalability and stability without any significant performance spikes, indicating its potential for handling larger workloads. The select random ranges workload produced similar results for both MySQL and TiDB, while select random points had slightly better performance in MySQL. Select operations performed similarly on both platforms. The test results suggest that TiDB has room for improvement in some workloads.

It is important to note that the TiDB tests were not conducted under optimized conditions, and there may be room for improvement in performance. To address this, there is a plan to contact TiDB support and seek feedback and recommendations.

5.3 Further development

The feature implementation schedule has been set for the next two years as part of the project's further development plan.

Table 2. Further plan for the development of the new NewSQL platform for Pipedrive.

Task	Complexity	Time
Perform failover tests	Medium	2-3 days
Scaling and Benchmarking	Medium	1-2 months
Integration and Migration	Hard	2-3 years
Monitoring and alerting	Medium	4-5 weeks
Disaster Recovery	Medium	3-4 days

6. Summary

The aim of this thesis has been successfully achieved through the completion of several key tasks. Firstly, a comprehensive comparison and analysis of various NewSQL platforms was conducted to determine the most suitable option for implementation in Pipedrive's architecture. Subsequently, a proof of concept was developed and tested through the creation of a test cluster and performance testing. Throughout the process, potential issues were identified and mitigated at an early stage to ensure smooth implementation.

As a result of the project, the author has gained both theoretical and practical knowledge in the field of NewSQL platforms and their implementation. The findings of the thesis provide valuable insights for organizations seeking to adopt a NewSQL platform, and the proposed steps for further development will improve the scalability, availability, and reliability of the platform while ensuring the security and compliance of the data.

References

[1] Connolly, T. and Begg, C., (2014). Database Systems: A Practical Approach to Design, Implementation, and Management. [Online] Available at: https://books.google.ee/books?hl=en&lr=&id=jJbnDxiJ4joC&oi=fnd&pg=PR33&dq=what+are+the+different+types+of+DBMS&ots=4hvTtALuNf&sig=h1CwuSDU1bh8vd10jjS_IGPgXrs&redir_esc=y#v=onepage&q&f=false [Accessed 16 April 2023].

[2] Rodgers, K. and Howlett, D., (2003). What is CRM? [Online] Available at: https://www.gapconsulting.co.uk/Downloads/WhatisCRM.pdf [Accessed 16 April 2023].

[3] "Pipedrive - CRM Software for Salespeople," Pipedrive, [Online]. Available: https://www.pipedrive.com/en/about. [Accessed 16 April 2023].

[4] “What is a Relational Database (RDBMS)?” Oracle Inc, [Online]. Available: https://www.oracle.com/uk/database/what-is-a-relational-database . [Accessed 16 April 2023].

[5] “What are NoSQL databases?”, IBM, [Online]. Available: https://www.ibm.com/topics/nosql-databases . [Accessed 16 April 2023]

[6] MongoDB. "NoSQL vs. SQL." NoSQL Explained. MongoDB, n.d. Web, [Online]. Available: https://www.mongodb.com/nosql-explained/nosql-vs-sql#:~:text=and%20fewer%20bugs.-,What%20are%20the%20drawbacks%20of%20NoSQL%20databases%3F,durability)%20transactions%20across%20multiple%20documents.

[7] García-García, J. A., Palma, J., & Sánchez, L. (2018). Top NewSQL databases and features: classification, characterization, and evaluation. Journal of Industrial Information Integration, 10, 22-34, [Online]. Available: https://d1wqtxts1xzle7.cloudfront.net/56520056/10218ijdms02-libre.pdf?1525842725=&response-content-disposition=inline%3B+filename%3DTOP_NEWSQL_DATABASES_AND_FEATURES_CLASSI.pdf&Expires=1682407386&Signature=KViL36dltdPGQX9bMNz9cnn17f6nQZraThp~zepafI3QHzZ-BTEWuuTLGZ4K3izpaObmYl2GM0aYpGvmRijuHbzg0TWSLGB8qdZIaKMI5FnQOLFdzAnOeHWGVYZ1pKJi3RQYdtm9SptzTK9zcfQ6WbDJI7WbY-iQHxozrQimDcYb-W8bZ4QJkl4ZkAI2WVDBZFlYlOgjI6MQEaiRFB7At5SVVat8aJi5yencAyfRCxhAU6pKFHwhgNRZ0XME1bBZggOQAvV8y2-uAxvqmOAU6LP-oA8jRWu~OhQXNEN5EHyUB-j0SzLUJKkITlpu1YoviH~wwjM62O8tRjGnrN12sA__&Key-Pair-Id=APKAJLOHF5GGSLRBV4ZA [Accessed 16 April 2023].

