POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

Adoption of ZKP in blockchain-based collaborative business intelligence for KPI values sharing

Tesi di Laurea Magistrale in
Computer Science and Engineering - Ingegneria Informatica

Author: **Alessandra de Stefano**

Student ID: 996017
Advisor: Prof. Pierluigi Plebani
Co-advisors: Giovanni Ennio Quattrocchi
Academic Year: 2023-24

Abstract

This thesis explores the development of a novel framework for enhancing Collaborative Business Intelligence (CBI) through the use of Zero-Knowledge Proofs (ZKPs) and blockchain technology. The proposed framework addresses the critical challenge of balancing data sovereignty with data trustworthiness, allowing organizations to share Key Performance Indicators (KPIs) without revealing sensitive underlying data. By leveraging ZKPs, organizations can verify data authenticity without exposing the raw data itself, thus preserving privacy and confidentiality. Blockchain technology is integrated to provide a tamper-proof, immutable record of data-sharing processes, further enhancing trust between parties. The framework integrates carefully selected cryptographic techniques and includes a custom-coded library that supports OLAP operations and is scalable for Zero-Knowledge Proofs (ZKPs). Extensive performance testing, focused on time efficiency and memory optimization as dataset size increased, demonstrated that the solution enables secure and reliable data exchanges while preserving privacy. The results further highlight the framework's scalability, ensuring its capability to handle growing data volumes without compromising security. This thesis presents a practical and efficient solution for collaborative environments, where the need for verifiable insights must be balanced with the protection of sensitive data.

Keywords: Collaborative Business Intelligence, Zero-Knowledge Proofs, Blockchain, Data Sovereignty, Data Trustworthiness.

Abstract in lingua italiana

Questa tesi presenta lo sviluppo di un framework innovativo per potenziare la Collaborative Business Intelligence (CBI) tramite l'uso delle Zero-Knowledge Proofs (ZKP) e della tecnologia blockchain. Il framework affronta la sfida centrale di bilanciare la sovranità dei dati con la loro affidabilità, consentendo alle organizzazioni di condividere Key Performance Indicators (KPI) senza rivelare informazioni sensibili. Grazie alle ZKP, è possibile verificare l'autenticità dei dati senza dover esporre quelli grezzi, garantendo così elevati livelli di privacy e riservatezza. La blockchain, invece, viene utilizzata per creare un registro immutabile e sicuro delle transazioni di condivisione dei dati, aumentando la fiducia tra le parti coinvolte. Il framework si avvale di tecniche crittografiche avanzate e integra una libreria personalizzata per supportare le operazioni OLAP, garantendo una soluzione scalabile nell'implementazione delle ZKP. I test eseguiti, incentrati su efficienza e ottimizzazione delle risorse con volumi di dati crescenti, hanno confermato la capacità della soluzione di facilitare scambi di dati sicuri e affidabili, mantenendo alti standard di riservatezza. Inoltre, i risultati dimostrano un'eccellente scalabilità del sistema, che gestisce grandi volumi di dati senza compromessi sulla sicurezza. Questa tesi propone una soluzione pratica ed efficace per ambienti collaborativi, dove è essenziale ottenere informazioni verificabili proteggendo al contempo i dati sensibili.

Parole chiave: Collaborative Business Intelligence, Zero-Knowledge Proofs, Blockchain, Sovranità dei Dati, Affidabilità dei Dati.

Contents

1 | Introduction

Data has become an asset for businesses, especially in the context of supply chain operations, where the ability to share data between organizations is crucial for optimizing decision-making and ensuring operational efficiency.

This has led to the rise of Collaborative Business Intelligence (CBI), which refers to the integration of business intelligence tools and practices across multiple organizations, enabling them to work together and share insights. The concept emerged in the early 2000s as part of the evolution of business intelligence tools and practices. It was developed in response to the increasing need for integrated and cooperative data analysis and organizational decision-making approaches. CBI plays a crucial role in pinpointing Key Performance Indicators (KPIs) that can make a significant difference in leading enhanced decision-making processes. As businesses increasingly recognized the value of sharing insights and collaborating on data-driven decisions, the term began to be used to describe this new approach that combined traditional BI tools with collaboration technologies. This new importance that CBI has been given has also raised a dichotomy between data sovereignty and data trustworthiness. Businesses want reliable data so that they can make informed decisions. Consequently, they want to live up to the expectations they have, so they try to provide accurate data. The thing is that businesses also want and need to protect their most sensitive data. Even though they desire to share precise data, they must also protect the inner information, which would go against them. Keeping control, confidentiality, and regulatory compliance is a real challenge since trying to be more transparent would inadvertently put at risk privacy, security, and competitive advantage.

If we analyze both trivial cases, it becomes immediately clear why there is the urge to find a trade-off between trust and transparency. Let's suppose that an organization wants to share with another some information. To choose data trustworthiness means that the organization shares the KPI without revealing any other data, and the recipient has the only choice but to trust the provider. Meanwhile, if the sender chooses complete transparency, it shares its entire raw data, compromising its privacy. These trivial cases highlight the reason why studying new solutions in search of balance.

In a CBI ecosystem, data sharing often occurs within the framework of data warehouses, where vast datasets are stored, processed, and analyzed to extract valuable insights. In this context, Online Analytical Processing (OLAP) operations are important as they allow organizations to analyze data and compute KPIs. These performance metrics, in turn, serve for inter-organizational data sharing, offering insights into decision-making without disclosing the underlying raw data. Still, ensuring the security and privacy of these OLAP-derived results while maintaining data integrity presents significant obstacles for businesses engaged in CBI initiatives. This thesis examines Collaborative Business Intelligence (CBI) with a focus on secure data sharing through the use of Data Spaces and Data Lakes, along with advanced cryptographic techniques like Zero-Knowledge Proofs (ZKPs) and blockchain technology. We will explore the architecture that enables such systems to ensure data sovereignty and trustworthiness.

1.1. Motivation and Research Objectives

The motivation behind researching and creating a new framework to increase collaboration among businesses is rooted in the benefits that collaborative business intelligence (CBI) can bring to organizations. In an increasingly interconnected and competitive global marketplace, companies continuously seek ways to optimize their decision-making processes, enhance efficiency, and gain a competitive edge. By facilitating more effective collaboration and data sharing between organizations, CBI offers a pathway to achieving these goals. Key benefits include enhanced decision-making, efficiency, innovation, and a significant competitive advantage.

This thesis finds its place in this area, trying to find a solution that maintains a certain balance between data sovereignty and data trustworthiness, enhancing businesses' ability to share more information. Given this objective, it is important to highlight that the proposed framework is extremely suitable to operate within data spaces—trusted ecosystems where multiple organizations can share data under strict control while maintaining sovereignty and compliance with regulations. These spaces aim to create environments where data can be shared securely while ensuring that participants keep control over their data and comply with legal requirements.

This work starts from the assumption that organizations use Data-Fact Models to organize their data for analytics and to compute the KPIs. Thus, the goal is to exploit Zero-Knowledge Proofs (ZK-Proofs) as a solution to achieve the balance mentioned above.

This framework is particularly well-suited for operation within data spaces, which are trusted environments that allow multiple organizations to share data under strict gover-

nance. These spaces ensure that data sovereignty is maintained, and organizations can share information while adhering to regulatory requirements related to data privacy and security. By integrating Data-Fact Models, ZKPs, and blockchain technology, this framework provides a scalable and secure solution for organizations to share KPIs derived from OLAP operations, all while ensuring the confidentiality and integrity of the underlying data.

ZK-Proofs are a type of cryptographic technique that lets one party prove to another that something is true without revealing any of its details. By using ZK-Proofs, a company can share the results of their KPIs with partners or other organizations without giving away the underlying data that generated those results. This means they keep their sensitive information safe while still being transparent.

The main advantage of this approach is that the company sharing the KPIs remains in complete control of its internal data - as requested by data sovereignty; thus, it does not have to worry about exposing too much, while the other party can trust that the shared indicators are accurate and reliable because they've been validated through a secure process.

In addition, the approach proposed in this work introduces blockchain technology as a further layer of trust, as data managed by it is virtually impossible to tamper with. By recording the shared KPIs on a blockchain, both parties can be confident that the data is secure and has a permanent, unchangeable record. This adds transparency and trust, making it easier for organizations to collaborate without concerns about data integrity.

In essence, the goal of this work is to combine Data-Fact Models, ZK-Proofs, and blockchain to allow organizations to share critical performance metrics in a secure and trustworthy way.

Lastly, the importance of regulatory compliance and security must be balanced. With the increasing emphasis on data privacy and regulatory compliance, businesses must ensure that any collaborative efforts do not compromise security or violate regulations. The proposed framework, focusing on ZK-Proofs and blockchain, provides a secure way to share information, thus helping organizations adhere to legal requirements while fostering collaboration. Implementing this framework empowers businesses to collaborate more effectively, protect their sensitive information, and confidently navigate the complexities of a competitive and data-driven world.

1.2. Thesis Structure

With this background, the following chapters are structured to explore these themes in detail:

- In **chapter 2**, we cover the essential concepts necessary for understanding this research. We begin with an overview of data spaces, data lakes, then continue with a study on Zero-Knowledge Proofs. The chapter continues with related technologies such as verifiable credentials, smart contracts, blockchain, and OLAP operations. Here lays the foundational knowledge required to grasp the innovations proposed later in the thesis and presents a review of the work currently present in the literature related to our approach.
- In **chapter 3**, we present the problem statement and explore the role of data in the modern world, the importance of data governance, and the critical need for secure data sharing. We analyze data sharing and data exchange, highlighting their similarities and differences after delving into the concept of Collaborative Business Intelligence, one of the core concepts of the thesis. Finally, we introduce the ZKP-based framework.
- In **chapter 4**, we introduce and explain the proposed solution, detailing each step of the framework, from the initial scenario setup to the integration of advanced technologies like Zero-Knowledge Proofs and blockchain for secure data sharing, outlining how these elements combine to enable secure data sharing in CBI.
- In **chapter 5**, we provide the details of the implementation of the proposed solution. The first section focuses on the practical aspects, discussing how the framework was applied and the challenges encountered during the process. The second section includes an analysis of the effectiveness of the solution to highlight its advantages and potential areas for improvement.
- Finally, **chapter 6** offers the conclusions of the research and suggests directions for future work.

2 | Background and Related Work

In this chapter, we will be focusing on the key concepts strictly related to the understanding of this research. So we will define zero-knowledge proof protocol, verifiable credentials, smart contracts, blockchain, data lakes, data spaces, and OLAP operations.

The chapter also contains a review of the work currently present in the literature that which has similar or related approaches to our.

2.1. Data Space

A Data Space refers to a trusted ecosystem where organizations can share and exchange data under governed frameworks. These spaces are pivotal to ensuring that data sovereignty is maintained, setting the stage for secure collaboration between entities.

The term Data Space was first introduced in 2005 by Franklin, Halvey, and Maier in 2005, and over the years, the definition of data space has never stopped evolving. The main concept is that a data space should be a sort of container of all the information linked to an organization independently of its format and location. Moreover, it should maintain a reference to all the relationships between data repositories [11]. The reader should not think of a data space as a new approach that aims to integrate data, but rather as a method that allows data to coexist no matter the way it is structured. The most valuable aspect of data spaces is that they allow for the reduction of a priori manual processing of data before sharing. Since data is always increasing, thinking of rendering all data always compatible upfront, and unifying schema across all sources is becoming more and more unthinkable. That is why data spaces act like a container that holds on to every bit of information and relies on automatic matching and mapping generation techniques which results in heterogeneous sources managed by the same actor [10].

More recently data spaces have been linked to contemporary concerns about data governance and control within distributed systems, which will be analyzed in the next chapter [38].

The way we understand and manage different types of data has developed significantly

over time. Initially, the main concern was how to link and work with data from various sources effectively. As technology advanced and regulations increased, the focus shifted to more comprehensive data management, including how data is governed and controlled across various systems and organizations. Each author who has contributed to the understanding of data spaces has modified the basic concept to better suit the modern needs for technology and compliance. This shows how dataspaces have grown from simple setups for handling data to complex systems designed to manage the intricate details of today's vast digital data networks.

Data spaces are not merely static repositories but dynamic environments that facilitate incremental data integration and enhancement over time. This approach allows organizations to initially use data without heavy preprocessing or integration, enabling quick access and utilization. As the needs for more precise data integration or refined data analysis emerge, data spaces can incrementally integrate and enhance data through sophisticated matching and mapping techniques. This "pay-as-you-go" model supports the evolutionary growth of data management practices within organizations, adapting to increasing data volumes and integration requirements without the overwhelming initial burden of unifying diverse data sources. This flexible approach is crucial for managing the expanding landscape of data types and sources, ensuring that data spaces remain scalable and adaptable to future technological and business developments.

This concept echoes the evolution and adaptive nature of data spaces, highlighting their capability to grow and evolve along with organizational needs and the broader data environment. Such an approach ensures that data spaces are not only repositories of data but active, evolving frameworks that support the progressive integration and enhancement of data as demanded by users and applications.

Data governance plays a crucial role in both data lakes and data spaces, ensuring that data is managed and utilized in compliance with legal and regulatory frameworks. Effective data governance establishes policies for data quality, data stewardship, and data lifecycle management. Frameworks like the General Data Protection Regulation (GDPR) in Europe or the California Consumer Privacy Act (CCPA) in the United States underscore the importance of having robust data governance in place. These regulations not only mandate how data should be shared but also enforce strict guidelines on data privacy, which are essential when considering cross-border data sharing initiatives.

Data spaces are supported by underlying infrastructures that facilitate basic functionalities like search, query, and administration across diverse data sources. These functionalities are provided regardless of how integrated the data sources are. Can we use data

lakes? Yes, a data lake can form a crucial part of the infrastructure for a data space by providing core capabilities such as data storage, retrieval, and preliminary analysis.

2.2. Data Lake

A data lake is a storage repository designed to grasp and store a large amount of all types of raw data, ranging from binary data and text to images. It comes from multiple sources, which flow into the same combined site. They support both structured and unstructured data and facilitate the generation of Key Performance Indicators through processes like OLAP operations. This process allows you to scale to data of any size while saving time in defining data structures, schema, and transformations. Once it is in the data lake, it can be used in machine learning or artificial intelligence (AI) algorithms and models for business purposes. Organizations can use data lakes to gain insights from historical data and create machine learning models. These models can forecast likely outcomes and recommend actions to achieve the best results.

For some companies, a data lake works best, especially those that benefit from raw data for machine learning.

This storage flexibility means that data does not need to be converted or processed before it is stored, which allows organizations to scale their data storage rapidly and cost-effectively. Data lakes support a variety of analytics, including SQL queries, big data processing, machine learning, and real-time analytics, providing the essential raw data necessary for deep analysis and advanced algorithmic processing. With technologies like Hadoop, NoSQL, and cloud-based storage solutions, data lakes can manage massive volumes of data and scale in terms of both storage capacity and processing power. Accessibility is another key feature, as users from different business units within an organization can access the data lake for insights, fostering innovation and improvements across the organization. However, it is crucial to implement robust data governance and security strategies to ensure that the data remains accessible while being protected from unauthorized access.

You have the flexibility to store your data as-is, without the need to structure it first. This allows you to perform a variety of analyses, including creating dashboards, visualizations, big data processing, real-time analytics, and machine learning, all of which can help in making well-informed decisions.

Data lakes can be utilized across various sectors by data professionals to address and solve business problems. Some examples include marketing, education, and transportation.

Marketing professionals can collect data on their target customer demographic's preferences from many different sources in a data lake. Data lakes enable marketers to analyze data, make strategic decisions, and build data-driven campaigns [39].

The education sector has begun using data lakes to track data on grades, attendance, and other performance metrics so that universities and schools can improve their fundraising and policy goals. A data lake provides the right amount of flexibility to handle these types of data.

A data lake is used when data scientists of airline and cargo companies cut costs and increase efficiency to support lean supply chain management.

In addition, data lakes allow various roles in an organization like data scientists, data developers, and business analysts to access data with their choice of analytic tools and frameworks. They allow to run analytics without the need to move data to a separate analytics system. The capacity to leverage greater volumes of data from diverse sources in shorter timeframes, while enabling users to collaborate and analyze data through various methods, results in more effective and expedited decision-making.

The main challenge with a data lake architecture is that raw data is stored with no oversight of the contents. To make data usable, it needs to have defined mechanisms to catalog, and secure data. Without these elements, data cannot be found, or trusted resulting in a "data swamp." Addressing the needs of broader audiences requires that data lakes incorporate governance, semantic consistency, and access controls.

This discussion adopts a specific perspective to clarify the roles of data lakes and data spaces for this research. While this approach facilitates our exploration and analysis, the reader should keep in mind that it represents just one of many possible ways to understand and integrate these concepts, and it aims to ease the reading and understanding of this thesis. Technically, we discuss data lakes as the foundational technology infrastructure that stores and manages large volumes of diverse data. From a managerial perspective, we examine data spaces as the framework within which data management and governance occur. For this study, we conceptualize the data space as operating within a data lake; the data lake provides the necessary technological support, making it the means to achieve the managerial goals set out in the data space.

In summary, data lakes and data spaces, while distinct, can be seamlessly integrated into a comprehensive data strategy. The data lake addresses the technological requirements of data storage and processing, while the data space provides the managerial framework for governance, sharing, and utilization. This integration allows organizations to maximize

the potential of their data, leveraging both cutting-edge technology and robust management practices to create a secure, efficient, and collaborative data ecosystem.

2.3. Zero Knowledge Proofs

Zero Knowledge Proof, also known as ZKP, is an interactive protocol where a party called the prover has the ability to convince the other party, called the verifier, that some claim is true without actually revealing any information regarding the assertion, except for its veracity [19].

Before technically analyzing ZKP, it seems useful to give the reader an example to clarify what we are going to focus on. Imagine your friend Victor is red-green color-blind while you are not. You have two identical-looking balls: one red and one green. To Victor, these balls appear identical. He doubts that the balls have different colors, and you aim to demonstrate to Victor that the balls are indeed differently colored without revealing which one is red and which one is green [6].

So you hand the two balls to Victor, and he hides them behind his back. He then brings one ball out to show you, places it behind his back, and randomly chooses whether to switch them. He then asks you, "Did I switch the balls?" This sequence is repeated a great number of times.

Naturally, you can quickly tell if he switched the balls by their colors. If the balls were indistinguishable, guessing correctly would only happen by chance, with a probability of 50%. After repeating this test several times, Victor should be convinced that the balls are differently colored without gaining information about distinguishing the balls or which color belongs to which ball. This is why this method is considered zero-knowledge: the test convince Victor without giving him new information about the color of the balls.

The emergence of Zero-Knowledge proofs (ZKPs) arose from the following question: "Is there an alternative approach, beyond nondeterministic polynomial (NP) proofs, to establish the validity of theorems?" For instance, can a prover convince a verifier that y is a quadratic residue mod N without disclosing the square root? [14]

In computational complexity theory, proofs are classified as nondeterministic polynomial (NP) if the solution to a problem can be verified in polynomial time. The term "nondeterministic" means that there is no predetermined rule controlling the guessing process. In addition, a problem is said to be NP-complete if it belongs to NP, and all other NP problems can be reduced to it in polynomial time. Given that any problem within this class can be transformed into any other member of the class, discovering an efficient algo-

rithm for any NP-complete problem implies the existence of an efficient algorithm for all NP problems. However, the question of whether NP-complete problems are tractable or intractable remains one of the most significant unresolved issues in theoretical computer science. The possibility of finding polynomial-time algorithms for NP-complete problems remains uncertain. One approach to these problems is approximating the solution using polynomial algorithms. While the resulting solution may only sometimes be optimal, it is often sufficiently close.

Here is where ZKP comes in handy. One of the main achievements associated with ZKPs is their capacity to defend privacy. The ability to exchange data while providing proof of validity without disclosing sensitive information is crucial in a data-centric world where privacy is of utmost importance.

Personally identifiable information (PII) can be exploited to steal someone's identity. Assume you have to authenticate yourself in order to finish a transaction, and you must use some evidence to ensure you are who you say you are (driver's license, identity card). Once this information is in the hands of a third party, you will have no control over its security. The central database of the third party may be the target of hackers, or the third party may disclose your information to other parties for purposes such as targeted advertising. With zero-knowledge proofs, you can avoid sharing this information and stick to proving its validity [8].

