

Original Article

STRIDE-DB: A Structured, Risk-Aware and Intelligent Framework for Legacy Database Transformation in Hybrid Cloud Environments

N. Siva Kumar¹, Suneel Pappala², Sripada Rama Sree³

^{1,2}Department of CSE, University of Technology, Vatika, Jaipur, India.

^{1,3}Department of CSE, Aditya University, Andhra Pradesh, India.

¹Corresponding Author: siva733.uet@gmail.com

Received: 12 August 2026

Revised: 15 September 2026

Accepted: 24 September 2026

Published: 28 October 2026

Abstract - Legacy database systems are still being required to support mission-critical applications, which need to be scalable, interoperable, resilient, governed, cloud-operated and more. The problem with modernizing these systems is that transforming the data from a database is not only hard to do because the data is much more than just schemas and data transfer, but also because of structural dependency, semantic correspondence, data quality, complexity of the transform, migration risk, resources required, deployment constraints, etc. Most of the current solutions address this issue individually, and not as part of a migration decisioning process. In this paper, the authors propose a hybrid cloud-based transformation framework for the evolution of the intelligent legacy database, called as Structured Transformation and Risk-aware Intelligent Database Evolution (STRIDE-DB). The intelligent cloud recommendation, automatic transformation and migration planning, validation tool, explainable AI, machine learning risk prediction, ensemble learning, business rule integrity, data quality, dependency discovery and verification, and metadata discovery are all features of the STRIDE-DB architecture. The following tests (BaseLine, SemanticStress, DataQualityStress, DependencyStress, CombinedStress and optimized migration) are run on a controlled experimental layer. The current implementation was tested with seven source tables with about 59825 records and 38 attributes of a relational database. The knowledge representation layer created 49 nodes and 43 edges, while the graph representation learning created 64-dimensional node embeddings. The pre-migration risk indicator was determined to be 0.1717 (low risk), and the transformation-complexity indicator was determined to be 0.2184 (low complexity). Out of 38 attribute mappings, 35 were chosen to be automatically transformed, and three were kept for further review due to a lack of confidence. The average target attribute coverage was 94.05% while the average target completeness was 99.66%. Furthermore, the framework offers a number of machine learning classifiers, multiple classifiers, ensemble learning, explanation using SHAP, auto-generated SQL and SQL-recommended and explanation for the hybrid cloud. The results show that the concept of integrated pipeline STRIDE-DB is viable for database modernization for structured and risk-aware databases. It is a small table-level data set, however, and is just a feasibility examination of the implementation stage, as a substitute for a generalization skill in the machine-learning evaluation. More large-scale, multi-dataset and multi-scenario evaluations to prove the predictive models are valid are required.

Keywords - Database modernization, Database transformation, Hybrid cloud, Migration risk assessment, Knowledge graph, Semantic schema mapping, Graph representation learning, Machine Learning, Explainable Artificial Intelligence, Intelligent migration planning.

1. Introduction

As these information systems and data-driven enterprise systems are being utilized, these systems are turning out to be much more important alternatives to traditional information database systems, particularly once cloud computing starts to achieve popularity. However, in a few mission-critical applications, there are other legacy databases too that hold data related to the business and operations that have been collected. When the legacy databases are deployed into the cloud environment, migrating to modernize the legacy

database can get complex, as the following could exist: implicit dependency, differences in schema, inconsistent naming, embedded business rules, inconsistent data quality of the legacy databases, and semantic incompleteness/differences between the legacy databases and the target databases' technologies. So, database modernization should involve more than simply moving data and migrating schemas. This is a combination of structure analysis, semantics, dependency preservation, migration risk analysis and migration planning and post-migration validation [1].