[8] PingCAP. (n.d.). TiDB Overview [Online]. Available: https://docs.pingcap.com/tidb/stable/overview [Accessed 16 April 2023].
[9] PingCAP. (n.d.). TiDB, [Online]. Available: https://docs.pingcap.com/tidb/stable/tidb-architecture#consensus-algorithm. [Accessed 16 April 2023].
[10] PingCAP. (n.d.). TiDB, [Online]. Available: https://docs.pingcap.com/tidb/stable/backup-and-restore-overview#tidb-backup--restore-overview. [Accessed 16 April 2023].
[11] PingCAP. (n.d.). TiDB Monitoring Framework. TiDB Documentation, [Online]. Available: https://docs.pingcap.com/tidb/stable/tidb-monitoring-framework . [Accessed 16 April 2023].
[12] PingCAP. (n.d.). TiDB, [Online]. Available: https://docs.pingcap.com/tidb/stable/mysql-compatibility#mysql-compatibility. [Accessed 16 April 2023].
[13] PingCAP. (n.d.). TiDB, [Online]. Available: https://docs.pingcap.com/tidb/stable/manage-cluster-faq#what-is-the-recommended-number-of-replicas-in-the-tikv-cluster-is-it-better-to-keep-the-minimum-number-for-high-availability . [Accessed 16 April 2023].
[14] PingCAP. (n.d.). TiDB, [Online]. Available: https://docs.pingcap.com/. [Accessed 16 April 2023].
[15] Cockroach Labs. (n.d.). CockroachDB, [Online]. Available: https://www.cockroachlabs.com/docs/stable/cockroachdb-in-comparison.html. [Accessed 16 April 2023].
[16] Cockroach Labs. (n.d.). CockroachDB, [Online]. Available: https://www.cockroachlabs.com/docs/stable/multi-active-availability.html. [Accessed 16 April 2023].
[17] Cockroach Labs. (n.d.). CockroachDB, [Online]. Available: https://www.cockroachlabs.com/docs/stable/online-schema-changes.html. [Accessed 16 April 2023].
[18] Cockroach Labs. (n.d.). CockroachDB, [Online]. Available: https://www.cockroachlabs.com/docs/stable/backup-and-restore-overview.html. [Accessed 16 April 2023].
[19] Cockroach Labs. (n.d.). CockroachDB, [Online]. Available: https://www.cockroachlabs.com/docs/stable/monitor-cockroachdb-with-prometheus.html . [Accessed 16 April 2023].
[20] Cockroach Labs. (n.d.). CockroachDB, [Online]. Available: https://www.cockroachlabs.com/docs/stable/enterprise-licensing.html . [Accessed 16 April 2023].
[21] Cockroach Labs. (n.d.). CockroachDB, [Online]. Available: https://www.cockroachlabs.com/docs/stable/licensing-faqs.html . [Accessed 16 April 2023].
[22] Yugabyte, Inc. (2021). YugabyteDB Documentation, [Online]. Available: https://docs.yugabyte.com/preview/explore/linear-scalability/ . [Accessed 16 April 2023].
[23] Yugabyte, Inc. (2021). YugabyteDB Documentation, [Online]. Available: https://docs.yugabyte.com/preview/architecture/core-functions/high-availability/. [Accessed 16 April 2023].