Two main criteria must be satisfied when talking about ZKPs: soundness and completeness. Soundness refers to the fact that a protocol cannot verify incorrect input as accurate. In other words, if the prover attempts to persuade the verifier of a false assertion, the protocol should reject the evidence. Soundness guarantees that dishonest provers cannot fool verifiers. Achieving this criterion frequently requires cryptographic measures to ensure that the prover cannot influence the verifier or present incorrect information convincingly.

Completeness, on the other hand, guarantees that if the input is correct, the protocol will always verify the statement. This means that if both parties are honest and the underlying information is correct, the protocol should accept the evidence.

If soundness complies, it assures the integrity and accuracy of the protocol's outcome. Completeness ensures that valid proofs are always accepted.

To make these concepts more straightforward, let's refer back to the color-blind example:

- Completeness: If the balls are actually different colors, the prover can always convince the verifier that they are different by correctly guessing if the balls have been switched or not. This is because you have access to truthful information (the actual

colors), so your proof will always be complete.

- Soundness: If the balls were the same color, the verifier would have no way of telling wether the balls have been switched. There would not be factual information to distinguish the balls, and any correct answer would be purely by chance. Iterating the test, the incorrect answers would make the prover realize that the claim might be false, demonstrating the soundness of the protocol.

To recap, completeness ensures that valid proofs are accepted, while soundness ensures that invalid proofs are rejected. The example with Victor and the colored balls illustrates these concepts: valid input leads to consistent and correct proofs, while invalid input leads to exposing incorrect answers.

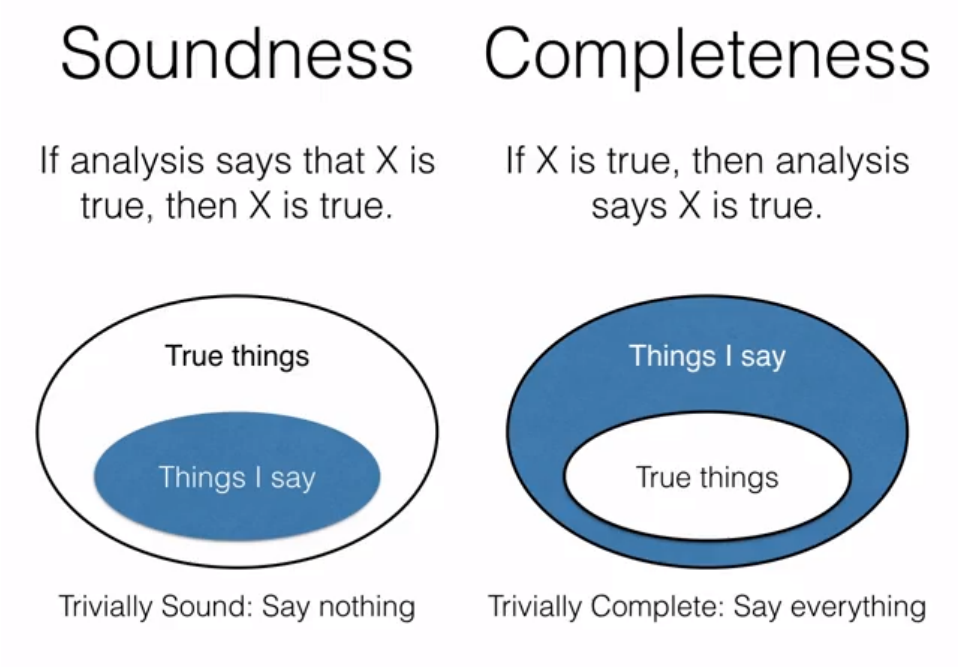


Figure 2.1: Soundness and Completeness [9]

We distinguish two types of ZKPs: interactive and non-interactive. The former allows the two parties to communicate back and forth. In this scenario, the prover answers the queries the verifier makes, providing proof for what has been asked to be validated. This can either happen in person or via the Internet. On the other hand, the latter does not involve this kind of active communication. The prover creates proof that the verifier can choose to exploit when it most sees fit. This means that once the proof is received, there is no need for more communication between the two participants. It seems trivial to highlight that non-interactive ZKP is more efficient since it does not require both the prover and the verifier to be present in the same place or online simultaneously. For this

reason, the majority of the protocols exploit non-interactive proofs like zk-SNARKs, stars, bulletproof, and plonk. This is why this research will be centered on this second category.

In these communications being able to trust the identity of the entity one is communicating with is a crucial point, and it can be accomplished with verifiable credentials.

2.4. Verifiable Credentials

A verifiable credential is a modern, efficient, and tamper-evident way of representing physical credentials [57]. They can be used, as we are usually accustomed to using physical ones, to prove something. They become extremely useful when there is the need to use some sort of credentials to ensure trust between two entities at a distance. The term verifiable does not mean that it is certain that what is wrapped up with verifiable credentials is true, but rather that it is possible for a verifier to check whether the information is true. Once the authenticity has been certified, the verifier is able to use its means to either assert the truth or falsehood of the credentials.

Verifiable credentials offer a modern and highly efficient method for digitally representing physical credentials, providing a reliable solution for credential verification. Similar to their physical counterparts, these credentials are essential for proving claims or assertions, particularly in scenarios where establishing trust between remote entities is key. There are several roles that make this mechanism work: holder, issuer, subject, verifier, and verifiable data registry.

The **holder** generally has one or more verifiable credentials and can realize verifiable presentation from them, in order to be verified. The **issuer** is the entity that creates verifiable credentials based on one or more claims. The issuer transmits the credentials to the holder. The **subject** is the topic of the claims; sometimes, the holder is the subject, but the subject may be only related to the holder. The **verifier** has the job of checking the veracity of the credentials. Finally, there is a **verifiable data registry** that maintains identifiers, schemas, and data relevant to the verifiable credentials it contains.

It is crucial to understand that the term "verifiable" does not guarantee absolute certainty regarding the validity of the information contained within the credentials. Instead, it signifies the ability of a verifier to determine the accuracy of the information. Once the authenticity of the credentials is verified, the verifier can then confirm or challenge the claims made within.

In essence, the ecosystem surrounding verifiable credentials operates on principles of transparency, accountability, and trust, enabling secure and reliable verification of digital cre-

dentials across a range of contexts.

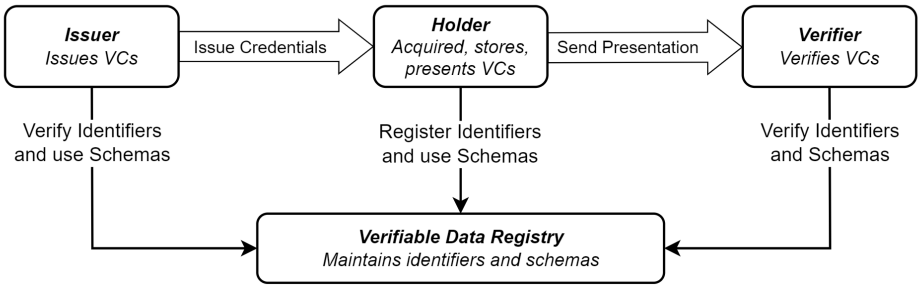


Figure 2.2: Verifiable credential schema [56]

An example to explain the roles would be a verifiable credential that represents a degree.

- **holder:** the student who graduates.
- **issuer:** the university the holder graduates from
- **subject:** the academic achievement
- **verifier:** an employer that wants to check the validity of the degree.

This case highlights that a digitally verifiable credential is not necessarily an ID or a substitute for an identity card or passport. It can be some property owned by someone, a study title, some sort of achievement, or anything that can be proved right.

2.5. Blockchain

Initially designed for cryptocurrency exchanges, blockchain technology has expanded to various domains where cooperation between untrusted parties is essential. Its key features of immutability and persistence make blockchain a popular choice for creating tamper-proof registries for documents and other digital assets. These capabilities make blockchain a promising technology for trusted process monitoring and identifying agreement violations without relying on trusted third parties [5].

Blockchain technology, as defined by IBM, is a distributed ledger that offers enhanced security, transparency, and immutability by allowing multiple parties to co-manage a reliable database without the need for a central authority [25]. This decentralized system includes blocks that record transactions, linked chronologically through cryptographic hashes, making alterations nearly impossible without consensus across the network [53].

The consensus mechanisms, such as proof-of-work or proof-of-stake, ensure that all network participants agree on the validity of transactions, reinforcing the integrity of the entire ledger [27].

One of the most innovative applications of blockchain is in supply chain management. Companies like IBM adopt blockchain to create transparent and immutable records for shipping containers, which enhances security, reduces losses, and prevents fraud by providing a single source of truth accessible to all participants in the supply chain [25].

Blockchain's capability extends to smart contracts, which automate contract enforcement when predefined conditions are met, thus minimizing the need for intermediaries and speeding up transactions. This technology is widely used in insurance claims, property transactions, and decentralized finance (DeFi) platforms that support activities like lending and trading without traditional financial institutions [25, 34].

Furthermore, blockchain is crucial in digital identity management, offering a secure and efficient method for verifying identities that could potentially streamline various digital interactions and enhance privacy [53].

2.6. Smart Contracts

Here we want here to give some details on smart contracts which represents an element of some blockchain platforms that has a key role in our proposal. These innovative constructs lie at the heart of blockchain technology, offering a revolutionary approach to executing agreements. Unlike traditional contracts that rely on human oversight, smart contracts are self-executing scripts programmed to perform specific tasks automatically when predetermined conditions are met. They are decentralized and can be anchored to the blockchain, allowing negotiating, carrying out, and enforcing the conditions of a legally binding agreement [48].

In easy words, smart contracts are the digital counterparts of traditional agreements, incorporating the terms and conditions of a legally binding contract in code. By leveraging the blockchain technology, these contracts facilitate negotiation, execution, and enforcement without the need for intermediaries or centralized authorities. This decentralized approach not only simplifies processes but also fosters transparency and trust among the parties involved.

In a conventional contractual setup, parties are bound by legal obligations governed by centralized entities. Smart contracts automate contract execution based on predefined conditions: this eliminates the need for intermediaries, reducing the likelihood of disputes

and lowering transaction costs.

It is important to stress that the integration of smart contracts with the blockchain technology enhances the reduction of transaction risks and costs, while enhancing efficiency. Blockchain ledgers provide a tamper-proof and transparent record of transactions. When paired with smart contracts, which execute only when conditions are met and are recorded on the blockchain, this synergy ensures a secure and auditable method of conducting business.

This is why they are considered to be a tool useful to reduce administration and service costs, especially when used within the blockchain. Since a smart contract is executed only if certain conditions are respected and the blockchain represents a trustworthy ledger that cannot be modified, these two concepts used together lead to a secure way.

The fusion of smart contracts and the blockchain technology presents a revolutionary solution for contract management and execution. By exploiting automation, decentralization, and cryptographic security, smart contracts not only improve efficiency but also redefine trust and accountability in contractual agreements. They represent a robust tool with the potential to change how various industries work and shape a more secure and efficient future for commerce and governance.

2.7. OLAP operations

In this work three OLAP operations were investigated so that they could be implemented: slicing, dicing, and roll-up. The reason behind this decision was due to the fact that the provider shares partially its data, still keeping a part of raw data a secret. This resulted in choosing three operations that allow to make sorts of cuts to the dataset. In technical words, the operations chosen were those that reduce the granularity of the data cube. Granularity refers to the level of detail in the data. Lower granularity means more detailed data (e.g., daily sales), while higher granularity suggests more aggregated, less detailed data (e.g., yearly sales). Below a brief explanation respectively of slice, dice, and roll-up operations.

2.7.1. Slice Operation

The Slice operation in OLAP allows you to create a new sub-cube by selecting a single dimension from the data cube. By fixing one dimension at a specific value, the Slice operation provides a focused view of the data, filtering out other dimensions focusing on a particular aspect of the dataset.

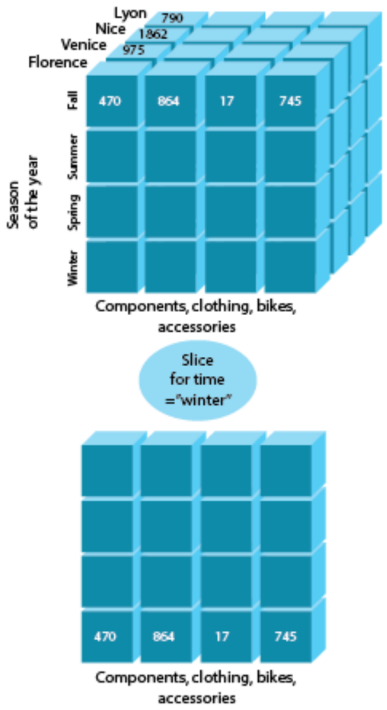


Figure 2.3: Slice Operation

Example: Initial Data Cube: The initial data cube shows the data across multiple locations, seasons, and product categories. Slice Action: The Slice operation filters the data by a specific value in the time dimension: Season = "Winter" Resulting Data Cube: The resulting sub-cube contains only the data for the Winter season across all locations and product categories. This allows analysts to focus on the winter sales performance alone.

2.7.2. Dice Operation

The Dice operation allows you to create a sub-cube by selecting specific values across two or more dimensions. Unlike Slice, which focuses on a single dimension, Dice involves filtering the data based on multiple dimensions, resulting in a more refined and specific sub-cube.

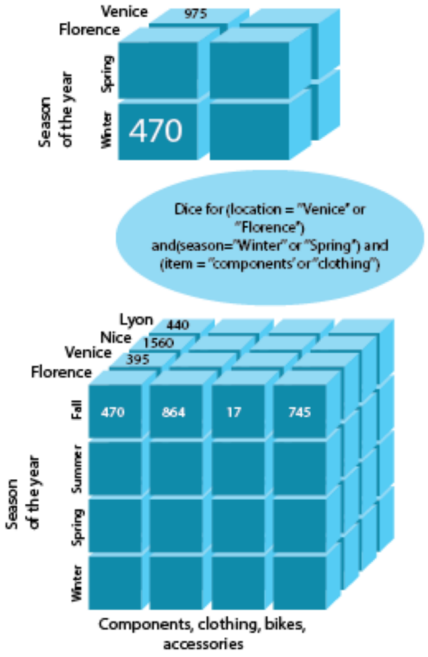


Figure 2.4: Dice Operation

Example: Initial Data Cube: The full cube includes data for several cities (Lyon, Nice, Venice, Florence), all seasons, and all product categories. Dice Action: The Dice operation selects data where: Location = "Venice" or "Florence" Season = "Winter" or "Spring" Product = "Components" or "Clothing" Resulting Data Cube: The resulting sub-cube now only includes data from Venice and Florence, for the Winter and Spring seasons, and only for components and clothing. This focused sub-cube allows a more detailed analysis of specific criteria.

2.7.3. Roll-up Operation

Definition: The Drill-Up operation, often referred to as Roll-Up, involves aggregating data by moving up a concept hierarchy or reducing the number of dimensions. This operation decreases the data's granularity, providing a broader, higher-level view of the information by summarizing it across a higher level of one or more dimensions.

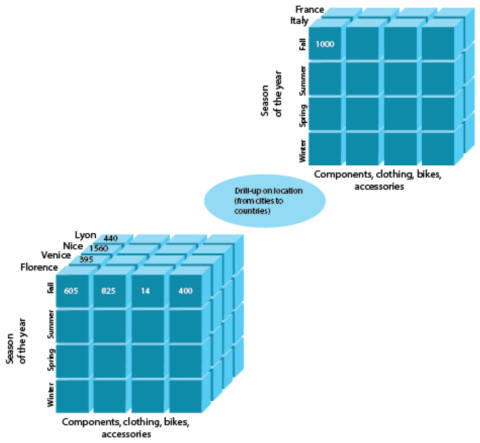


Figure 2.5: Roll-Up Operation

Example: Initial Data Cube: The initial cube contains data for different cities in France and Italy, split across four product categories (components, clothing, bikes, accessories) and four seasons (Winter, Spring, Summer, Fall). Roll-Up Action: The Drill-Up operation aggregates the data from the city level to the country level. This means that instead of looking at the data for individual cities, we are now looking at the combined data for each country. Resulting Data Cube: The new sub-cube shows the total values for France and Italy across the same product categories and seasons, providing a broader view.

2.8. Related Work

Nowadays, data is considered to be one of the most influential assets organizations can have, which results in harnessing data sharing. The increasing importance of data in the business context is also seen in the shift towards data as a core resource in business models [15, 24]. As highlighted in [28], the term "data sharing" does not always capture all the necessary aspects for its effective implementation.

First, it is important to understand the difference between 'data sharing' and 'data exchange,' especially if one focuses on raising awareness of the importance of data sharing. This intent entails what one can call BI 2.0, or in other words, Collaborative Business Intelligence (CBI) [55]. This can represent a new approach to push businesses to share their information to identify accurate KPIs (Key Performance Indicators). Since KPIs help understand what influences and makes a difference in terms of possible income, costs, and risks, they can be extremely important for an organization. Knowing this information from another business that has already done the 'research' can be a lifesaver. This

approach can lead to innovation across entire value chains or industry ecosystems, which consequently leads to better performance [21].

Several frameworks have been proposed to address the challenges of implementing CBI. Fahad and Darmont introduced an ontology-based CBI framework that enables organizations to connect, share, and visualize data collaboratively while respecting their autonomy. This framework leverages a collaborative ontology knowledge base to facilitate decision-making beyond enterprise boundaries, democratizing data access and enhancing collective analysis [17]. Similarly, cloud-based implementations of CBI have been developed to scale collaborative platforms across multiple organizations while maintaining data control and autonomy [3].

However, the implementation of CBI faces significant challenges: data trustworthiness and data sovereignty. Data trustworthiness, as defined by Saha and Srivastava [43], refers to the reliability and credibility of data. Data sovereignty, on the other hand, focuses on maintaining control over one's data, even when it is shared or processed by others. Weber et al. [51] argue that data sovereignty is crucial in collaborative environments to ensure compliance with regulations and protect competitive advantages.

Performance evaluations of existing CBI platforms have also revealed areas where improvement is needed. A systematic review conducted by Author et al. (2022) identifies key performance bottlenecks in collaborative platforms and emphasizes the need for more efficient OLAP processing to optimize business intelligence systems in collaborative settings [4]. These evaluations reinforce the importance of secure and efficient data sharing mechanisms to maximize CBI performance.

In the face of these challenges, we can look forward to the emergence of zero-knowledge proofs (ZKPs) as a potential solution. ZKPs, as introduced by Goldwasser et al. [20], provide a cryptographic method to confirm the veracity of a statement without revealing any additional information. The use of ZKPs can enable organizations to share insights derived from their data without exposing the underlying sensitive information [36]. If organizations are worried about revealing something they do not want to make public, they can 'cut' a portion of information, create a zero-knowledge proof, and guarantee the requester its information while safeguarding its secrets.

Blockchain technology offers a promising platform for implementing CBI with ZKPs. Its immutable ledger can enhance data trustworthiness by providing a tamper-proof record of data provenance and transformations [32]. As discussed by Zheng et al. [58], blockchain's inherent properties of decentralization, immutability, and transparency make it well-suited for secure and auditable data sharing. When combined with ZKPs, blockchain can facili-

tate trustworthy collaborative analytics while preserving data privacy [29].

As aforementioned, ZKP could represent the missing puzzle between secret keeping and data sharing, especially when implemented on a blockchain. It can enable data sovereignty by allowing organizations to share insights without losing control over their raw data [59]. By integrating Collaborative Business Intelligence with zero-knowledge proofs and blockchain technology, businesses can create a secure and trustworthy framework for data sharing. This innovative approach addresses long-standing privacy and security concerns, enabling companies to collaborate more effectively and extract new value from shared data, all while maintaining control over sensitive information.

3 | Problem Statement

In this chapter, we will first describe the motivation for applying ZKP to data sharing, and in the next chapter, we will thoroughly enumerate and explain the several steps required to do it, helping the reader understand the importance of this work-study. In order to understand this research, it is crucial to have in mind the reason why it has been decided to dig deeper towards this path. What was so vital about investigating in the field of the interchange of data? As we already mentioned, data can be worthy and dangerous, so protecting it has become fundamental.

Nowadays, the amount of data created each day is enormous. Approximately 328.77 million terabytes of data are made each day [16]. This quantity of information requires a new manner to manage data; to do so, the meaning of data must be evident. As data is the bare noun indicating a piece of information, it is necessary to distinguish among the different roles data can assume within enterprises. We can identify four: enabler, product, source of business, and strategic resource [38]. Let's analyze them.