There are a variety of ways to migrate data to a common database, including schema conversion, data replication, Extract – Transform – Load (ETL) pipelines, and even transferring data. The following are reasons why these methods do not work well for multiple relationships between the source data elements and target data elements, for large amounts of source data, ambiguous correspondence between attributes in the source and target databases, or when the source data types are heterogeneous or when business rules are stored in the source database. The correctness and order of the transformation operations can be influenced by structural dependencies, and there can be semantic inconsistencies because of the difference between the concepts in the source and target languages. Depending on the application, the size of schemas and data volume may be too limited to capture the criticality of the dependency, the mapping uncertainty, data quality and/or the complexity of the data transformations to apply to the data. Do not migrate based on the number of records in the DB and/or size of structures; do it based on your migration plan.

There is one other decision factor to consider in the era of the hybrid cloud: When considering a migration process, consider not only if the legacy database can be transformed, as well as if the workload created is appropriate to a particular deployment environment. The characteristics of the workload, deployment constraints, resource features, risk of moving workloads to the cloud, and the complexity of the transformation process and importance of the scalability requirements could all impact the suitability of the public cloud, private cloud or hybrid (public cloud and private cloud) environment. Based on the above, there should be an overall model developed which takes into consideration the properties of the databases, and helps with the selection of migration, risk indicator and deployment advice.

Towards this, a structured and risk-aware approach to convert legacy databases to a hybrid cloud (STRIDE-DB) environment that can address some of the above issues is proposed. This involves discovery of metadata related to the structure, visualization of schema, dependency analysis, business-rule and integrity analysis, data-quality assessment, semantic schema mapping, building and analysing a knowledge graph, complexity estimation of the transformation, migration risk assessment, analysis of the compatibility of the hybrid cloud, migration planning, automated transformation, post-migration validation, estimation of migration risks with the help of machine learning, ensemble learning and explainable AI. There are two types of risk indicators, one conceptual and one machine learning and risk indicators can reliably predict the risks. Confidence-based mapping can also be used to store correspondences with low confidence, which can be mapped to a semantic uncertainty and can keep low confidence correspondences for future use. Moreover, there are cases of controlled migration which are applied to test the framework

in the following baseline, semantic-stress, data-quality-stress, dependency-stress, combined-stress and optimized migration cases.

Typically, dependency analysis, data-quality assessment and schema migration planning are done separately from the cloud and machine learning-based decision support - it may be possible to do these in combination. There is still a need for a consistent migration process to ensure there is a link between structural dependencies, semantic correspondence, data quality, complexity of the migration, risk of migration, and the suitability of the hybrid cloud – as well as post-migration validation. It is emphasized that, although a schema conversion decision is technically correct, semantically, a schema conversion decision may be influenced by semantic uncertainty and dependency criticality. The problem tackled in this research is that of developing an integrated framework which can analyse the characteristics of the legacy databases, conclude on the relationships between structures and semantics, estimate the complexity of transformation and the risk of migration of the legacy databases to a hybrid-cloud environment, enable decision making about the deployment of a hybrid-cloud application, perform a controlled transformation for a hybrid-cloud environment and validate the transformed database. These are the requirements STRIDE-DB has been designed for and has a migration lifecycle built into it.

1.1. Background

Legacy database modernization is the process of converting an existing database to a target database with the same structure, semantics, dependences, business rules, data quality and more. Data transfer is not as straightforward as it might appear, as the schema, attribute names, data types, relationships, constraints and business requirements of legacy and target databases may vary. So, the process of structural and dependency analysis is an important process to understand the relationships between tables, attributes, primary keys, foreign keys and other objects in the database. A schema dependency graph, which can be used for dependency analysis and assessment of criticality for migration, is validated by dependencies represented as a schema dependency graph.

The other main challenge is the same name of attributes for distinct concepts and the other way round. STRIDE-DB transforms the data between sources and targets by using multiple evidence sources: name similarity, semantic similarity, datatype compatibility, value profiles and relationship compatibility. The high-confidence mappings are automatically made, and ambiguous mappings are retained for review.