[24] Yugabyte, Inc. (2021). YugabyteDB Documentation, [Online]. Available: https://docs.yugabyte.com/preview/manage/backup-restore/. [Accessed 16 April 2023].
[25] Yugabyte, Inc. (2021). YugabyteDB Documentation, [Online]. Available: https://docs.yugabyte.com/preview/explore/observability/. [Accessed 16 April 2023].
[26] Yugabyte, Inc. (2021). YugabyteDB Documentation, [Online]. Available: https://docs.yugabyte.com/preview/integrations/apache-kafka/. [Accessed 16 April 2023].
[27] Yugabyte, Inc. (2021). YugabyteDB Documentation, [Online]. Available: https://docs.yugabyte.com/preview/migrate/known-issues/mysql-oracle/. [Accessed 16 April 2023].
[28] Yugabyte, Inc. (2021). YugabyteDB Documentation, [Online]. Available: https://docs.yugabyte.com/preview/explore/ysql-language-features/sql-feature-support/. [Accessed 16 April 2023].
[29] Yugabyte, Inc. (2021). YugabyteDB Documentation, [Online]. Available: https://docs.yugabyte.com/preview/contribute/. [Accessed 16 April 2023].
[30] Yugabyte, Inc. (2021). YugabyteDB Documentation, [Online]. Available: https://www.yugabyte.com/wp-content/uploads/2023/03/yugabytedb-managed-product-brief.pdf. [Accessed 16 April 2023].
[31] Yugabyte, Inc. (2021). YugabyteDB Documentation, [Online]. Available: https://docs.yugabyte.com/preview/contribute/docs/. [Accessed 16 April 2023].
[32] Yugabyte, Inc. (2021). YugabyteDB Documentation, [Online]. Available: https://docs.yugabyte.com/. [Accessed 16 April 2023].
[33] IBM. (n.d.). CAP theorem, [Online]. Available: https://www.ibm.com/topics/cap-theorem#:~:text=CP%20database%3A%20A%20CP%20database,until%20the%20partition%20is%20resolved. [Accessed 16 April 2023].
[34] Apache Software Foundation. (n.d.). Apache Kafka, [Online]. Available: https://kafka.apache.org/intro. [Accessed 16 April 2023].
[35] Striim. (n.d.). Change data capture (CDC): What it is and how it works, [Online]. Available: https://www.striim.com/blog/change-data-capture-cdc-what-it-is-and-how-it-works/. [Accessed 16 April 2023].
[36] Amazon Web Services, Inc. (n.d.). Serverless computing, [Online]. Available: https://aws.amazon.com/serverless/. [Accessed 16 April 2023].
[37] Kubernetes. (n.d.). Nodes, [Online]. Available: Retrieved April 16, 2023, from https://kubernetes.io/docs/concepts/architecture/nodes/.
[38] Helm. (n.d.). Helm - The Kubernetes Package Manager, [Online]. Available: https://helm.sh/. [Accessed 16 April 2023].
[39] Hostinger International, Ltd. (n.d.). What is MySQL?, [Online]. Available: https://www.hostinger.com/tutorials/what-is-mysql. [Accessed 16 April 2023].

[40] Oracle Corporation. (n.d.). MySQL InnoDB Cluster, [Online]. Available: https://dev.mysql.com/doc/mysql-shell/8.0/en/mysql-innodb-cluster.html. [Accessed 16 April 2023].
[41] Kubernetes. (n.d.). Local persistent volumes, [Online]. Available: https://kubernetes.io/blog/2019/04/04/kubernetes-1.14-local-persistent-volumes-ga/. [Accessed 16 April 2023].
[42] Stepan Davidovic. (2019, September 19). To Helm or not to Helm, [Online]. Available: https://stepan.wtf/to-helm-or-not/. [Accessed 16 April 2023].
[43] HashiCorp. (n.d.). Introduction to Terraform, [Online]. Available: https://developer.hashicorp.com/terraform/intro. [Accessed 8 May 2023].
[44] Ansible. (n.d.). How Ansible Works, [Online]. Available: https://www.ansible.com/overview/how-ansible-works. [Accessed 8 May 2023].
[45] DigitalOcean. (n.d.). How to Use ProxySQL as a Load Balancer for MySQL on Ubuntu 16.04, [Online]. Available: https://www.digitalocean.com/community/tutorials/how-to-use-proxysql-as-a-load-balancer-for-mysql-on-ubuntu-16-04. [Accessed 12 May 2023].
[46] HAProxy. (n.d.). HAProxy, [Online]. Available: https://www.haproxy.org/. [Accessed 12 May 2023].

Appendix 1 – Non-Exclusive License for Reproduction and Publication of a Graduation Thesis[1]

I Daniil Mandrikov

1. Grant Tallinn University of Technology a free non-exclusive license for my thesis entitled "Selection and Implementation of a NewSQL Platform for Pipedrive", supervised by Juri Hudolejev:
   1.1. to be reproduced and published for preservation and electronic publication of the graduation thesis.
   1.2. to include the thesis in the digital collection of the library of Tallinn. University of Technology until the expiration of the term of copyright and publishing it via the web of Tallinn University of Technology.
2. I understand that I also retain the rights specified in clause 1 of the non-exclusive license.
3. I confirm that granting this non-exclusive license does not infringe on other persons' intellectual property rights, the rights arising from the Personal Data Protection Act, or rights arising from other legislation.

[1] 1 The non-exclusive licence is not valid during the validity of access restriction indicated in the student's application for restriction on access to the graduation thesis that has been signed by the school's dean, except in case of the university's right to reproduce the thesis for preservation purposes only. If a graduation thesis is based on the joint creative activity of two or more persons and the co-author(s) has/have not granted, by the set deadline, the student defending his/her graduation thesis consent to reproduce and publish the graduation thesis in compliance with clauses 1.1 and 1.2 of the non-exclusive licence, the non-exclusive license shall not be valid for the period.