First, it may seem trivial to assert that data is an *enabler*, but it is important to understand the depth of this sentence. To be able to perform integration and automation of a business, it is necessary to manage data resources efficiently. If correctly used, data can enable operational excellence. This leads us to the second role data can have: data as a product. It can be so useful that its value can be extremely high because it provides insights into user behavior, preferences, and trends. Companies collect data from users through various interactions such as online purchases, social media activity, and website visits. This data is then analyzed to optimize marketing strategies, improve product development, and enhance customer experiences. By leveraging data effectively, companies can increase revenue, reduce costs, and gain a competitive edge in the marketplace [45]. Hence, data is considered a valuable product because of its ability to drive business outcomes and inform decision-making processes. This is why it is considered a source of business innovation. And fourth, data is seen as a strategic resource, essential for the long-term sustainability of the economy.

As businesses navigate the complexities of the digital age, understanding and harnessing

the power of data has become fundamental. By recognizing data as an enabler, product, source of business innovation, and strategic resource, businesses can unlock new opportunities for growth, innovation, and long-term success in an increasingly data-driven world.

3.1. Collaborative Business Intelligence

Today, Business Intelligence (BI) systems are acknowledged as one of the most important technologies for assisting decision-making across all levels of management in organizations [31, 37]. There is still an urgent need for organizations to develop and implement a new generation of BI systems that can keep pace with the increasing importance, speed, and rapidly growing volume of data today. Existing BI solutions offer features that present results in various formats and summaries based on user needs.

Collaborative Business Intelligence can be seen as a new version of Business Intelligence. We may refer to it as BI 2.0 (where BI stands naturally for Business Intelligence). In fact, [55] describes the current environment where enterprises do business: exchanging emails and documents still seems to be the main way businesses act. Of course, this could result in losing information, if not worse, and consequently to a poorly advised decision-making process. In typical business intelligence (BI) environments, cooperative decision-making is frequently obstructed because stakeholders do not share their knowledge, expertise, or information effectively [52].

Collaborative Business Intelligence (CBI hereafter from now on) tries to overcome this lack of information sharing in order to improve the decision-making process via broadening collaboration, cooperation, and communication between the actors that participate in the interchange of data. It represents the integration of business intelligence tools across multiple organizations, enabling them to share insights without compromising data sovereignty. This approach relies on secure data exchanges that ensure the trustworthiness of shared insights while protecting proprietary information.Collaborative BI, where a company's information assets are empowered thanks to collaboration and data sharing with other companies and organizations, extends the decision-making process beyond the company boundaries [42]. [7] explains that CBI integrates business intelligence with collaboration technologies, thereby enhancing decision-making by combining the knowledge and expertise of multiple stakeholders. In other words, CBI is becoming increasingly a new approach for those enterprises that aim to make the most of their businesses. Nonetheless, this approach is barely supported by currently available tools. The process of collaborative business intelligence brings to light a fundamental tension between ensuring data trustworthiness and maintaining data sovereignty [55].

Chaudhuri and Dayal (1997) [7] state that OLAP provides a flexible and convenient method for navigating data through a multidimensional structure, which aids in information retrieval and the presentation of that information. Despite the extensive time and effort dedicated to Business Intelligence (BI) technologies in organizations, these systems frequently fail to impact managerial decision-making [1].

OLAP systems are designed to facilitate the analysis of large datasets by enabling operations like data aggregation, slicing, dicing, and pivoting across multiple dimensions, which are essential for calculating meaningful KPIs, playing a key role in this research. Sharing these KPIs within collaborative environments can be extremely effective, allowing organizations to gain valuable insights while maintaining control over their data.

KPI stands for Key Performance Indicator, and it refers to performance measuring over time for a specific objective [39]. This proves to be extremely useful in decision-making progress. This is why KPIs are considered to be valuable and something to be protected: they are a critical part of the achievement of informed decisions. Since they allow having an empiric measurement of how an enterprise is performing towards a certain objective and how its actions are contributing to its achievement, it is crystal clear how crucial KPIs are and also how, from a different point of view, having knowledge of a business's KPIs can either be extremely perilous (KPIs owner) or abundantly exploitable (secondary business). Having raised this concern, it seems trivial to remind the reader that this is the main reason why, even if data can be the most powerful and profitable thing businesses do have, it is also what they try to protect and not overshare. That is what prompted us to pour our energies into this field of research, which then led to the framework that will be presented and explained in the following chapters.

Given that the concept of data is clear, the next logical step is understanding the several ways developed during the years to share data. That is the main goal of the research: finding a way to share data, securely, being able to prove the veracity of information, keeping to a minimum close to 0 the probability of leaking any information we do not want to share. In the following section, we will be defining what data sharing means.

In literature, often the terms *data exchange* and *data sharing* are used as synonyms, even though they do not mean the same thing. For the clarity of the research, we need to distinguish between the two.

3.2. Data Exchange

Data exchange refers to the interchange of data after it has been modified in a technical way. In fact, "data exchange" is also used to express operations like "data translation" or "data integration," which are clearly actions taken upon data to obtain analogous information, yet altered [26]. In the paper previously mentioned, two main definitions of data exchange are provided:

"Data exchange, also known as data translation, can be succinctly described as the problem of transforming data structured under one schema, called the source schema, into data structured under a different schema, called the target schema [...]. Data exchange is typically formalized using schema mappings between the source schema and the target schema" [2].

"Data exchange is the problem of transforming data that is structured under a source schema into data structured under another schema, called the target schema, so that both the source and target data satisfy the relationship between the schema " [22].

The definitions of data exchange, as presented, span a decade, yet the core essence remains consistent. It is important to keep this in mind because it comprises one big difference between data exchange and data sharing. While ten years separate these definitions, both emphasize the main concept of data exchange: transforming data structured under one schema, the source, into another, the target. This underlines that data exchange has a well-defined meaning, with minor variations focusing on specific aspects.

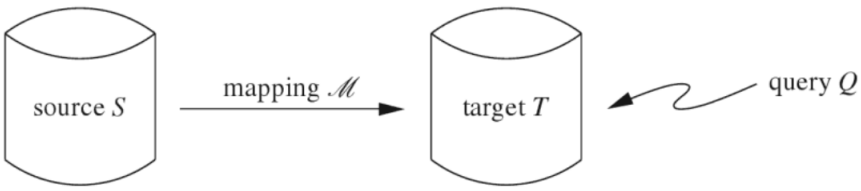


Figure 3.1: Data Exchange Example

The concept of data exchange goes beyond just changing data from one format to another. It is a detailed process that ensures data is not only transformed but also accurately represented in the receiving system. This involves several key steps:

- **Schema Matching and Mapping:** this step bridges the differences between the source and target schema. Specialized tools are used to find matching elements between the schema, ensuring the transformed data fits well into the target system [23].

- **Data Cleaning and Standardization:** here, the focus is on fixing errors and inconsistencies in the source data. Techniques like data cleaning algorithms and standardization ensure that the exchanged data is accurate and follows the required standards [54].
- **Data Validation:** this step checks if the transformed data meets the standards set by the target schema. It involves applying rules and checks to confirm that the data fits the target system's requirements.

Choosing the right data exchange method depends on factors like data size, complexity, how often the data is exchanged, and security needs. Some common methods include:

- **File Transfer Protocols (FTPs):** these are used for securely transferring large datasets between trusted parties.
- **Application Programming Interfaces (APIs):** APIs allow for automated data exchange between different applications, making integration smoother.
- **Cloud-based Integration Platforms:** these platforms help exchange data between cloud-based applications, offering flexibility and security.

A good example is Electronic Data Interchange (EDI), introduced in the 1960s [12]. Similarly to the majority of technological improvements, EDI was developed to ease processes in the military environment. In fact, EDI was created to enable electronic communication among the shipment supply chains, allowing instant long-distance communication before the Internet, reducing cost, increasing processing speed, and reducing errors. Original EDI implementations were entirely dependent on trading partners sharing the same standard format software. EDI documents flow straight back and forth between the sender's PC and the receiver's PC with no need to involve people [13]. The crucial thing about EDI is that it requires a standard format in order to function properly.

Data exchange is crucial for making different systems work together. It encourages collaboration and sharing of information across different areas like finance, healthcare, and research. However, successful data exchange requires careful planning, addressing challenges like data quality issues and security risks, and choosing the right methods and tools.

3.3. Data Sharing

Data sharing can be defined as the process that makes data accessible to other parties, typically within an organization or across several organizations. It does not necessar-

ily imply a movement of data files but focuses on the access and availability of them among entities. It comprises the process of facilitating its interchange between the actors interested in the data of concern.

Assuming that data is the infrastructure of science and is at the core of lots of businesses, we can see how important it is to share it. Indeed, it can be seen as the principal subject science is moving forward to, since sound data and data management are required to achieve all kinds of goals across almost every discipline [50]. Data sharing encompasses a diverse range of groups and actors, such as data producers, buyers, users, owners, providers, and consumers engaging in data sharing across different contexts and interactions. Some examples are Business-to-Business (B2B), Business-to-Consumer (B2C), Government-to-Government (G2G), and Government-to-Business (G2B). Since it involves different kinds of interactions, it also involves the sharing of different types of data, such as anonymous personal data, metadata, and aggregated data, each serving distinct purposes and subject to specific regulations. Overall, data sharing has the potential to drive innovation, encourage collaboration, and enhance transparency, but it requires careful planning to ensure its responsible and beneficial implementation. This entails the establishment of the environment and the infrastructure where the data sharing happens, as well as the application of legal and ethical standards. The use of technologies like blockchain and cloud computing has greatly improved data sharing by ensuring secure, transparent, and efficient transfer of data [49, 50]. As a result, data sharing has become increasingly popular over the last few years, gaining more traction than data exchange [28].

The growing emphasis on data sharing is a response to the increasing data intensiveness of modern science, which is propelled by technological advancements that enable massive data generation and require collaborative efforts for scientific breakthroughs. Data sharing allows for the re-analysis and reinterpretation of data, reason why it has become fundamental in collaborative environments where different stakeholders need to access and use the same datasets to drive collective outcomes, such as in scientific research, where it enhances reproducibility and facilitates new discoveries by building on existing data [41], contributing to scientific progress, particularly in interdisciplinary research. It also ensures data integrity and conservation of resources, acting as a safeguard against misconduct and serving as a training ground for emerging researchers [49].

Data sharing and exchange entail substantial ethical considerations, especially in balancing innovation with privacy. The ethical implications of data sharing encompass the necessity of obtaining informed consent from data subjects, ensuring transparency in data usage, and preventing the misuse of data. In addition, in the context of AI and machine learning, there is also the concern that shared data could unintentionally perpetuate bi-

ases if not handled with care. Organizations must integrate ethical AI practices into their data sharing processes to mitigate these risks, safeguarding against unintended harm or discrimination in data-driven insights.

To conclude, the drive towards data sharing is a response to both internal scientific needs and external policy developments. As data becomes more and more important to science, it's crucial to create an environment where sharing data is normal. This can be done by improving infrastructure, setting clear rules, and encouraging a culture that supports data sharing. By doing this, we can take scientific discovery to new levels.

3.4. Data Exchange as part of Data Sharing

The main difference between data sharing and data exchange lies in the scope and intent. Data sharing is mostly about availability and access, aiming to make data usable by others without necessarily transferring the data itself. It can be achieved through controlled access, such as user permissions, shared databases, or cloud storage where data is centrally located and accessed by users. In contrast, data exchange emphasizes the transfer of data from one system to another, often involving transformation to ensure compatibility. One of these is data exchange. This point of view makes it more clear that it actually represents the technical aspect that materializes the movement of data and must occur in compliance with the agreement established between the parties [40]. Data sharing is common in collaborative environments where multiple users need to access and use the same data, such as in cloud computing environments, shared research databases, or corporate intranets. Data exchange is often seen in scenarios involving the integration of systems, such as merging databases after a company merger, or syncing customer data between a company's CRM and ERP systems. As we have mentioned, the definition of data sharing is blurry and can mean several things. As proposed in [28], data sharing can be seen as a process, with data exchange being one of its components. The following image illustrates this relationship:

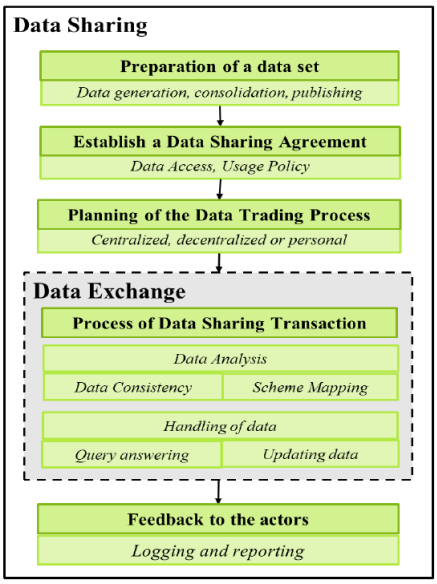


Figure 3.2: Data Exchanging as part of Data Sharing [28]

Despite receiving widespread support from policymakers, funding agencies, and scientific journals, academic researchers rarely share their research data with others. [18]. Businesses act in the same way. The thing is that data sharing in research is attributed to a vast potential for scientific progress: it allows the reproducibility of study results and the reuse of old data for new research questions, and it can improve market research.

3.5. Data Trustworthiness and Data Sovereignty

We define data trustworthiness as the reliability and accuracy of data, indicating how much it can be trusted for making informed decisions, conducting analyses, and supporting business processes. Trustworthy data has distinct qualities that ensure its credibility and dependability [33].

Data sovereignty refers to the ability of individuals, organizations, or even nations to control and regulate how their data is used, shared, and stored. It focuses on ensuring self-determination over data and protecting the rights of the data provider. In their paper, [44] emphasize the importance of data sovereignty in establishing trust between data providers and consumers, particularly in complex digital ecosystems. By outlining core aspects such as contractual agreements, data infrastructure, and trust, the authors highlight how data sovereignty plays a key role in enabling secure data-sharing practices while balancing economic and societal interests.

In their efforts to achieve collaborative business intelligence, organizations frequently face

the challenge of sharing valuable insights, like KPIs, without compromising the privacy and security of their raw data—known as data sovereignty. The recipients, whether another department or a different organization, must trust that the KPIs are authentic and accurately reflect the underlying raw data. This creates a tension between maintaining data sovereignty and ensuring the trustworthiness of the shared insights [40].

Implementing Collaborative Business Intelligence (CBI) has its challenges. Organizations often face obstacles such as data silos, where information is trapped within departments and not easily shared across the enterprise. Varying data quality standards can further complicate collaboration, as inconsistent or inaccurate data undermines trust in shared insights. Moreover, integrating different BI systems across organizations requires careful planning and coordination to ensure compatibility and interoperability. Successful implementations, like those seen at multinational corporations such as Procter & Gamble, have relied on establishing clear data governance policies and investing in cross-functional teams to drive CBI initiatives.

3.6. A Four-Phase ZKP-based Data Sharing Framework

In this section, we have presented the theoretical foundation that is central to our research. The structure closely mirrors that of section 3.3, emphasizing the foundational basis from which we developed our concept and subsequent framework. Our process can be divided into four phases illustrated in the schema below. It represents the process of our data trading framework, integrating zero-knowledge proofs to ensure data validity and privacy.

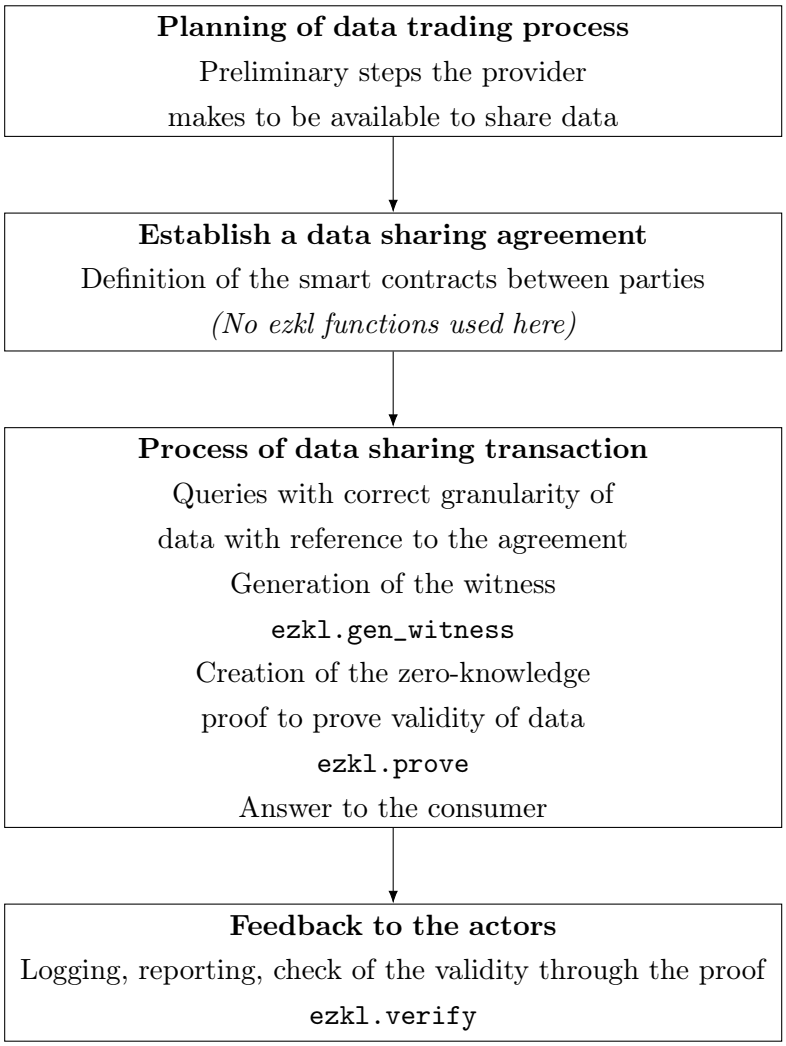


Figure 3.3: Data Sharing Process with Zero-Knowledge Proofs

1. **Planning of Data Trading Process:** In this phase, the data provider takes initial steps to prepare for data sharing. This involves publishing the hash of its private raw data. Along with the hash, the provider shares a revised data fact model that represents the structured data it is willing to share.

2. **Establish a Data Sharing Agreement:** Formalizing the smart contract between parties, defining the terms and conditions of data exchange. The required contracts for the framework are two, coded in Solidity. The first one manages the storing and retrieving of hash values. It is composed of:

   - storedHash: A public variable that stores a hash value of type bytes32.
   - setHash() function: This function allows any user to store a new hash value. It takes a bytes32 hash as input and updates the storedHash variable.

- getHash() function: This function allows users to retrieve the currently stored hash value.

The second contract is designed to manage a set of dimensions for querying business intelligence data and to check whether the requested query dimensions are allowed. It is formed as follows:

- Dimensions struct: Defines a set of boolean values representing different data dimensions such as ProductName, Category, Supplier, MonthOfSale, etc. Each dimension is either true (allowed) or false (disallowed).
- allowedDimensions: This is a public instance of the Dimensions struct. It is initialized in the constructor to define which dimensions are allowed by default.
- isQueryAllowed() function: This function takes an array of query dimensions (as strings) and checks if each one is allowed based on the current state of allowedDimensions. It compares the input with the predefined dimensions. If any dimension is not allowed, the function returns false.
- executeQuery() function: Before executing a query, this function checks if the dimensions in the query are allowed using isQueryAllowed(). If they are not, it throws an error. If they are allowed, it emits an event QueryCheck, which records the query details and the user who executed the query.
- QueryCheck: This event is emitted when a query is executed, allowing external systems to listen to and track the queries made by users.

3. **Process of Data Sharing Transaction:** Execution of data queries as per the agreement, and consequent configuration of the environment needed to create the ZK proof. The process begins by defining the circuit settings, which specify the configuration of the ZKP circuit, which defines the computation that will be proven. Calibration follows, ensuring the ZKP circuit is properly aligned with the computation and optimized for generating the proof. Once the settings are in place, a witness is generated using the function ezkl.gen_witness. A witness includes both the raw input data and intermediate results derived from applying the computation or model. Intermediate steps are also part of the witness, as they help prove that the entire computation was done correctly. The actual proof creation is performed using ezkl.prove. This step involves running the ZKP algorithm to produce cryptographic proof that demonstrates the integrity and correctness of the shared data.
4. **Feedback to the Actors:** This phase focuses on logging and reporting the results of the data-sharing transaction. The generated proofs are validated using ezkl.verify

to ensure the integrity and accuracy of the process. Verification confirms that the shared data complies with the agreed-upon terms.