Several factors influence the complexity of transformation and the risk of migration, such as the dependencies, semantic ambiguity, datatype conversions, data quality, data volume and the requirements of migration. These

aspects are therefore taken into account in the transformation-complexity and pre-migration risk assessment that is carried out by STRIDE-DB. The engineered risk indicator is not used in predicting the next step in the machine learning process. Lastly, scalability, workload characteristics, the complexity of the transformation, the risk of migration, resource needs and deployment constraints must be taken into account in hybrid-cloud deployment. All these factors are built into STRIDE-DB's hybrid-cloud compatibility analysis and migration planning. Therefore, STRIDE-DB integrates the structural analysis, dependency analysis, semantic mapping, transformation complexity, migration risk, hybrid-cloud suitability and validation in one database modernization lifecycle.

1.2. Research Gap and Motivation

While much work has been done in the process of moving databases, transforming schemas, moving to the cloud [4], creating knowledge graphs and more recently, machine learning-based decision support, there are a number of important issues to address.

During the process of this research, the following research gaps were identified and summarised as follows:

1. Fragmented migration analysis: One of these activities is typically included in most of the database modernization solutions, but not all of them are connected in a single database modernization process, like schema conversion, data movement and dependency analysis.
2. Developmentally immature intelligences: structurally and semantically. It is not necessary that two sentences have the same structure for them to be semantically equivalent. The migration framework needs to have dependency-aware structural analysis and confidence-aware semantic mapping features.
3. Wrong use of uncertainty of migration: Automatic transformations for migration might be required for the default transformation pipeline. This can result in semantic errors being passed on to the target environment. However, with STRIDE-DB, we cannot afford to sacrifice any mapping and any reviewing.
4. Transformation plan –minimum amount of risk is put in place: The problems of complexity of transformation and migration risk are often regarded as separate operational problems. These indicators need to be carried forward in a coordinated manner in the migration planning.
5. Not good knowledge - from a graph: In relational databases, there are interconnected relationships among entities, but not necessarily will relational migration pipelines fully explore the graph representation to get into the criticality and dependency relationships of entities.
6. Lack of explanation of making smart migration decisions: Adding risk prediction (which can be done using machine learning) can increase the automation, although this might be hard to understand for migration engineers. It would be great if you could integrate explainability and include

it as part of the intelligent pipeline of decision support; STRIDE-DB would appreciate it[3].

7. No relationship between migration and clouds' location: It is also a consideration whether workloads will be suitable for a particular deployment environment, and the impact on the workload when deciding to make a transformation.
8. Ecologically constrained controlled assessment of the migration stress: Systematic assessments of the migration frameworks under the evaluation conditions should be carried out under baseline conditions, semantic ambiguity condition, degradation of data quality, dependency stress and combined condition.

STRIDE-DB's motivation is to progress beyond:

The traditional way of Migration: Schema Conversion + Data Transfer.

towards:

Intelligent Migration has the following main concepts: Structure + Dependencies + Semantics + Business Rules + Quality + Complexity + Risk + Cloud Suitability + Validation.

The vision observed is integrated; this will be the yardstick of the proposed framework, STRIDE-DB.

1.3. Main Contributions

The following are the key contributions of this research: Designed/Developed a structured, risk-aware and intelligent approach for a hybrid cloud environment – STRIDE-DB for legacy (RDB) database transformation. There is no demarcation between the schema conversion, dependency analysis and dependency risk assessment in this case; these are all a part of a lifecycle, unlike Stride-db. Following the extensions, the framework is proposed to allow for the discovery of schema (including more metadata), generation of a dependency graph and translation of semantics with a degree of confidence, for reliable schema translation. It also enables you to conduct business rule (Business integrity) analysis, or assign semantics to the Business operations during migration. Provides a Novel Risk-Aware mechanism, embedded, enhances the transparency and reliability and segregates away pre-migration Risk indicators along with the machine learning-based risk prediction. In addition, graph representation learning algorithms are also leveraged to represent structures and dependencies of relationships within STRIDE-DB to enable smart migration planning, as well as incorporating hybrid cloud systems. Moreover, the use of Explainable AI (XAI) features allows a meaningful understanding and interpretation of the risks recognised and options considered in the process of transformation, not only making the framework technically effective, but also readily available and accessible. STRIDE-DB showcases its ability to support high target-attribute coverage, low level of data transformation complexity, and low risk of migration in all scenarios: baseline, semantic-stress, data-quality-stress,

dependency-stress, and combined-stress to ensure it is truly an equal and complete solution for database modernization.