This structured approach ensures a secure, transparent, and verifiable data trading process, leveraging cryptographic techniques to maintain data confidentiality and trust between parties.

As organizations scale their data sharing practices, the computational efficiency of Zero-Knowledge Proofs (ZKPs) becomes a critical consideration. While ZKPs provide robust security, they can be computationally intensive, especially when dealing with large datasets or complex proofs. Techniques like zk-SNARKs (Succinct Non-interactive Arguments of Knowledge) and zk-STARKs (Scalable Transparent Arguments of Knowledge) offer optimizations that reduce the computational burden and enhance scalability. These advancements ensure that the use of ZKPs remains feasible even as data volumes grow, enabling organizations to maintain security without compromising on performance.

With the theoretical foundations and framework now established, the subsequent chapter will delve into the practical implementation of these concepts.

4 | Proposed Solution

In this chapter, we will thoroughly explain the framework we have devised and designed, while in the next, we will focus on its implementation. In particular, we want to apply ZKP techniques to allow an organization to share in a trustworthy way the values of the KPIs of some internal processes without revealing the raw data used to compute them. This because raw data is so valuable they are not keen to share, would remain secret to everyone except the owner. At the same time, the receiver, even without having access to the raw data, has a reasonable certainty that the organization sharing the KPIs is not cheating on their values.

In order to have a clear understanding of this research, our approach will be top-down, starting with an overview and then explaining each step to achieve the goal.

To achieve this goal, we propose a framework that leverages immutable records on a blockchain and the cryptographic guarantees of ZKPs, to create an environment where businesses are empowered to share insights without ever exposing their most sensitive data.

Figure 4.1 reports the overview of the proposed framework.

4.1. Steps of The Solution

To introduce the approach, let assume that ORG1 and ORG2 belongs to a supply chain in a manufacturing domain. Notably, ORG2, to better organize its internal processes would like to have information about the performances on some of the internal processes enacted in ORG1. For this reason, it could be useful for ORG2 to know which are the values of a given set of KPIs related to the interested processes.

Based on this scenario, ORG1 is the data provider, while ORG2 is the requester. They find themselves working with a data space from a managerial point of view and a data lake from a technological one. In particular, we assume that the computation of the KPIs performed by ORG 1 are based on raw data organized, according to a DFM, in a data lake.

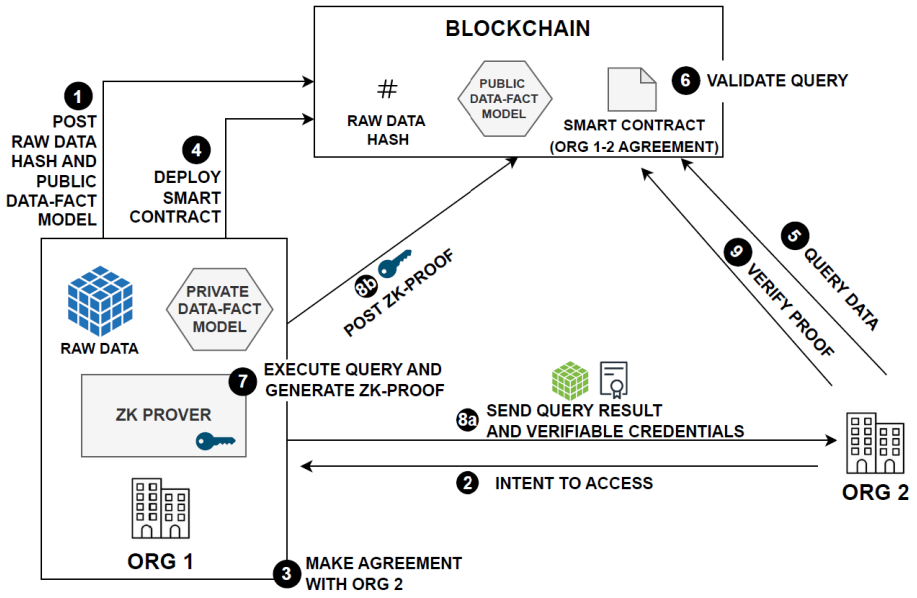


Figure 4.1: Framework [40]

The Data-Fact-Model (DFM) is a data model used in to organize structured data to allow their efficient analysis[47]. Usually adopted in data warehousing, also current data lake platforms offer capabilities to manage data according to the DFM. Here's a breakdown of each component:

- **Data**: This refers to raw, unprocessed information collected from various sources. Data can be structured (e.g., databases, spreadsheets) or unstructured (e.g., text, images). In the DFM framework, data serves as the foundation for generating facts and building models. Data quality and relevance are crucial for any analysis's success.
- **Fact**: Facts are insights or conclusions derived from the data. They represent the processed, cleaned, and structured information that can be directly interpreted or used for decision-making. Facts typically result from analyzing data using statistical methods, data mining, or other analytical techniques.
- **Model**: A model is a computational representation designed to mimic real-world processes constructed using data and derived facts. It is a tool for predicting outcomes, simulating scenarios, and analyzing relationships. Models can vary in complexity, from straightforward linear equations to sophisticated machine learning algorithms.

Let's see how the sharing follows. ORG 1 holds its raw data as a hypercube, managed by its in-house data warehouse. This data is represented through a DFM. Both the raw data

and the comprehensive DFM, which includes the full granularity and transformations of the data, are restricted and sensitive. Sharing them could inadvertently reveal underlying data patterns or business logic, thus threatening data sovereignty, privacy, and security.

ORG 1 begins by computing a hash of the dataset to initiate a secure sharing process. This hash is a unique identifier, ensuring data integrity and enabling verification without exposing the data. This hash provides a secure data fingerprint. Alongside this, ORG 1 crafts a public version of the DFM, a model that delineates a subset of operations and transformations that external parties can perform on the data. This public model is specifically tailored to allow the derivation of KPIs of interest while safeguarding the confidentiality of the raw data and the comprehensive transformations encapsulated in the private DFM.

Both the hash and the public DFM are then posted to a blockchain. This action ensures transparency, allowing other organizations to verify the data's integrity and understand the potential transformations applicable to KPI analysis. By leveraging blockchain technology, ORG 1 ensures an immutable record of the hash and the public model, ensuring trust and collaboration among participating entities without compromising data security or intellectual property.

When ORG 2 identifies a need to access specific data from ORG 1, it expresses its intent to engage in data analysis or collaboration based on predefined terms and conditions. This initiates a detailed negotiation phase where ORG 1 and ORG 2 outline the terms of data access and utilization. This agreement specifies which data operations are permissible, focusing on the scope delineated by the public DFM provided by ORG 1. The agreement ensures that both parties clearly understand the data access boundaries, permissible transformations, and the intended use of the derived insights. These agreed terms are then codified into a smart contract, which is deployed on the blockchain.

The smart contract regulates the data operations allowed within the boundaries set by the public DFM. It includes mechanisms to access the DFM stored on the blockchain, ensuring that all data operations initiated by ORG 2 adhere to the agreed terms. Furthermore, the smart contract is endowed with the capability to verify ZK-Proofs, facilitating the validation of query results and data transformations without exposing the underlying raw data. The smart contract can also reference the hash of the raw data stored on the blockchain to validate the integrity and authenticity of the data being accessed and analyzed.

With the smart contract in place, ORG 2 encodes the desired operation, adhering to the constraints and capabilities defined in the public DFM shared by ORG 1. This operation is

then packaged as an input and sent to the smart contract on the blockchain for validation. The smart contract performs an automated validation process, assessing the requested operation against the agreed-upon terms encapsulated within the contract and the public DFM. It ensures that the operation is within the scope of allowed transformations and does not violate any data privacy or integrity rules set by ORG 1. Once validated, an event is emitted, notifying ORG 1 of the approved request.

ORG 1 then initiates the execution of the requested transformation through a dedicated library coded to perform specific operations on the raw data.

Alongside computing the result of the transformation, the ZK circuit also generates a ZK-Proof. This proof serves a dual purpose: it attests to the correct execution of the requested operation and confirms that the operation was performed on the raw data whose hash matches the one previously posted on the blockchain by ORG 1. The ZK-Proof is crucial, as it allows ORG 2 to verify the integrity and authenticity of the operation's result without exposing the underlying raw data.

The hash of the ZK-Proof generated by ORG 1 is then posted to the blockchain. This ensures the immutability and non-repudiation of the proof, making it publicly accessible for validation and auditing. The blockchain acts as a transparent ledger where all executed operations, verifiable through ZK-Proofs, are logged, providing a tamper-proof record of all data interactions. The result of the transformation, a reduced hypercube of aggregated or transformed data, is then transmitted to ORG 2. Along with the result, ORG 1 sends verifiable credentials, which contain the transaction identifiers of both the original data operation request made by ORG 2 and the transaction that recorded the corresponding ZK-Proof on the blockchain. This dual-reference mechanism ensures traceability and verifiability, linking the result directly to its provenance and the integrity of its computation.

Thanks to the received files sent with the verifiable credentials, ORG 2 engages the smart contract on the blockchain, precisely the ZK-Verifier function embedded within it. This function allows ORG 2 to validate the ZK-Proof against the stored hash of the raw data, confirming that the requested operation was indeed executed on the exact dataset represented by the hash on the blockchain, thereby ensuring data integrity and the validity of the transformation. This interplay between ZK-Proofs, smart contracts, and blockchain technology ensures that ORG 2 can confidently utilize the transformed data, knowing it is derived directly from ORG 1's authentic raw data. Importantly, this entire process maintains the utmost data privacy, as the raw data itself is never exposed or transferred; only the proof of its correct transformation is shared. Moreover, the process upholds data

integrity and verifiability, which are vital components for trust in collaborative business environments.

The framework prioritizes data integrity, a critical aspect of secure data sharing. Data integrity is preserved through hashing and blockchain technology, which create a tamper-proof record of all data transactions. Hashing transforms data into a fixed-length string of characters, serving as a unique digital fingerprint for that data. Once this hashed data is recorded on a blockchain, it becomes embedded in an immutable ledger, ensuring that any attempt to alter the data is instantly detectable. This mechanism protects the integrity of the shared information by making unauthorized modifications evident. The security is further enhanced by the decentralized structure of the blockchain, which distributes the data across multiple nodes, making it exceptionally challenging for malicious actors to compromise the system.

In addition to its technical capabilities, the framework also addresses the issue of data ownership and control, which is becoming increasingly important as organizations seek to maintain sovereignty over their information in an era of pervasive data sharing. By using smart contracts, the framework allows data providers to set strict access controls and conditions for data usage, ensuring they retain complete control over their information even after it has been shared. This is particularly valuable in industries where data is crucial, as it reduces the risk of tampering, collusion, or fraud by third parties. The ability to enforce these controls through smart contracts not only enhances security but also provides a level of automation and efficiency that traditional methods cannot match.

Furthermore, by addressing these diverse and interconnected security concerns, the proposed framework does more than enhance the safety and reliability of data sharing between organizations. It lays the groundwork for a more trustworthy and collaborative business environment, where organizations can share data with confidence, knowing that cutting-edge cryptographic techniques and robust decentralized infrastructure protect their information. This, in turn, fosters more significant innovation and efficiency, as organizations can collaborate more freely and effectively without the constant fear of data breaches or unauthorized access. The framework also has the potential to set new standards for data security and privacy in the industry, serving as a model for other organizations to follow as they seek to improve their data-sharing practices.

In conclusion, the proposed framework represents a significant data security and privacy advancement. Integrating blockchain technology with zero-knowledge proofs and smart contracts offers a comprehensive solution that addresses the myriad challenges associated with inter-organizational data sharing. From ensuring data privacy and integrity

to protecting against cyber threats and empowering data providers with greater control over their information, this framework provides the tools necessary to create a secure, transparent, and trustworthy environment for data exchange. As organizations continue to navigate the complexities of the digital age, frameworks like this will be essential in safeguarding sensitive information and fostering collaboration in a secure and reliable manner.

5 | Implementation

Now that we have a clear idea of how the framework works, we will dig deeper into the actual implementation, starting with some choices from the beforehand. This chapter highlights the factuality of the framework, opening up to new ideas of data sharing and new opportunities for businesses to cooperate, grow, and improve. Before analyzing the architecture, it seems fair to mention the library chosen to develop the Zero-Knowledge Proof part and explain why.

EZKL allows for easy off-chain execution of large and complex computations, making programming custom functions in Python simple. It offers flexibility by not limiting you to predefined functions, supports unlimited input sizes using hashing, and does not rely on a centralized sequencer. Ezkl is remarkable for its decentralization, contrary to other tools like Plonk or Pragma, which need specific setups and do not allow decentralization like Ezkl. It doesn't need a centralized sequencer, meaning the entire process stays as decentralized as the blockchain. This makes EZKL a strong choice.

One of the most significant limitations in the blockchain world is the abrupt learning curve, especially regarding complex cryptographic methods like SNARKs. These often require an in-depth understanding of cryptography to use effectively. Plonk has made things easier than older SNARKs, but still, it requires technical knowledge. EZKL, however, completely changed the rules by allowing developers to use Python, a language many are already familiar with. This makes advanced cryptographic tools accessible to a broader range of developers, not just experts. Traditional SNARKs and even more advanced tools like Plonk come with predefined functions limiting how much you can customize your solution. This was the main reason ezkl was the ultimate choice: flexibility and the possibility to design unique functions following the framework logic were needed.

Moreover, as projects grow, the ability to handle large amounts of data efficiently becomes crucial. Unlike Plonk and similar frameworks, which are mainly known for their speed at the expense of other things they lack, such as setups and the input size they need. Even in this case, EZKL comes to our rescue: EZKL straightforwardly tackles this problem by using hashing techniques to manage virtually unlimited input sizes without requiring

complicated configurations. This makes it easier to scale your application without getting caught up in the complexities of cryptographic optimization.

With EZKL, you can craft specific and secure computational processes that are tailored to your needs, thanks to the platform's robust design. This is particularly useful when implementing complex logic without compromising security. Giving developers autonomy is key as the blockchain evolves. They want the freedom to design, implement, and modify their processes without being restricted by the tools they use.

To recap, while tools like Plonk, Pragma, and traditional SNARKs offer significant benefits in zero-knowledge proofs and off-chain computation, EZKL stands out for aligning with the core values of blockchain—decentralization, accessibility, and flexibility. It combines ease of use with scalability and customization, allowing developers to build sophisticated, secure, and scalable applications without deep cryptographic expertise

5.1. System Architecture

The architecture consists of two Solidity smart contracts, which handle decentralized storage and verification, and Python/PyTorch library and finally scripts that together perform data processing and zero-knowledge proof generation. The system is designed to ensure data integrity and confidentiality, making it suitable for applications requiring trustworthy data handling and complex analytics.

As we have imagined it in the previous chapter, ORG 1 will be our provider, while ORG 2 will be the requester of data.

Before the process starts, ORG 1 decides which data it is willing to share and creates a data fact model outlining the dimensions it intends to make available. ORG 1 publishes the model, along with the hash of the complete and private data fact model, on the blockchain.

Upon receiving a request from ORG 2, ORG 1 retrieves the hash from the blockchain and compares it with the hash of the private data it stores locally. If the hashes do not match, the process halts to prevent any data integrity issues. If the hashes match, the smart contracts then verify whether the requested data dimensions are allowed based on the data fact model. If they are unauthorized, the process stops. Otherwise, the data flow proceeds to the next step.

After successfully passing these checks, ORG 1 begins its operation on the dataset. The non-numeric data is encoded, converted into a tensor, and processed through the defined

operations using the EZKL library. These operations can be combined, enabling multiple queries to be answered with a single proof, which saves time, energy, and cost.

It is time now to export an onnx model that is one of the needed component to create the proof. In fact, exploiting the ezkl library the proof is created with the following line of code:

```
proof = ezkl.prove(witness_path, compiled_filename,
                        pk_path, proof_path, "single")
```

```
res = ezkl.gen_settings(model_onnx_path, settings_filename)
```

The first step is to generate a settings file based on the ONNX model. This settings file includes important parameters needed to configure the circuit and proof generation process. Parameters:

- ‘model_onnx_path‘: Path to the ONNX model file.
- ‘settings_filename‘: Output path for the generated settings file.

```
srs_path = await ezkl.get_srs(settings_filename, logrows=logrows)
```

Next, you generate or retrieve the Structured Reference String (SRS), which is crucial for setting up the zero-knowledge proof system. The SRS enables efficient proof creation. Parameters:

- ‘settings_filename‘: Path to the settings file that influences the SRS generation.
- ‘logrows‘: A parameter that determines the size or complexity of the SRS.

```
res = await ezkl.calibrate_settings(input_json_path,
                model_onnx_path, settings_filename, "resources")
```

After generating the initial settings and SRS, the next step is to calibrate the settings. Calibration optimizes the parameters for running the circuit efficiently, potentially based on resource usage. Parameters:

- ‘input_json_path‘: Path to the JSON file containing the input data.
- ‘model_onnx_path‘: Path to the ONNX model file.
- ‘settings_filename‘: Path where the calibrated settings will be stored.
- ‘"resources"‘: Likely a mode for calibration, focusing on resource optimization.

```
ezkl.compile_circuit(model_onnx_path, compiled_filename, settings_filename)
```

This command compiles the machine learning model (likely in ONNX format) into a format that can be used by the zero-knowledge proof system.

```
res = ezkl.setup(compiled_filename, vk_path, pk_path)
```

After compiling the circuit, the next step is to set up the proving key (‘pk‘) and verification key (‘vk‘). These keys are essential for generating and verifying the proof, respectively. Parameters:

- ‘compiled_filename‘: Path to the compiled circuit.
- ‘vk_path‘: Output path for the verification key.
- ‘pk_path‘: Output path for the proving key.

```
res = await ezkl.gen_witness(input_json_path, compiled_filename, witness_path)
```

Finally, the witness is generated. The witness represents the input data that satisfies the circuit's constraints and is necessary for proving that the computation was performed correctly. Parameters:

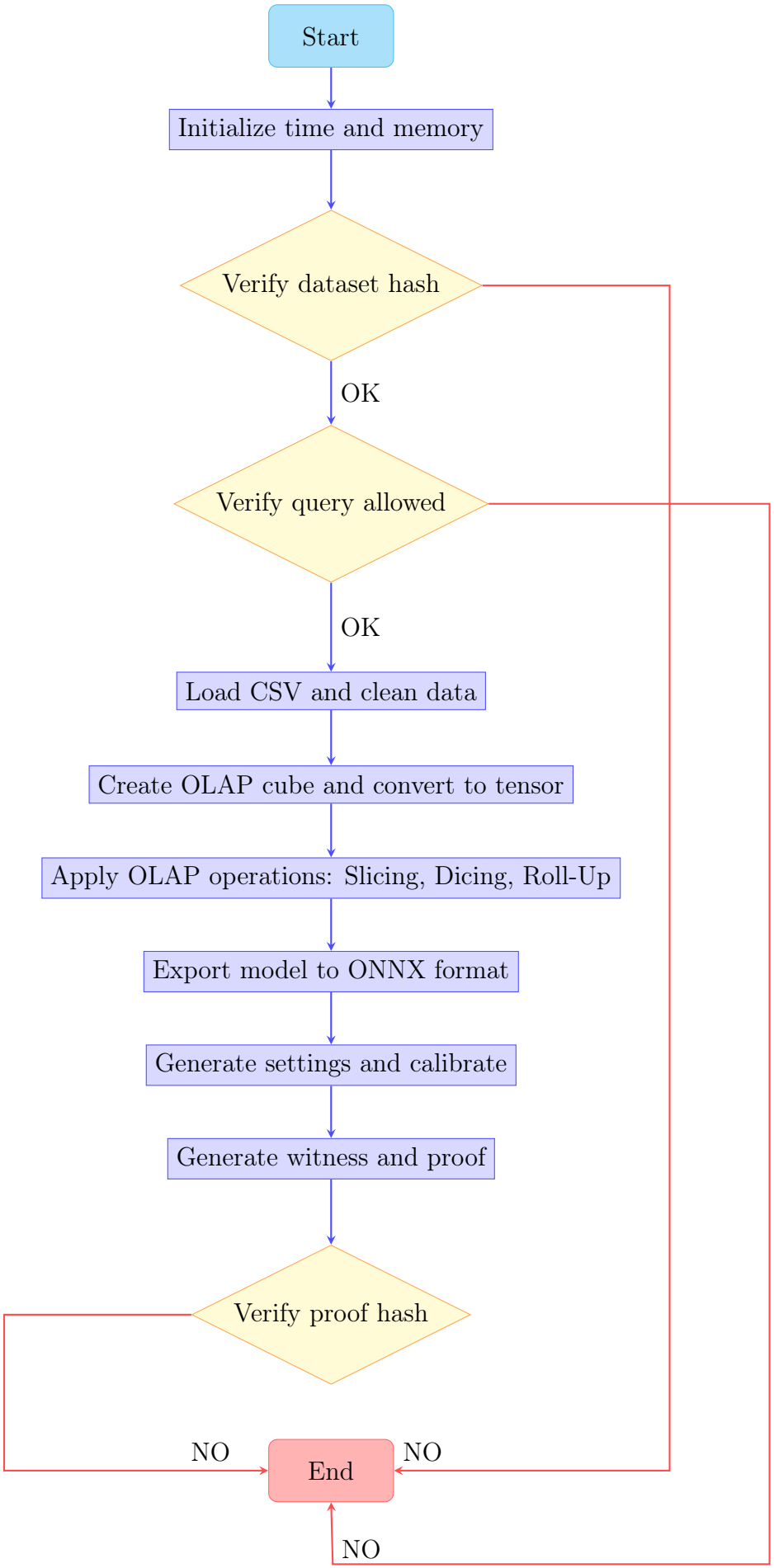
- ‘input_json_path‘: Path to the JSON file containing the input data.
- ‘compiled_filename‘: Path to the compiled circuit.
- ‘witness_path‘: Output path where the generated witness will be saved.

Once the proof is generated, its hash is computed and then published on the blockchain. ORG 1 sends all files necessary to the verification to ORG 2, using verifiable credentials to guarantee its identity. Now is ORG 2 turn to verify the hash of the proof the provider has sent with the hash published on the blockchain. If they correspond one can proceed with the verification. If it is not the case the two proofs do not match.

```
res = ezkl.verify(proof_path, settings_filename, vk_path)
```

This command verifies the generated proof against the verification key (‘vk‘) to ensure that the computation was performed correctly as per the settings file. Parameters:

- ‘proof_path‘: Path to the proof file that needs to be verified.
- ‘settings_filename‘: Path to the settings file that was used during the proof generation.
- ‘vk_path‘: Path to the verification key that is used to check the validity of the proof.


Start
Initialize time and memory
Verify dataset hash
OK
Verify query allowed
OK
Load CSV and clean data
Create OLAP cube and convert to tensor
Apply OLAP operations: Slicing, Dicing, Roll-Up
Export model to ONNX format
Generate settings and calibrate
Generate witness and proof
Verify proof hash
NO
End
NO
NO

5.2. Experimental Validation

In this section, we will focus on analyzing the performance of a series of queries performed on several datasets used to explore the functioning of the software in terms of different aspects and technicalities. Let's first focus on which databases were chosen and which queries were written to study the software performance. After a thorough research on Kaggle, it was deemed wise to choose three databases which differs in size, indeed every dataset is almost triple of the precedent.

- A relatively small one with 2,800 rows (Small database) [30];
- A medium-sized one with 10,000 rows (Medium database) [46];
- A large database with 30,000 rows (Big database) [35].

For the purposes of the analysis, all the datasets are corporate datasets, so while they differ in size, number of columns, and number of rows, they all represent the target audience we have discussed about many times: companies that may be interested in sharing data. In fact, they are all business related. The first database, the smaller one that we will also call database 1, is about selling motorcycles, cars, and other vehicles. It it is made of the following columns:

1. ORDERNUMBER: unique identifier for each order
2. QUANTITYORDERED: the number of units of a product ordered in a specific transaction
3. PRICEEACH: price per unit of the product ordered
4. ORDERLINENUMBER: the line number in the order, useful for orders containing multiple items
5. SALES: the total sales amount for the order, calculated as QUANTITYORDERED * PRICEEACH
6. ORDERDATE: the date when the order was placed, which can be used for analyzing sales trends over time
7. STATUS: the current status of the order, such as 'Shipped', 'Cancelled', etc.
8. QTR_ID: the quarter of the year when the order was placed (1 for Q1, 2 for Q2, etc.)
9. MONTH_ID: the month of the year when the order was placed (e.g., 1 for January, 12 for December)

10. YEAR_ID: the year when the order was placed
11. PRODUCTLINE: the category or product line to which the ordered item belongs (e.g., Classic Cars, Motorcycles)
12. MSRP: manufacturer's suggested retail price, which is the recommended selling price for the product
13. PRODUCTCODE: the unique code assigned to the product, which can be used to track and manage inventory
14. CUSTOMERNAME: the name of the customer who placed the order
15. PHONE: the phone number of the customer, useful for contact and customer service purposes
16. ADDRESSLINE1: the primary address of the customer, where the product is shipped
17. ADDRESSLINE2: an additional address line for the customer, if applicable (often used for apartment numbers or suites)
18. CITY: the city where the customer resides or where the order is being shipped
19. STATE: the state or region of the customer's address (not always present for international customers)
20. POSTALCODE: the postal code or ZIP code for the customer's address
21. COUNTRY: the country where the customer resides or where the order is being shipped
22. TERRITORY: the sales territory assigned to the order, often used for sales performance tracking by region
23. CONTACTLASTNAME: the last name of the primary contact person for the order
24. CONTACTFIRSTNAME: the first name of the primary contact person for the order
25. DEALSIZE: the size of the deal or order, typically categorized as 'Small', 'Medium', or 'Large', which can be used for sales analysis and forecasting.

It represent the sales record of vehicles from 2003 to 2005 of a company. The second one differs since it contains credit scores, money spent, credit cards, and salaries—all kinds of information useful for studying people's behaviors. In particular it is made of these columns:

1. CustomerId: unique identifier for each customer
2. Surname: the surname or last name of the customer
3. CreditScore: the credit score of the customer, which can indicate their creditworthiness
4. Geography: the country or region where the customer resides
5. Gender: the gender of the customer, typically categorized as 'Male' or 'Female'
6. Age: the age of the customer in years
7. Tenure: the number of years the customer has been with the bank
8. Balance: the current balance in the customer's account
9. NumberOfProducts: the number of different banking products the customer has with the bank
10. HasCrCard: indicator of whether the customer has a credit card (1 for yes, 0 for no)
11. ActiveMembers: indicator of whether the customer is an active member (1 for yes, 0 for no)
12. EstimatedSalary: the estimated annual salary of the customer
13. Exited: indicator of whether the customer has left the bank (1 for yes, 0 for no)
14.

The third and largest one, database 3, is about pizza sales.

1. pizza_id: unique identifier for each pizza in the dataset
2. order_id: unique identifier for each order, which can include one or more pizzas
3. pizza_name_id: identifier for the specific type of pizza, used to track different pizza variations
4. quantity: the quantity of the specific pizza ordered in a particular transaction
5. order_date: the date when the order was placed, useful for analyzing sales patterns over time
6. order_time: the time of day when the order was placed, which can be used to study peak ordering times

7. unit_price: the price of a single unit of the pizza ordered
8. total_price: the total price for the quantity of pizzas ordered, calculated as quantity * unit_price
9. pizza_size: the size of the pizza (e.g., Small, Medium, Large), indicating the portion size of the pizza
10. pizza_category: the category of the pizza (e.g., Classic, Veggie, Supreme), used to group similar types of pizzas
11. pizza_ingredients: the list of ingredients used in the pizza, which can help analyze customer preferences based on ingredients
12. pizza_name: the name of the pizza, often used for menu and marketing purposes

With the aim of obtaining fair, comparable results despite the differences between databases and being able to appreciate the differences in time and memory used as the number of rows increased, three queries were submitted to the three different databases. Each of them increased the difficulty. The first represents a slicing operation, the second adds a dicing operation to the first, and the third adds a roll-up to these two.

Every test (submitted query) was performed 5 times, and then the mean of the metrics we were interested in was computed. The tests were conducted on a PC with the following specifications: 13th Gen Intel Core i9-13900H processor with a clock speed of 2.60 GHz, 32 GB of RAM, and a 64-bit operating system based on x64 architecture.

We will evaluate calibration, proof generation, verification times, and memory usage to better understand the system's scalability and identify potential optimization areas.

5.2.1. Query 1

Let's start with Query 1: the slicing operation. For database one a slice operation was performed on the year of sale (YEAR_ID), specifically on year 2005. The second database, was filtered on the gender (Gender, woman was selected) column and the one that keep tracks on whether a client is active or not (ActiveMembers). Regarding the pizza sales dataset, the slicing was based on the pizza's size (pizza_size) and category (pizza_category): medium classic pizzas were selected.

Metric	Test 1	Test 2	Test 3	Test 4	Test 5	Mean
Time Metrics (s)						
Calibration Time	10.05	9.61	9.60	9.84	9.40	9.70
VK Time	0.416	0.368	0.374	0.380	0.404	0.3884
PK Time	0.288	0.294	0.279	0.315	0.302	0.2956
Proof Time	0.811	0.39	0.39	0.795	0.815	0.8048
Verify Time	0.35	0.39	0.39	0.38	0.44	0.39
Total Time	14.29	13.46	13.47	13.85	13.74	13.762
Hash Verification Time	0.06	0.06	0.05	0.07	0.05	0.058
Query Verification Time	0.13	0.11	0.08	0.13	0.09	0.108
OLAP Operations Time	0.05	0.03	0.03	0.04	0.03	0.036
ONNX Export Time	0.32	0.26	0.23	0.24	0.26	0.262
Proof Generation Time	13.70	12.98	13.06	13.35	13.29	13.276
Memory Usage Metrics (MB)						
Calibration Memory Usage	72.23	71.11	71.26	71.02	70.98	71.32
Total Memory Usage	184.11	157.13	151.91	154.99	176.34	164.896
Hash Verification Memory Usage	0.58	0.64	0.70	0.71	0.74	0.674
Query Verification Memory Usage	0.26	0.26	0.26	0.26	0.26	0.26
OLAP Operations Memory Usage	7.88	8.67	7.82	7.82	8.00	8.038
ONNX Export Memory Usage	28.84	27.80	28.62	28.72	28.57	28.51
Proof Generation Memory Usage	145.92	119.11	113.86	116.90	138.13	126.784

Table 5.1: Time and Memory Usage Metrics for Small DB - Query 1

Metric	Test 1	Test 2	Test 3	Test 4	Test 5	Mean
Time Metrics (s)						
Calibration Time	70.22	69.69	68.93	70.12	70.79	69.95
VK Time	6.330	6.515	6.392	6.980	7.901	6.82
PK Time	8.352	9.321	9.208	9.310	10.324	9.33
Proof Time	15.786	16.554	18.439	18.986	17.44	
Verify Time	0.742	0.682	0.676	0.716	0.707	0.7046
Total Time	183.04	182.62	182.69	182.29	188.05	183.74
Hash Verification Time	0.05	0.06	0.06	0.04	0.05	0.052
Query Verification Time	0.07	0.08	0.10	0.06	0.08	0.078
OLAP Operations Time	0.11	0.11	0.11	0.08	0.19	0.12
ONNX Export Time	0.38	0.38	0.40	0.41	0.56	0.426
Proof Generation Time	182.14	181.69	181.75	181.45	186.91	182.79
Memory Usage Metrics (MB)						
Calibration Memory Usage	2320.57	2355.44	2341.02	2345.06	2365.68	2345.55
Total Memory Usage	3118.43	3199.28	3113.16	3145.12	3157.94	3146.79
Hash Verification Memory Usage	0.70	0.64	0.71	0.63	0.62	0.66
Query Verification Memory Usage	0.26	0.26	0.26	0.26	0.26	0.26
OLAP Operations Memory Usage	7.91	8.48	8.54	8.01	8.38	8.26
ONNX Export Memory Usage	30.58	29.97	29.91	30.47	30.36	30.26
Proof Generation Memory Usage	3067.81	3148.77	3062.59	3094.59	3107.17	3096.19

Table 5.2: Time and Memory Usage Metrics for Medium DB - Query 1

Metric	Test 1	Test 2	Test 3	Test 4	Test 5	Mean
Time Metrics (s)						
Calibration Time	164.26	156.94	155.77	155.43	157.44	157.97
VK Time	15.918	15.93	15.451	15.723	15.39	15.68
PK Time	19.520	20.267	19.208	20.340	20.52	19.97
Proof Time	37.396	37.282	37.516	35.906	34.37	36.49
Verify Time	1.915	1.924	1.888	1.751	1.897	1.88
Total Time	406.79	398.49	403.79	392.75	397.78	399.92
Hash Verification Time	0.09	0.08	0.09	0.10	0.08	0.088
Query Verification Time	0.07	0.08	0.09	0.08	0.07	0.078
OLAP Operations Time	0.27	0.22	0.29	0.24	0.27	0.258
ONNX Export Time	0.34	0.36	0.36	0.44	0.38	0.376
Proof Generation Time	405.39	397.12	402.22	391.12	396.28	398.43
Memory Usage Metrics (MB)						
Calibration Memory Usage	6327.52	6386.02	6052.45	6389.38	6252.97	6281.67
Total Memory Usage	7120.81	7020.82	6784.70	7091.38	7043.58	7012.26
Hash Verification Memory Usage	0.56	0.65	0.63	0.64	0.58	0.612
Query Verification Memory Usage	0.26	0.26	0.26	0.26	0.26	0.26
OLAP Operations Memory Usage	13.52	13.31	13.26	13.27	13.24	13.32
ONNX Export Memory Usage	29.67	29.87	29.62	29.52	29.58	29.65
Proof Generation Memory Usage	7046.18	6946.11	6710.32	7017.07	6969.30	6937.80

Table 5.3: Time and Memory Usage Metrics for Big DB - Query 1

We will focus on the time and memory necessary for the generation of the proof. Then, we will examine the same characteristics for the verification of the proof, as they clearly differ in performance between database generation and verification as the database size increases.

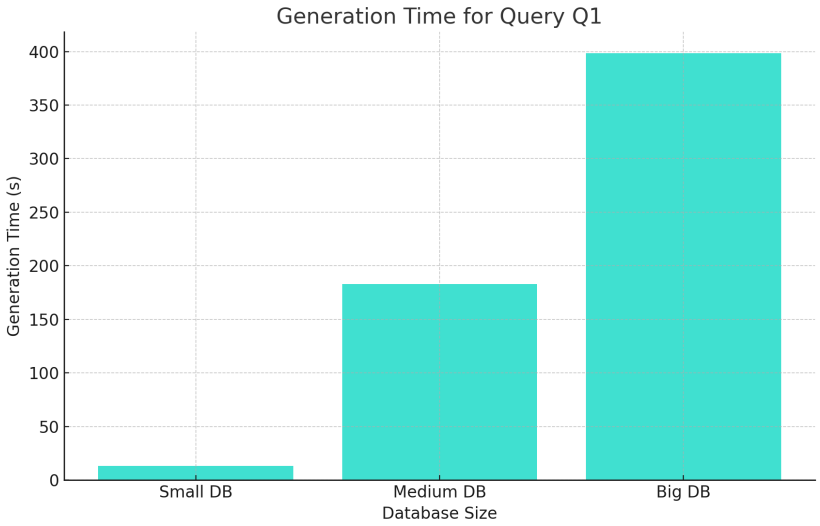


Figure 5.1: Generation Time for Query Q1

The time required for database generation increases significantly as the database size grows. For the first database, it takes 13.83 seconds, while for the bigger database, it leaps to 383.85 seconds. This indicates that the generation process is strongly influenced by the size of the database. The reason behind this is the increasing complexity and the amount of data that needs to be processed and generated as the database grows.

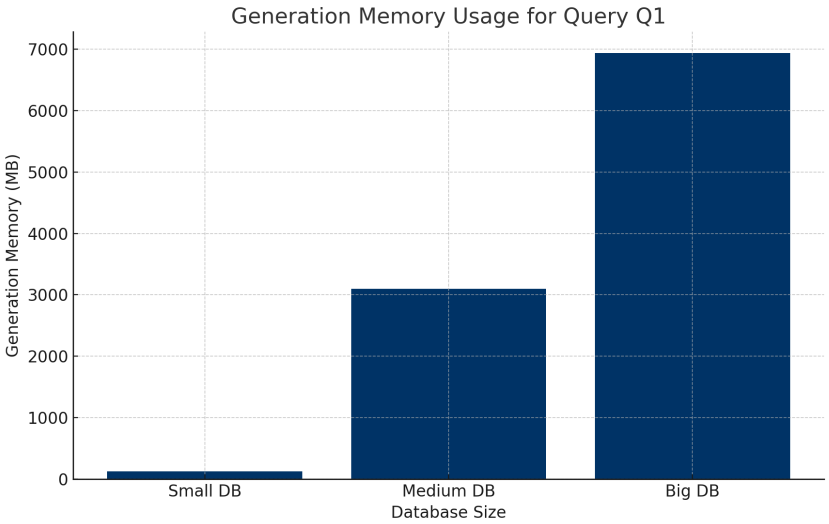


Figure 5.2: Generation Memory Usage for Query Q1

The bar chart represents the memory usage in MB for query Q1 as it scales with different

database sizes. The observations show that for the Small DB, memory usage is minimal and similar to the generation time. In the case of the Medium DB, there is a considerable increase in memory usage, surpassing 3,000 MB. The Big DB uses the most memory, close to 7,000 MB. In conclusion, the memory usage for query generation grows significantly with database size, indicating that as the amount of data increases, the system requires a much larger memory footprint to process the query.

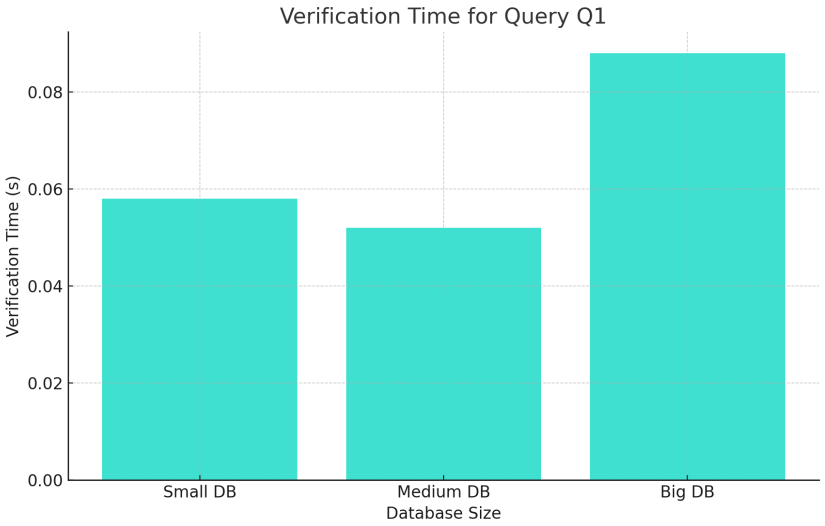


Figure 5.3: Verification Memory Usage for Query Q1

The bar chart illustrates the time required to verify the results of query Q1 for databases of varying sizes . It is observed that the verification time for the Small DB is approximately 0.06 seconds, while for the Medium DB, it slightly decreases to just under 0.06 seconds. As for the Big DB, the verification time increases to just over 0.08 seconds.

In conclusion, the verification time for query Q1 remains consistently low across all database sizes, with a slight increase as the database size grows. This suggests that the verification process remains efficient, even for larger databases.

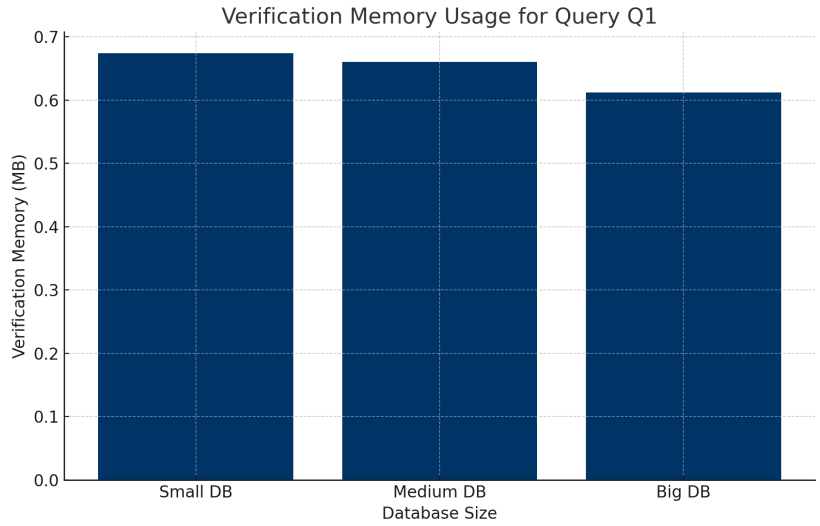


Figure 5.4: Verification Memory Usage for Query Q1

The bar chart depicts the memory usage during the verification of query Q1 for various database sizes. It is observed that the memory usage remains fairly consistent at around 0.65 MB across all database sizes. This indicates that the verification process for query Q1 is memory-efficient, showing consistent memory usage regardless of the dataset size. This suggests that the memory overhead for verifying the results does not significantly increase as the dataset size grows.