1.4. Paper Organization

In this paper, the rest of the paper will be arranged in this manner. Section 2 covers background knowledge and related Research: Schema Transformation, Dependency-aware Migration, Knowledge-Graph based Analysis, Hybrid Cloud deployment, Migration risk Assessment and Explainable Artificial Intelligence (XAI) for Intelligent migration frameworks. The latter will be further discussed in a subsequent part of the book – namely: Background knowledge and related research: Schema Transformation, Dependency-aware Migration, Knowledge-graph based Analysis, Hybrid Cloud Deployment, Migration risk Assessment and Machine Learning and Explainable Artificial Intelligence (XAI) for Intelligent migration frameworks.

In section 3, a problem formulation is carried out, and a description of the STRIDE-DB system modelling is done that includes modelling structural representation of systems, dependency graph and semantic mapping of those systems and a complexity estimation for transformation is done, then modelling of migration risk is done, and analysing the compatibility of hybrid cloud is carried out. The process starts with the description of the dataset, discovery of the metadata, structure pre-processing and structural characterisation, and is described in Section 4. Next, you will get to know the architecture of the STRIDE-DB: some large components, namely dependency analysis, dependency knowledge graph, semantic mapping, risk assessment and smart migration plan.

The project's focus is on section 6, including a machine-learning ensemble learning model for the prediction of migration risk and using explainable artificial intelligence to explain the migration risk results obtained by the machine-learning model. Ask your BMC Help Desk who is hosting your instance, for details of the automated transformation pipeline that you have configured to read – SQL generation, execution and SQL validation on migration. The experimental setup design is detailed in Section 8, as well as the scenarios of controlled migration to be tested and evaluated, metrics that can be used to measure the migration, and detailed configuration aspects. The results of these experiments, covering assessment of semantic stress, transformation performance, risk assessment and accuracy and validation, are summarised in Section 9. The results and the implications for practicability are mentioned shortly in Section 10, and limitations and threats to the validity are provided in Section 11. Ideas for future research are provided in the same section. Our last step was to provide a list of findings and what the paper can offer in section 12.

2. Background Work

Database modernization and migration have taken a step from schema-conversion to intelligent, semantic and risk-

aware transformation frameworks. In the past few years, different research works on schema evolution, automated schema matching, semantic correspondence, migration based on machine learning, dependency analysis, data quality assessment, cloud migration planning, etc., have been carried out. They are largely investigated independently, without taking into account the semantic uncertainty of the transformation, its dependency criticality and complexity, as well as the possibility of migration in the same pipeline. It thus reviews existing works on intelligent schema matching, evolution of databases, dependency-aware migration, data quality and cloud migration and identifies the existing research gap, which motivates the proposed STRIDE-DB framework.

2.1. Evolution of Intelligent Database Migration and Data Integration

Based on the results of this study, it can be hypothesized that in the future, schema conversion will move towards knowledge-driven and adaptive conversion, and towards intelligent conversion. B. Abu-Salih et al. envisioned several different technologies to provide live migration in 2026 and stressed that the continuity of the services during a live migration of “a computational entity” from one environment to another would be necessary in 2026. Only migration, not just a data transfer operation, an operationally constrained migration should be considered [7]. For iterative relation extraction with a large language model, knowledge graph and semantic clustering, Hakju Oh et al. [8] proposes DSE-RE, which includes a knowledge graph and dynamic schema to support and facilitate the process. They found that the dynamic change of semantic structures might be useful to enhance relation extraction in dynamic environments, and schema drift and inconsistency of LLM's are issues that need to be addressed.