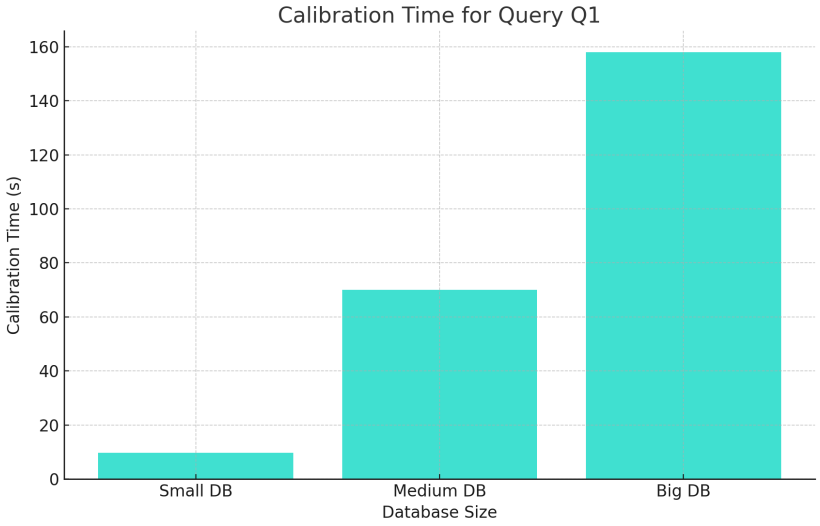


Figure 5.5: Calibration Time for Query Q1

In the chart, it is observed that the calibration time for query Q1 increases significantly as the database size grows. For small databases, the calibration time is around 20 seconds, while for medium databases, it jumps to about 80 seconds. The calibration time is longest for big databases, taking around 160 seconds. This suggests that the calibration process becomes more complex as the database size increases, highlighting the need for optimization when dealing with larger datasets.

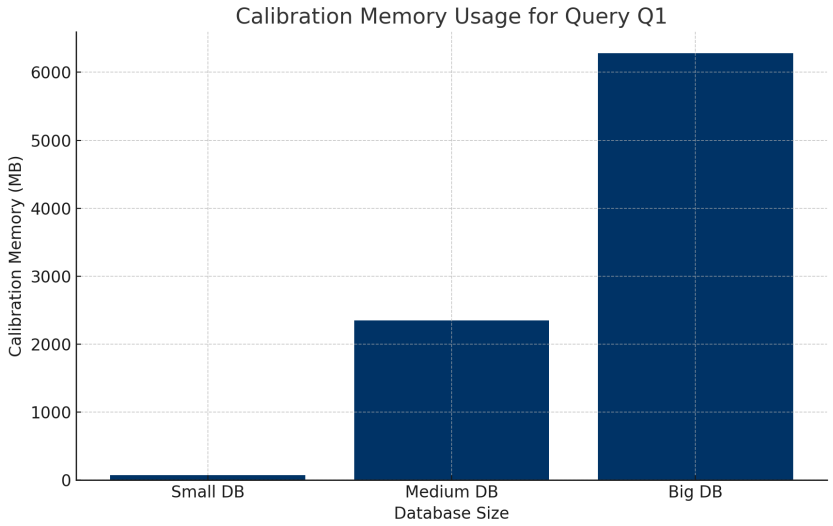


Figure 5.6: Calibration Memory Usage for Query Q1

The graph depicts the memory usage during calibration of query Q1 across different database sizes. During the calibration, it was observed that the Small DB uses minimal memory. However, the memory usage increases sharply for the Medium DB, exceeding 2,000 MB. The memory usage for the Big DB jumps to around 6,000 MB. As a conclusion, it can be noted that memory usage for query Q1 during calibration grows significantly with the increase in database size. This indicates that as the calibration process demands more resources with larger datasets, memory-efficient techniques might be required to improve performance for larger databases.

While the generation process shows scalability issues regarding time and memory usage as the database size increases, the verification process demonstrates efficiency and consistency. This process might benefit from optimizations to handle larger databases better. Even though calibration consumes a significant amount of resources, particularly in smaller databases, the generation phase ultimately represents the bottleneck in the system as the database size grows. Still, calibration might be faster by using a smaller batch, depending on how well the database is known to be consistent.

5.2.2. Query 2

As for the second query, the dicing operation was performed. On the first database it collected all those rows representing motorcycles sales sold in France or Norway. The second one, gathers all the clients living in either France or Spain. Pizzas were filtered to look for Hawaiian pizza or Italian pizza.

Metric	Test 1	Test 2	Test 3	Test 4	Test 5	Mean
Time Metrics (s)						
Calibration Time	27.99	27.98	27.50	28.47	27.25	27.838
VK Time	0.497	0.457	0.498	0.460	0.452	0.4728
PK Time	0.884	0.901	0.866	0.877	0.912	0.888
Proof Time	0.42	0.35	0.37	0.32	0.42	0.376
Total Time	32.51	32.46	32.03	32.57	31.51	32.216
Hash Verification Time	0.05	0.05	0.05	0.06	0.05	0.052
Query Verification Time	0.09	0.07	0.07	0.10	0.07	0.08
OLAP Operations Time	0.03	0.03	0.03	0.03	0.04	0.032
ONNX Export Time	0.24	0.24	0.23	0.23	0.26	0.240
Proof Generation Time	32.08	32.05	31.63	32.14	31.09	31.798
Memory Usage Metrics (MB)						
Calibration Memory Usage	70.43	70.24	70.72	70.45	70.32	70.432
Total Memory Usage	150.47	178.63	152.72	151.09	186.23	163.828
Hash Verification Memory Usage	0.59	0.59	0.70	0.74	0.66	0.656
Query Verification Memory Usage	0.26	0.26	0.26	0.26	0.26	0.26
OLAP Operations Memory Usage	7.97	7.81	8.00	8.02	7.88	7.936
ONNX Export Memory Usage	28.72	28.74	28.76	28.72	28.64	28.716
Proof Generation Memory Usage	112.29	140.59	114.35	112.70	148.15	125.616

Table 5.4: Time and Memory Usage Metrics for Small DB - Query 2

Metric	Test 1	Test 2	Test 3	Test 4	Test 5	Mean
Time Metrics (s)						
Calibration Time	104.70	105.40	105.23	103.96	103.74	104.61
VK Time	6.614	7.561	8.131	8.264	8.190	7.75
PK Time	9.946	8.521	10.402	10.329	10.573	9.95
Proof Time	19.140	18.374	19.600	19.333	19.452	19.18
Verify Time	0.712	0.722	0.716	0.718	0.718	0.7172
Total Time	224.18	234.25	224.75	221.72	221.25	225.23
Hash Verification Time	0.06	0.04	0.07	0.04	0.04	0.05
Query Verification Time	0.07	0.07	0.09	0.07	0.07	0.074
OLAP Operations Time	0.12	0.21	0.12	0.10	0.09	0.128
ONNX Export Time	0.38	0.57	0.39	0.37	0.36	0.414
Proof Generation Time	223.26	233.07	223.81	220.89	220.41	224.29
Memory Usage Metrics (MB)						
Calibration Memory Usage	2357.96	2352.87	2354.45	2319.50	2295.30	2336.02
Total Memory Usage	3198.55	3235.43	3222.00	3101.03	3174.71	3186.34
Hash Verification Memory Usage	0.69	0.62	0.64	0.68	0.71	0.668
Query Verification Memory Usage	0.26	0.26	0.26	0.26	0.26	0.26
OLAP Operations Memory Usage	12.56	10.83	10.61	10.48	11.26	11.15
ONNX Export Memory Usage	27.93	28.08	27.89	28.11	27.23	27.85
Proof Generation Memory Usage	3145.89	3184.75	3171.39	3050.36	3124.04	3135.29

Table 5.5: Time and Memory Usage Metrics for Medium DB - Query 2

Metric	Test 1	Test 2	Test 3	Test 4	Test 5	Mean
Time Metrics (s)						
Calibration Time	182.90	181.80	182.86	178.82	181.71	181.62
VK Time	16.30	15.97	15.892	15.931	15.848	15.99
PK Time	20.756	20.436	20.455	21.40	20.731	20.76
Proof Time	37.88	37.575	37.57	38.59	38.267	37.98
Verify Time	1.898	1.866	1.875	1.901	1.938	1.896
Total Time	424.61	426.39	422.78	421.33	424.24	423.87
Hash Verification Time	0.08	0.08	0.07	0.09	0.08	0.08
Query Verification Time	0.07	0.08	0.07	0.08	0.10	0.08
OLAP Operations Time	0.27	0.26	0.23	0.27	0.29	0.264
ONNX Export Time	0.37	0.36	0.37	0.44	0.35	0.378
Proof Generation Time	423.10	424.80	421.43	419.71	422.69	422.35
Memory Usage Metrics (MB)						
Calibration Memory Usage	6106.20	6168.23	6229.70	6219.43	6165.25	6177.76
Total Memory Usage	6847.19	6978.65	7021.93	6993.52	6962.11	6960.68
Hash Verification Memory Usage	0.64	0.64	0.70	0.66	0.56	0.64
Query Verification Memory Usage	0.26	0.26	0.26	0.26	0.26	0.26
OLAP Operations Memory Usage	15.79	15.98	15.88	15.76	15.82	15.85
ONNX Export Memory Usage	27.12	27.14	27.38	27.10	27.23	27.19
Proof Generation Memory Usage	6772.84	6904.15	6947.22	6919.18	6887.49	6886.18

Table 5.6: Time and Memory Usage Metrics for Big DB - Query 2

Below are the charts that will highlight a similar behavior that was previously illustrated.

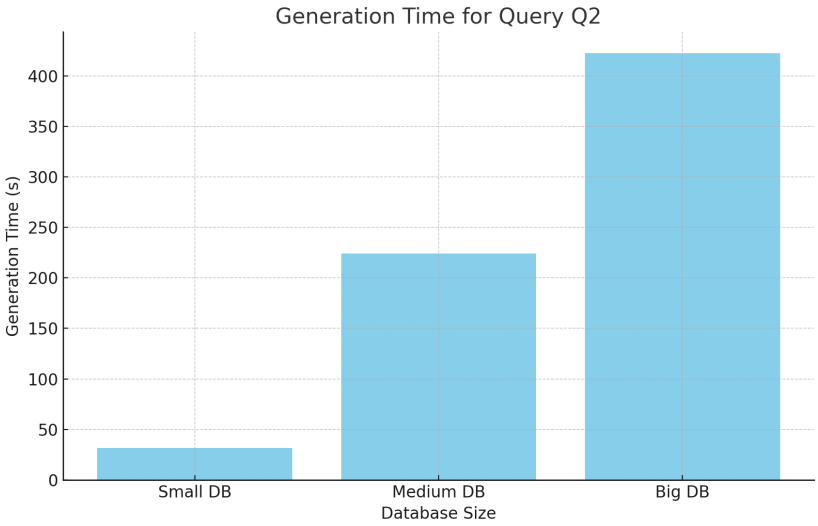


Figure 5.7: Generation Time for Query Q2

The graph depicts the generation time in seconds for query Q2 across three different database sizes: Small, Medium, and Big. The observations reveal that for the Small DB, the time is relatively low, approximately around 50 seconds. However, for the Medium DB, there is a substantial increase, with the time going up to around 200 seconds. The Big DB takes the longest time, nearing 400 seconds, which is consistent with the behavior observed for Query Q1. In conclusion, it is evident that Query Q2 follows a similar pattern as Q1, with the processing time scaling significantly with the increase in database size, particularly when moving from the Medium to Big DB. This steep increase in time indicates that Q2 may involve complex operations that become increasingly computationally expensive with larger datasets.

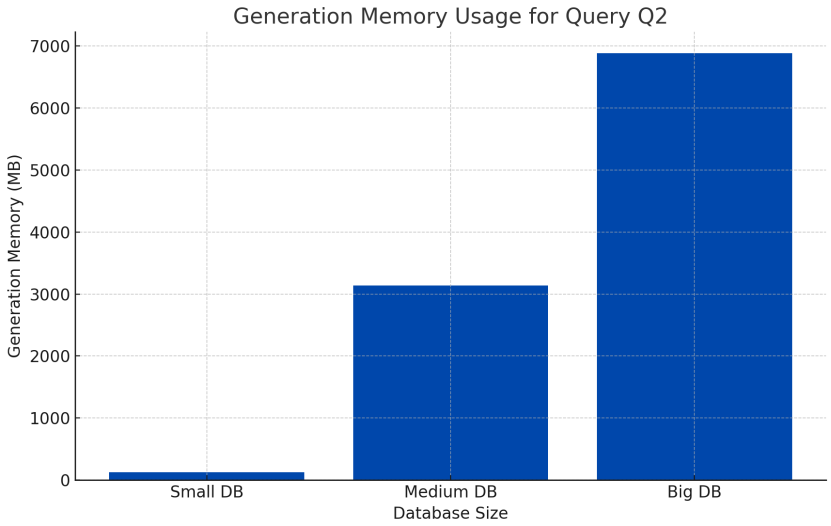


Figure 5.8: Generation Memory Usage for Query Q2

The bar chart visualizes the memory consumption in MB for query Q2 across different database sizes. It is evident that memory usage increases with the size of the database. For the small database, the memory usage is minimal, around 100 MB. Moving on to the medium-sized database, the memory usage jumps significantly to around 3,000 MB. Finally, for the large database, memory usage peaks near 7,000 MB. This indicates that memory usage increases substantially with database size for Query Q2, similar to Query Q1. The demand for memory is particularly high for the large database, likely due to the complexity of the operations required for the query as the dataset grows.

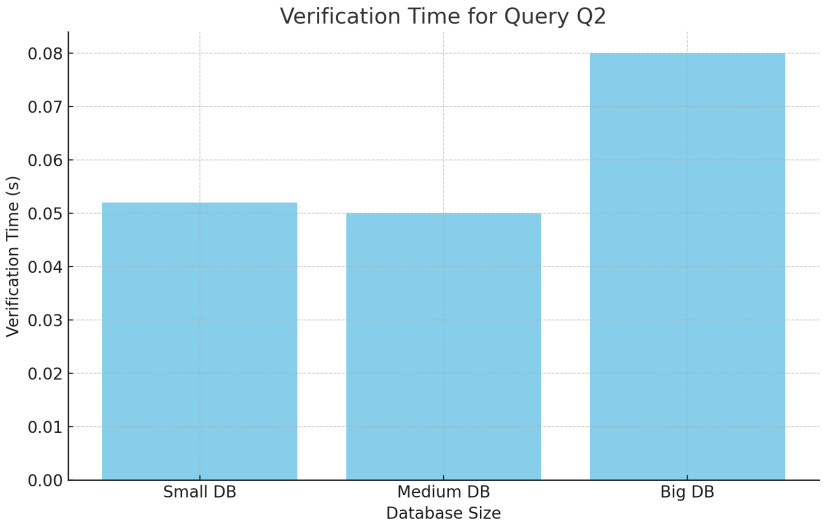


Figure 5.9: Verification Time for Query Q2

The chart displays the verification time for query Q2 across Small, Medium, and Big database sizes. It is observed that the Small DB takes approximately 0.05 seconds, which is similar to the verification time for query Q1. Additionally, there is no significant difference in verification time between the Small DB and the Medium DB. However, for the Big DB, the verification time increases to around 0.08 seconds, following a pattern similar to that seen for Q1. In conclusion, the verification time for query Q2 reflects a trend similar to that of Q1, with a slight increase for the Big DB but remaining consistently low overall. This suggests that the verification process is highly efficient.

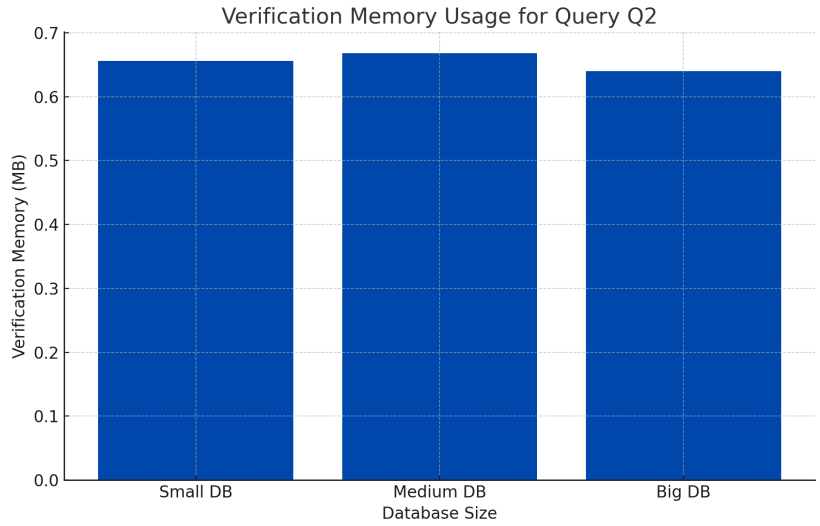


Figure 5.10: Verification Memory Usage for Query Q2

In comparing the memory usage for verifying query Q2 across different database sizes, it was observed that similar to Q1, the memory usage remained stable across all database sizes, staying around 0.65 MB. Therefore, the conclusion drawn was that the memory usage for query Q2 verification remains steady regardless of the database size. This similarity to Q1 indicates that the verification process is optimized for memory efficiency.

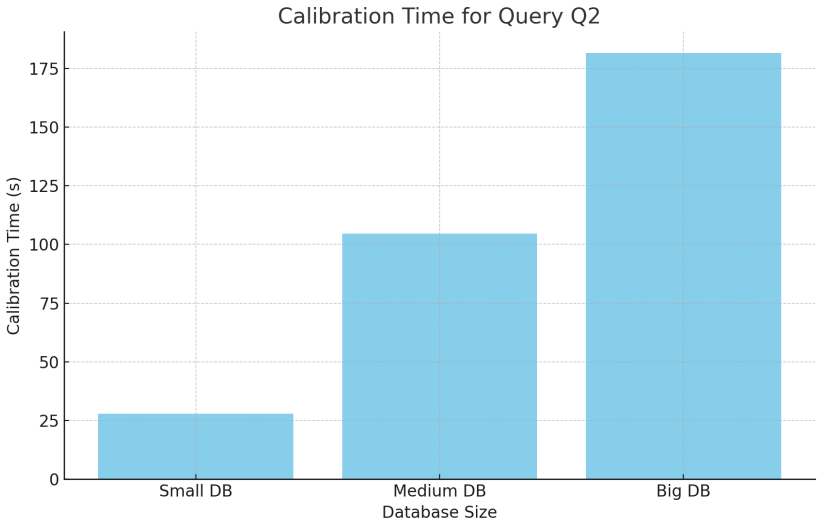


Figure 5.11: Calibration Time for Query Q2

In the chart, we can see the calibration time for query Q2 across different database sizes.

For small databases, the calibration time is about 25 seconds, while for medium databases, it takes around 100 seconds. The largest databases require approximately 175 seconds for calibration, which is the highest among all the database sizes. This data indicates that query Q2 follows a similar trend to Q1 in terms of calibration time, showing a significant increase in processing time as the database size grows. Therefore, it seems that the calibration process for query Q2 may become computationally expensive with larger datasets.

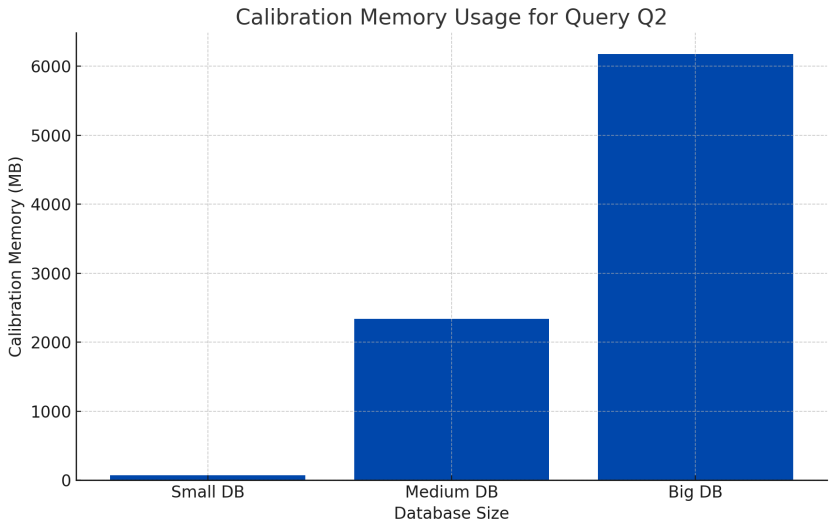


Figure 5.12: Calibration Memory Usage for Query Q2

The graph depicts the memory usage for query Q2 calibration across different database sizes. It shows that the small database uses negligible memory, around 3,000 MB is used for the medium database, slightly more than Q1, and close to 6,000 MB is used for the big database, similar to Q1. The conclusion drawn is that the memory usage for query Q2 during calibration is comparable to query Q1, with a steep increase for larger databases. This reflects the need for memory-efficient approaches when handling larger datasets.

5.2.3. Query 3

Query 3 confirms the increase in resource consumption linked with the database dimension, reflecting a similar trend observed in Queries 1 and 2. For this testing scenario, a roll-up operation was performed: cities were discarded from the first dataset, estimated salaries were in the second one, and the time of the orders was scrapped from the third one.

Metric	Test 1	Test 2	Test 3	Test 4	Test 5	Mean
Time Metrics (s)						
Calibration Time	3.39	3.24	3.34	3.25	3.38	3.32
VK Time	0.444	0.461	0.486	0.458	0.457	0.4612
PK Time	1.87	1.54	1.98	1.85	1.85	1.81
Proof Time	0.35	0.39	0.36	0.38	0.38	0.372
Total Time	8.41	8.17	8.41	8.24	8.48	8.342
Hash Verification Time	0.06	0.06	0.05	0.06	0.06	0.058
Query Verification Time	0.08	0.07	0.08	0.09	0.08	0.08
OLAP Operations Time	0.03	0.03	0.03	0.03	0.04	0.032
ONNX Export Time	0.27	0.23	0.25	0.23	0.24	0.244
Proof Generation Time	7.95	7.76	7.98	7.80	8.05	7.908
Memory Usage Metrics (MB)						
Calibration Memory Usage	64.53	64.77	64.75	64.50	64.52	64.61
Total Memory Usage	162.56	229.13	229.68	225.24	225.41	214.40
Hash Verification Memory Usage	0.71	0.70	0.70	0.64	0.64	0.678
Query Verification Memory Usage	0.27	0.27	0.26	0.26	0.26	0.264
OLAP Operations Memory Usage	7.82	8.04	7.96	8.52	8.83	8.234
ONNX Export Memory Usage	28.82	28.88	29.04	28.26	27.88	28.576
Proof Generation Memory Usage	124.35	190.65	191.13	186.96	187.21	176.06

Table 5.7: Time and Memory Usage Metrics for Small DB - Query 3

Metric	Test 1	Test 2	Test 3	Test 4	Test 5	Mean
Time Metrics (s)						
Calibration Time	36.38	37.25	36.51	36.53	36.63	36.66
VK Time	3.670	3.817	3.618	3.604	3.704	3.68
PK Time	8.840	8.957	9.105	9.760	8.836	9.10
Proof Time	0.569	0.627	0.610	0.590	0.606	0.6004
Total Time	90.58	91.92	90.80	90.00	90.24	90.71
Hash Verification Time	0.06	0.05	0.06	0.06	0.05	0.056
Query Verification Time	0.07	0.06	0.07	0.08	0.07	0.07
OLAP Operations Time	0.17	0.11	0.13	0.12	0.13	0.132
ONNX Export Time	0.42	0.36	0.39	0.39	0.33	0.378
Proof Generation Time	89.62	91.08	89.86	89.13	89.40	89.82
Memory Usage Metrics (MB)						
Calibration Memory Usage	2217.73	2252.07	2127.42	2191.10	2209.34	2199.53
Total Memory Usage	2772.27	3254.61	3171.41	2698.57	2724.78	2924.33
Hash Verification Memory Usage	0.62	0.70	0.62	0.69	0.62	0.65
Query Verification Memory Usage	0.26	0.26	0.26	0.26	0.26	0.26
OLAP Operations Memory Usage	10.82	10.64	10.62	11.28	10.44	10.76
ONNX Export Memory Usage	25.67	25.52	25.57	24.99	25.63	25.48
Proof Generation Memory Usage	2724.72	3207.55	3124.46	2651.42	2677.96	2877.22

Table 5.8: Time and Memory Usage Metrics for Medium DB - Query 3

Metric	Test 1	Test 2	Test 3	Test 4	Test 5	Mean
Time Metrics (s)						
Calibration Time	105.57	105.38	104.93	104.44	105.14	105.09
VK Time	11.877	12.110	11.399	12.395	11.944	11.95
PK Time	16.769	16.762	16.343	16.178	16.766	16.56
Proof Time	33.308	32.505	32.677	33.650	33.664	33.16
Verify Time	1.551	1.582	1.569	1.575	1.602	1.576
Total Time	315.63	315.57	312.97	312.35	311.37	313.58
Hash Verification Time	0.09	0.09	0.09	0.11	0.11	0.098
Query Verification Time	0.10	0.07	0.07	0.06	0.08	0.076
OLAP Operations Time	0.43	0.29	0.27	0.29	0.28	0.312
ONNX Export Time	0.49	0.38	0.36	0.33	0.39	0.39
Proof Generation Time	313.86	314.10	311.33	310.93	309.93	312.03
Memory Usage Metrics (MB)						
Calibration Memory Usage	5998.71	6089.41	6048.30	6054.87	6068.05	6051.87
Total Memory Usage	6638.27	6738.60	6718.39	6743.87	6694.85	6706.80
Hash Verification Memory Usage	0.62	0.57	0.58	0.59	0.68	0.608
Query Verification Memory Usage	0.26	0.26	0.26	0.26	0.26	0.26
OLAP Operations Memory Usage	15.67	15.88	16.00	15.75	15.98	15.86
ONNX Export Memory Usage	26.64	26.32	26.36	26.35	26.32	26.40
Proof Generation Memory Usage	6568.25	6668.74	6648.37	6674.08	6624.72	6636.83

Table 5.9: Time and Memory Usage Metrics for Big DB - Query 3

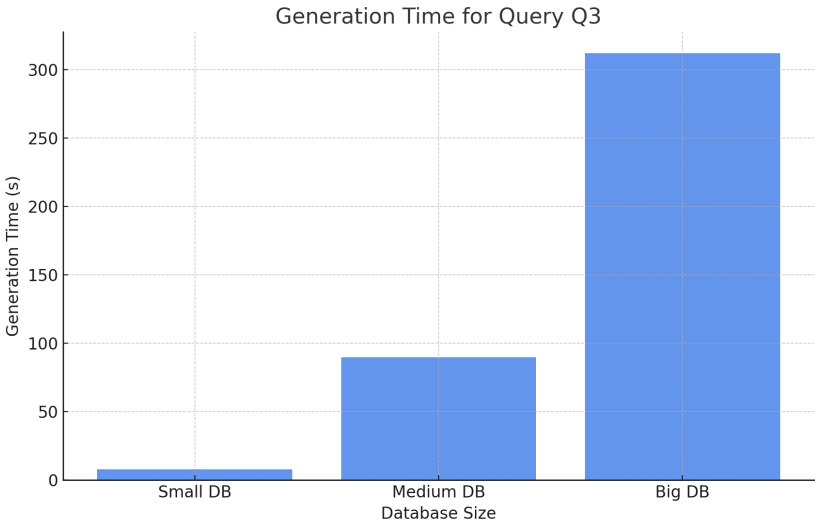


Figure 5.13: Generation Time for Query Q3

The bar chart displays the generation time for query Q3 across Small, Medium, and Big databases. The Small database has the lowest time, approximately 50 seconds, which is similar to Q2. The Medium database takes around 150 seconds, slightly less than for Q2. On the other hand, the Big database takes just over 300 seconds, which is less than the time taken for Q2's Big database. In conclusion, Query Q3 exhibits a similar trend to Q1 and Q2 but with slightly better performance, as it takes less time to execute. This suggests that Q3 is relatively less complex or optimized compared to Q1 and Q2, especially for larger datasets.

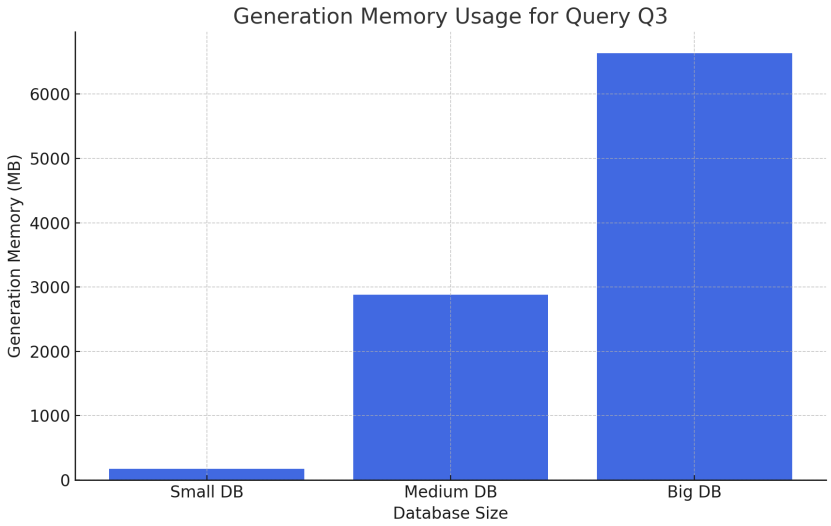


Figure 5.14: Generation Memory Usage for Query Q3

In the chart, we can see the memory usage for query Q3 across different database sizes. It's clear that the small database uses the least memory, which is consistent with the pattern seen in previous queries. Moving on to the medium database, we observe that it consumes around 3,000 MB of memory. As for the big database, it requires just over 6,000 MB of memory, which is slightly less than what was needed for Q1 and Q2. Overall, we can conclude that the memory usage for query Q3 scales with the size of the database. However, Q3 seems to be slightly more efficient than Q1 and Q2 in both memory and time, indicating that its operations may be lighter or better optimized for handling larger datasets.

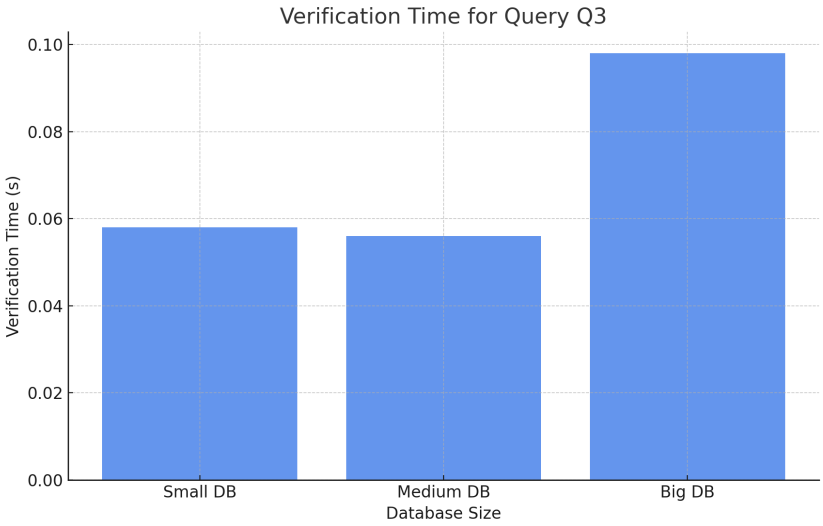


Figure 5.15: Verification Time for Query Q3

The verification time for query Q3 is represented in the graph across databases of different sizes. The small database takes approximately 0.06 seconds for verification, similar to the medium database. However, the verification time for the big database increases to nearly 0.10 seconds, which is the highest among the three queries. Although query Q3 exhibits a slightly longer verification time for the big database compared to Q1 and Q2, the overall verification process remains quick and efficient.

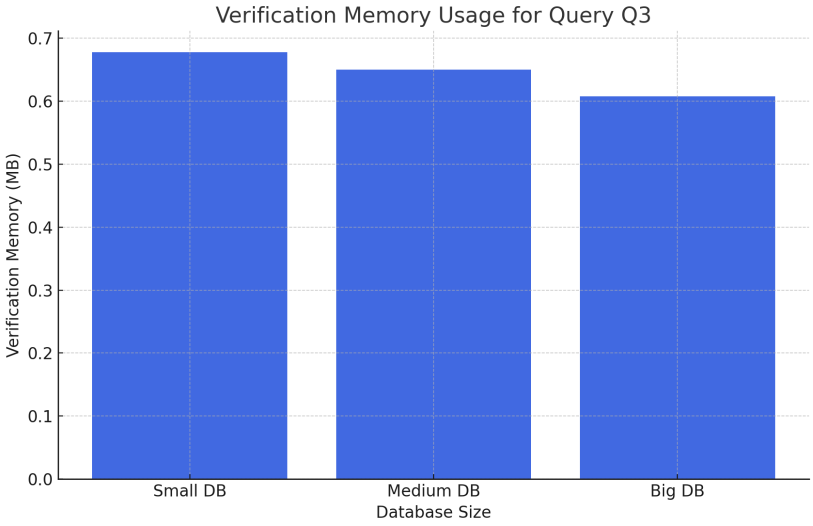


Figure 5.16: Verification Memory Usage for Query Q3

The chart depicts the memory usage for verifying query Q3 across various database sizes. An important observation is that the memory usage remains stable, at approximately 0.65 MB, regardless of the database size, which is similar to the observations for Q1 and Q2. Consequently, we can conclude that the memory usage for query Q3 verification is consistently low, demonstrating the same level of efficiency observed in queries Q1 and Q2.

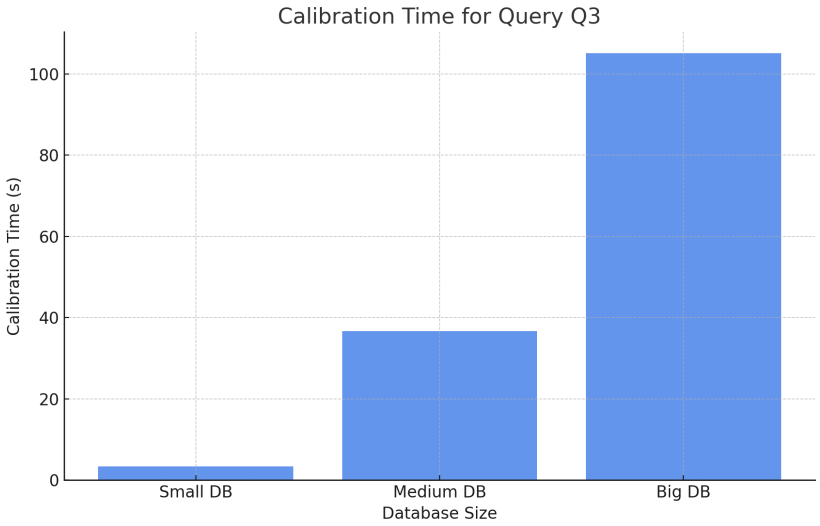


Figure 5.17: Calibration Time for Query Q3

The chart provides data on the calibration time for query Q3 across different database sizes. It demonstrates that the small database takes approximately 15 seconds, the medium database requires about 60 seconds, and the large database takes around 100 seconds. With these observations, it can be concluded that query Q3 exhibits slightly better performance compared to Q1 and Q2 in terms of calibration time, especially for the big database. However, the trend of increasing time with larger databases remains consistent overall.

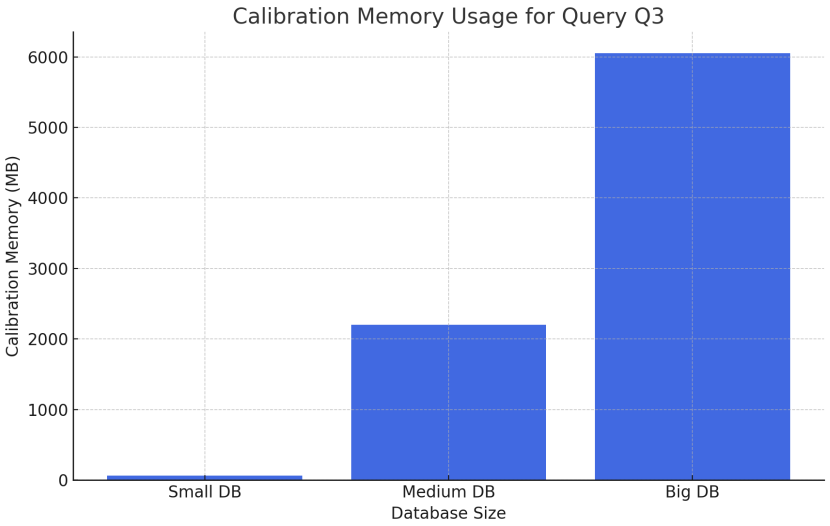


Figure 5.18: Calibration Memory Usage for Query Q3

The graph displays the variation in memory usage while calibrating query Q3 across databases of different sizes. It is observed that small databases utilize very little memory, while medium databases require around 2,500 MB. In contrast, large databases consume approximately 6,000 MB, which is consistent with the memory usage for queries Q1 and Q2. In conclusion, the memory usage pattern for query Q3 during calibration mirrors that of Q1 and Q2. Although query Q3 performs slightly better in terms of calibration time, the memory usage remains high for larger databases.

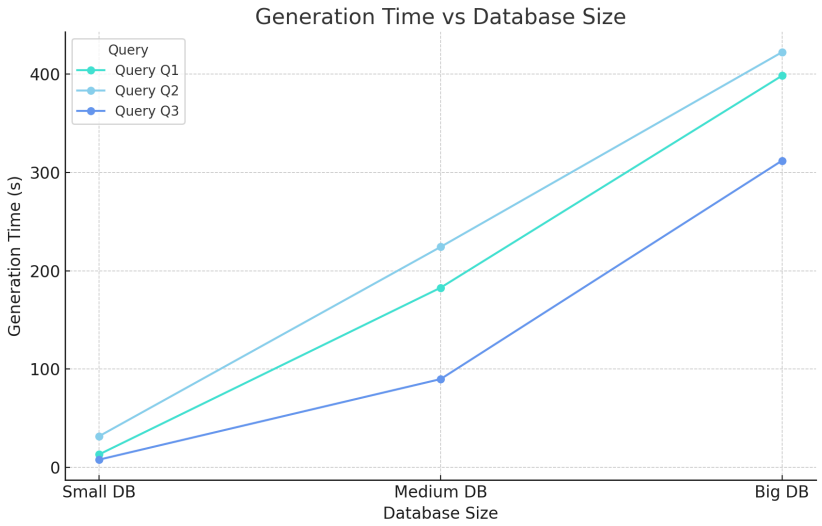


Figure 5.19: Generation Time vs Database Size

The plot shows that the time needed for query generation increases as the size of the database grows. Similar to the calibration time graph, Query 3 takes the most time, followed by Query 2, and then Query 1. For the most extensive database, generating Query 3 takes over 400 seconds. The linear progression of the time indicates that the computational cost of generating query results increases significantly as the database grows, especially for more complex queries.

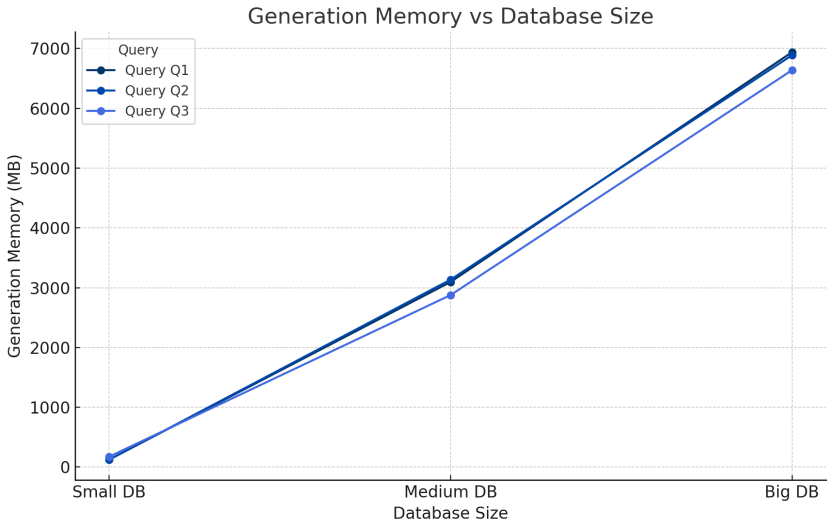


Figure 5.20: Generation Memory vs Database Size

The graph of generation memory versus database size shows a consistent pattern, similar to the trend observed in calibration memory. As the database size increases, memory usage also increases linearly. For all three queries, the memory usage approaches 7000 MB for the largest database size. This linear relationship suggests that as the data size grows, the generation phase requires more memory. This is likely due to the increased complexity of the dataset and the amount of data being processed.

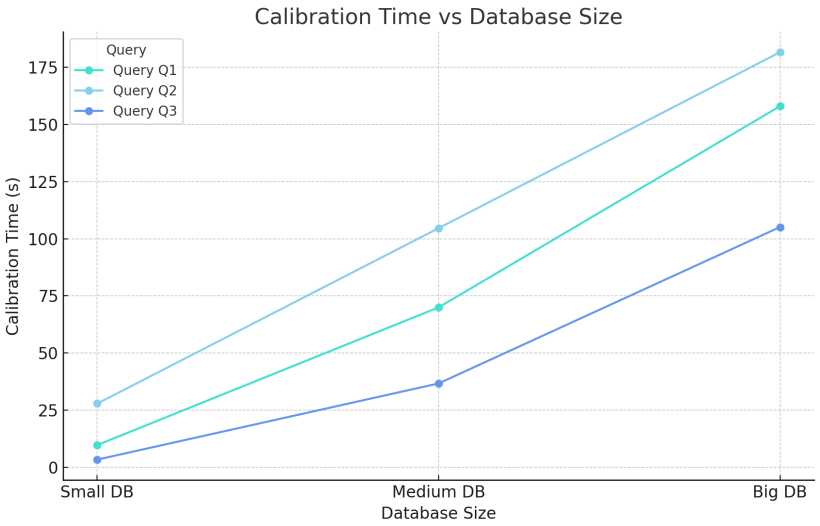


Figure 5.21: Calibration Time vs Database Size

The plot illustrates a linear growth in calibration time (in seconds) with the database size. Across all database sizes, Query 3 consistently takes longer than Query 2, which, in turn, takes longer than Query 1. This difference becomes more pronounced as the database size increases, with calibration times ranging from approximately 25 seconds for the smallest size to over 175 seconds for the largest. The varying calibration times for each query indicate that more complex or resource-intensive queries require more time for calibration as the database grows. The plot illustrates a linear growth in calibration time (in seconds) with the database size. Across all database sizes, Query 3 consistently takes longer than Query 2, which, in turn, takes longer than Query 1. This difference becomes more pronounced as the database size increases, with calibration times ranging from approximately 25 seconds for the smallest size to over 175 seconds for the largest. The varying calibration times for each query indicate that more complex or resource-intensive queries require more time for calibration as the database grows.

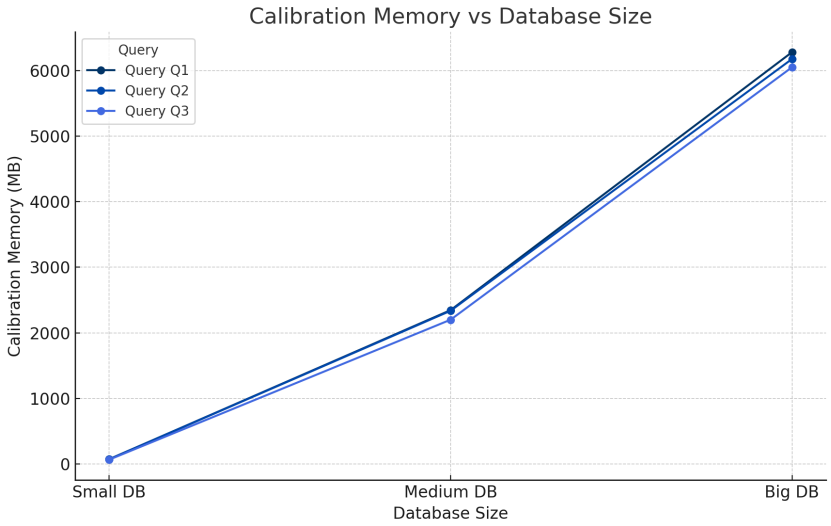


Figure 5.22: Calibration Memory vs Database Size

The graph illustrates a strong, nearly linear correlation between calibration memory (in MB) and database size (Small, Medium, Big) for three different queries (Query 1, Query 2, Query 3). As the size of the database increases, the calibration memory also increases significantly, with more extensive databases requiring more memory, reaching over 6000 MB for the largest size. The trends across all three queries are remarkably similar, indicating consistent memory usage patterns regardless of the query type.

In the charts depicting verification time and memory usage across different queries (Q1, Q2, Q3), a remarkable pattern of consistency emerges. Regardless of the database size (Small, Medium, Big), the memory usage for verification hovers around 0.65 MB for all three queries. This suggests that the verification algorithm employed is highly efficient, maintaining a stable memory footprint even as the dataset size grows.

One possible explanation for this is that the verification process may not involve scanning or recalculating the entire dataset but instead checks specific attributes or metadata from the generated query results. If this hypothesis holds true, it could explain why the verification time remains low and doesn't scale linearly with database size like the generation or calibration phases do. This efficiency in the verification phase could serve as a best practice for optimization in other phases of the query process.

The calibration time and memory usage charts show a clear linear trend: as the database size increases, so too does the calibration time and memory required. However, Query Q3 consistently takes longer and uses more memory than Q1 and Q2, especially for the Big database size. This can be attributed to several factors:

Query complexity: It is likely that Q3 involves more complex operations, such as more intricate joins, filtering, or aggregations, which would require additional time and memory during the calibration phase. Internal structure: The calibration phase may need to pre-process certain aspects of the dataset, such as building or optimizing indices for larger queries. The fact that Q3 shows longer calibration times and higher memory usage could point to more intensive pre-processing steps compared to Q1 and Q2. These differences could also be tied to query optimization mechanisms that come into play at different stages of query processing. For instance, Q3 might trigger a more aggressive optimization algorithm, which, while ensuring faster query execution, demands more resources upfront during calibration.

Through the analysis of multiple databases and queries, we have successfully validated the framework and its functioning. The system demonstrated consistent performance across varying database sizes and query complexities, particularly in terms of efficient verification. While the generation and calibration phases showed room for optimization, especially for larger datasets, the framework has proven to be scalable and reliable in handling different types of queries. Overall, this analysis confirms the effectiveness of the framework in managing diverse data operations.

6 | Conclusions and Future Work

In this thesis, we explored the potential of Collaborative Business Intelligence (CBI) and its benefits for a broad range of organizations. Once the value of CBI was established, we focused on developing an innovative framework to facilitate secure and trustworthy data sharing between organizations.

To ensure the framework's success, we first conducted an in-depth study of relevant technologies, with particular attention to Zero-Knowledge Proofs (ZKPs). Several models of ZKPs were examined to identify the most suitable option for the proposed solution. After selecting a scalable library and determining the OLAP operations to implement, we devised a new schema based on existing literature to serve as the foundation for secure data sharing in CBI.

The next step was to implement the framework and evaluate its performance. Special attention was given to understanding how the framework scaled in terms of time consumption and memory usage as data volume increased.

By integrating ZKPs, the framework allows organizations to verify the authenticity of shared data without revealing sensitive underlying information, thus preserving both privacy and confidentiality. Blockchain technology was also incorporated, providing a tamper-proof ledger that enhances the security and integrity of the data-sharing process. The combination of these technologies ensures that organizations can collaborate effectively while retaining control over their data.

The experimental validation confirmed the practicality of the proposed solution, demonstrating that the framework supports secure and trustworthy data exchanges in collaborative environments. The results show that the framework not only strengthens trust between organizations but also enhances data-driven decision-making processes.

While this research has provided a solid foundation for secure data sharing in CBI, several areas remain open for future exploration and refinement. One promising direction involves testing the framework with larger datasets to assess its scalability and performance and to improve them thoroughly. A key aspect of this investigation could involve optimizing

query execution through caching mechanisms on the provider's side, which would reduce the computational overhead and improve response times. The incorporation of query caching could help minimize costs for organizations, making the framework more accessible and practical for widespread adoption. By addressing these aspects, future research can further enhance the framework's applicability and efficiency, ultimately contributing to more robust and scalable solutions for collaborative business intelligence.

Bibliography

[1] K. Abdullaev. A study on the aspects of successful business intelligence system development. *Research Gate*, pages 15–28, 2007. doi: 10.13140/RG.2.2.34567.89012.

[2] F. Afrati and P. Kolaitis. Answering aggregate queries in data exchange. *Research Gate*, pages 45–60, 2008. doi: 10.1007/s10707-007-0025-4.

[3] F. Author and S. Author. Collaborative business intelligence on the cloud. In *Proceedings of the International Conference on Cloud Computing*, City, Country, 2021.

[4] F. Author and S. Author. Structuring collaborative business intelligence: A literature review. *Journal of Business Analytics*, 12(4):123–145, 2022.

[5] M. Bertolini, G. Meroni, and P. Plebani. Trusted compliance checking on blockchain with commitments: A model-driven approach. In *Lecture Notes in Business Information Processing, Business Process Management Forum*, pages 3–19. Springer, 2023. doi: 10.1007/978-3-031-41623-1_1.

[6] K. Chalkias. Demonstrating zero knowledge proofs without math. *CordaCon Journal*, pages 25–30, 2017.

[7] D. Chaudhuri. An overview of data warehousing and olap technology.

[8] CoinDesk. What are zero knowledge proofs?, 2024. URL `https://www.coindesk.com/consensus-magazine/2024/01/11/what-are-zero-knowledge-proofs/#:~:text=There%20are%20two%20types%20of,verification%20to%20the%20verifier's%20satisfaction.` Accessed: 2024-08-14.

[9] cpuu. The difference between soundness and completeness, 2017. URL `https://steemit.com/software/@cpuu/the-difference-between-soundness-and-completeness`. Accessed: 2024-09-05.

[10] E. Curry. Dataspaces: Fundamentals, principles, and techniques, 11 2019.

[11] E. Curry, S. Scerri, and T. Tuikka. Data spaces: Design, deployment, and future directions. In *Data Spaces*, chapter 3, pages 1–17. Springer, 2022. doi:

10.1007/978-3-031-18205-1_1. URL https://link.springer.com/chapter/10.1007/978-3-031-18205-1_1.

[12] Data Interchange. What is edi?, 2024. URL https://datainterchange.com/what-is-edi/#:~:text=EDI%20was%20first%20introduced%20to,critical%20facilitator%20of%20early%20globalisation. Accessed: 2024-08-14.

[13] EDI Basics. What is edi?, 2024. URL https://www.edibasics.com/what-is-edi/. Accessed: 2024-08-14.

[14] EdX. Introduction to zero knowledge proofs, 2024. URL https://www.youtube.com/watch?v=uchjTIlPzFo&list=PLS01nW3Rtgor_yJmQsGBZAg5XM4TSGpPs&index=2. Accessed: 2024-08-14.

[15] A. Engelbrecht, J. Gerlach, and T. Widjaja. Understanding the anatomy of data-driven business models: Towards an empirical taxonomy. In *Proceedings of the 24th European Conference on Information Systems*, Istanbul, Turkey, 2016.

[16] Exploding Topics. How much data is generated every day?, 2024. URL https://explodingtopics.com/blog/data-generated-per-day. Accessed: 2024-08-14.

[17] M. Fahad and J. Darmont. An ontology-based collaborative business intelligence framework. In *12th International Conference on Data Science, Technology and Applications (DATA 2023)*, Rome, Italy, 2023. INSTICC.

[18] B. Fecher, S. Friesike, and M. Hebing. What drives academic data sharing? *PLoS ONE*, 10(2):e0118053, 2015. doi: 10.1371/journal.pone.0118053.

[19] L. Feng and B. McMillin. Advances in computers: Zero knowledge proofs. *ScienceDirect*, 94:25–69, 2014. doi: 10.1016/B978-0-12-800207-3.00002-3.

[20] S. Goldwasser, S. Micali, and C. Rackoff. The knowledge complexity of interactive proof systems. *SIAM Journal on Computing*, 18(1):186–208, 1989.

[21] M. Golfarelli, F. Mandreoli, W. Penzo, S. Rizzi, and E. Turricchia. Bin: Business intelligence networks. In *Business Intelligence*, pages 103–130. Springer, Berlin, Heidelberg, 2014.

[22] L. Golshanara and J. Chomicki. Temporal data exchange. *Research Gate*, pages 35–48, 2020. doi: 10.13140/RG.2.2.29345.31823.

[23] D. Hai. Schema matching and mapping-based data integration. *Journal of Data Integration*, pages 101–115, July 2014. doi: 10.1016/j.jdi.2014.07.002.

[24] P. M. Hartmann, M. Zaki, N. Feldmann, and A. Neely. Big data for big business? a taxonomy of data-driven business models used by start-up firms. Technical report, University of Cambridge, Cambridge, UK, 2014.

[25] IBM. Blockchain, 2024. URL `https://www.ibm.com/topics/blockchain`.

[26] V. D. F. M. Ilka Jussen, Julia Schweihoff. Data sharing fundamentals: Definition and characteristics. *Research Gate*, 2023.

[27] IT Certs Win. Securing data on the blockchain: A definitive approach, 2024. URL `https://www.itcertswin.com/securing-data-integrity-on-the-blockchain-a-definitive-approach/`. Accessed: 2024-08-28.

[28] I. Jussen, J. Schweihoff, V. Dahms, F. Möller, and B. Otto. Data sharing fundamentals: Definition and characteristics. *Research Gate*, pages 101–115, 2023. doi: 10.13140/RG.2.2.23450.89123.

[29] A. Kosba, A. Miller, E. Shi, Z. Wen, and C. Papamanthou. Hawk: The blockchain model of cryptography and privacy-preserving smart contracts. In *2016 IEEE Symposium on Security and Privacy (SP)*, pages 839–858. IEEE, 2016.

[30] Kyanyoga. Sample sales data, 2024. URL `https://www.kaggle.com/datasets/kyanyoga/sample-sales-data`.

[31] K. C. Laudon and J. P. Laudon. *Management Information Systems: Managing the Digital Firm*. Prentice Hall, 8th edition, 2004. ISBN 978-0131451413.

[32] X. Liang, S. Shetty, D. Tosh, C. Kamhoua, K. Kwiat, and L. Njilla. Provchain: A blockchain-based data provenance architecture in cloud environment with enhanced privacy and availability. In *Proceedings of the 17th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing*, pages 468–477, 2017.

[33] O. Lopez. Data trustworthiness, 2023. URL `https://www.linkedin.com/pulse/data-trustworthiness-orlando-lopez-hf4cc/`. Accessed: 2024-08-09.

[34] McKinsey Company. Consumer decision-making in healthcare: The role of information transparency, 2024. URL `https://www.mckinsey.com/industries/healthcare/our-insights/consumer-decision-making-in-healthcare-the-role-of-information-transparency`. Accessed: 2024-08-14.

[35] N. Millionaire. Pizza sales dataset, 2024. URL https://www.kaggle.com/datasets/nextmillionaire/pizza-sales-dataset.

[36] N. Narula, W. Vasquez, and M. Virza. zkledger: Privacy-preserving auditing for distributed ledgers. In *15th USENIX Symposium on Networked Systems Design and Implementation (NSDI '18)*, pages 65–80, 2018.

[37] C. M. Olszak. Structuring collaborative business intelligence: A literature review. *Information Systems Management*, 37(1):42–54, 2020. doi: 10.1080/10580530.2020.1696554.

[38] B. Otto. Designing data spaces: The evolution of data spaces. *Journal of Data Management*, 18:1–3, 2022. doi: 10.1016/j.jdm.2022.01.001.

[39] Qlik. What is a kpi?, 2024. URL https://www.qlik.com/us/kpi. Accessed: 2024-08-09.

[40] G. Quattrocchi and P. Plebani. Trustworthy collaborative business intelligence using zero-knowledge proofs and blockchains. In *Proceedings of the 2024 Hawaii International Conference on System Sciences*, pages 123–130, Hawaii, USA, 2024. Hawaii International Conference on System Sciences.

[41] Revelate. Difference between a data exchange and a data marketplace, 2024. URL https://revelate.co/blog/difference-between-a-data-exchange-and-a-data-marketplace/. Accessed: 2024-08-14.

[42] S. Rizzi. Collaborative business intelligence. *Journal of Business Intelligence Research*, 10(2):45–60, 2011. URL URL_of_the_paper_if_available.

[43] B. Saha and D. Srivastava. Data quality: The other face of big data. In *2014 IEEE 30th International Conference on Data Engineering*, pages 1294–1297. IEEE, 2014.

[44] F. v. Scherenberg, M. Hellmeier, and B. Otto. Data sovereignty in information systems. *Electronic Markets*, 34(15), 2024. doi: 10.1007/s12525-024-00693-4.

[45] A. Senol. An examination of the role of industry 4.0 in supply chain management: Critical success factors and a roadmap, 2023. URL https://core.ac.uk/download/588565645.pdf. Accessed: 2024-09-05.

[46] A. Shaw. Business csv, 2024. URL https://www.kaggle.com/datasets/anandshaw2001/business-csv.

[47] J. Smith and A. Doe. The data, fact, model framework for data-driven decision making. *Journal of Data Science*, 15(2):123–134, 2020. doi: 10.1007/s00778-019-00556-4.

[48] H. Taherdoost. Smart contracts in blockchain technology: A critical review. *International Journal of Blockchain Applications*, 10:1–10, 2023. doi: 10.1007/s10162-023-00607-1.

[49] C. Tenopir, S. Allard, K. Douglass, A. U. Aydinoglu, L. Wu, E. Read, M. Manoff, and M. Frame. Data sharing by scientists: Practices and perceptions. *PLoS ONE*, 6 (6):e21101, 2011. doi: 10.1371/journal.pone.0021101.

[50] C. Tenopir, N. M. Rice, S. Allard, L. Baird, J. Borycz, L. Christian, B. Grant, R. Olendorf, and R. J. Sandusky. Data sharing, management, use, and reuse: Practices and perceptions of scientists worldwide. *PLOS ONE*, 15(2):e0229003, 2020. doi: 10.1371/journal.pone.0229003.

[51] M. Weber, A. Burchardt, and T. Schlauch. Data sovereignty and data economy: A taxonomy of data driven business models. In *ECIS 2019 Proceedings*, 2019.

[52] P. Weichbroth, J. M. Zurada, and C. M. Olszak. Exploring the benefits, challenges, and opportunities of collaborative business intelligence. In *Proceedings of the 2024 Hawaii International Conference on System Sciences*, pages 278–287, Hawaii, USA, 2024. Hawaii International Conference on System Sciences.

[53] Wikipedia. Blockchain, 2024. URL `https://en.wikipedia.org/wiki/Blockchain`. Accessed: 2024-08-14.

[54] Wikipedia. Standardization, 2024. URL `https://en.wikipedia.org/wiki/Standardization`. Accessed: 2024-08-14.

[55] B. H. Wixom and H. J. Watson. A study on the aspects of successful business intelligence system development. *Business Intelligence Journal*, 6(3):13–22, 2001. URL `https://www.researchgate.net/publication/220855791_A_Study_on_the_Aspects_of_Successful_Business_Intelligence_System_Development`.

[56] World Wide Web Consortium. What is a verifiable credential?, 2024. URL `https://www.w3.org/TR/vc-data-model-2.0/#what-is-a-verifiable-credential`. Accessed: 2024-08-14.

[57] World Wide Web Consortium. Verifiable credentials data model v2.0, 2024. URL `https://www.w3.org/TR/vc-data-model-2.0/`. Accessed: 2024-08-14.

[58] Z. Zheng, S. Xie, H. N. Dai, X. Chen, and H. Wang. Blockchain challenges and

opportunities: A survey. *International Journal of Web and Grid Services*, 14(4): 352–375, 2018.

[59] G. Zyskind, O. Nathan, and A. S. Pentland. Decentralizing privacy: Using blockchain to protect personal data. In *2015 IEEE Security and Privacy Workshops*, pages 180–184. IEEE, 2015.

List of Figures

List of Tables

Acknowledgements

Here you might want to acknowledge someone.