N.R. Palnati et al. [9] In this case study, the system was experimented with numerous times, taking it through the various phases of the system, from a relational system to a NoSQL system, SkyServer. They have demonstrated that a favorable schema for a set of workloads for a particular domain for a certain time period is not optimal when the workload (or the domain) evolves over time, and that it's expensive and time-consuming to transfer data. Another step forward in the investigation of database-evolution was taken by I.M. Putrama and P. Martinek [10] Braghetto, who went from structural to semantic evolution, as they introduced mechanisms to address evolving semantics, such as translation, grouping, ungrouping, preprocessing and query-rewriting. All research shows that the evolution of the database should be a continuous change and structural change, not a transformation only (which can be realized at the semantic level).

This is clearly reflected in the new publications too, which are slowly progressing towards automation. Nishan

Mills et al. [11] suggested to create an executable migration schema from the schema model, design a schema comparison and design a schema reverse engineering schema from the database. In some cases, there are changes that can be avoided using a model-driven approach, but there are other cases where the change is semantically ambiguous and needs to be reviewed and understood by a human, who must understand the domain knowledge. So far, Maldonado Carrascosa et al. [12] have developed a new tool to extract data from a column family DB, which can be called over time – MoDEvo. They are based on the model approach to migrate the migration script, and are concerned about the consistency of the data in the denormalized (NoSQL) system. Inspired by the fact that, in the wild, Large-scale tabular schema matching (SMUTF) is a very challenging task, H. Foidl, et al. [13] had an idea to use generative language models as a fusion of the rule-based features with the Pretrained language models. The above developments, however, proved the knowledge of the process of migration studies, model-driven transformation of migration studies, and semantically matching. M. Asif-Ur-Rahman et al. [14] had suggested to optimally allocate the resources in the databases in the Cloud environment with an intelligent prediction using machine learning. The suggestions provided by R. Nasfi et al. [15] for the cost estimation and optimize the Cloud migration plan for the web applications migration in the cloud using the results from the quantitative Decision Support Model (DCM) were suggested.

2.2. Schema Evolution and Data Migration

Schema evolution is one of the challenges faced during a database modernization project. If the schema of the database needs to change, the change has to be incorporated into the data in the database. P. Suárez-Otero, et al. [16] have introduced the first quantitative migration-strategy framework for NoSQL document databases that takes into account schema changes, database size, search requirements and latency to choose the migration-strategies. Their results show evidence that the type of migration strategy used is dependent on the characteristics of the migration scenario, and that a single global migration mechanism will not be appropriate.

In standard lifecycle management, Zhang, Y. et al. [17] did a study on word embeddings, semantic schema matching. The following paper draws attention to the semantic integration in the following cases: two or more information systems have the same concept, but different representations. When deploying ML systems in the field, it is critical that the decision-making process is transparent, as M. H. Hasan et al. [18] pointed out the complexities and problems of explaining complex predictive models to create such models that are interpretable.

To overcome the accuracy–interpretability dilemma, C. Daraio and Li have introduced a new model, called SEE-Net, which evolves into a balance between the accuracy of neural networks and the interpretability [19]. Thus, the paradigm of

the research topic of migration has shifted from schema and data transformation to semantic correspondences and model interpretability and intelligent decision support as observed in the literature.

2.3. Schema Matching and Semantic Correspondence

A problem that arises in heterogeneous database integration is the discovery of concepts with similar meanings, but different attribute names, structures, and/or representations of data, which is called schema matching. Rodrigues et al. [20] systematically reviewed the approaches in the legacy to cloud migration and found that the majority of them do not consider the integration of legacy systems with the quality aspects in the pre-migration, migration and post-migration phases.

Kulmukhametov, A. et al. [21] propose a Semi-Automated Hybrid Matcher (SAHM) that takes into account the schema-level and instance-level information. Based on their empirical findings, they have determined that a multiple-source approach to matching, rather than a single-source approach, is more accurate. In the schema matching, Tute, E. et al. [22] investigated applying a domain-specific pretrained language model. The models they propose, namely Energy BERT and their Energy Sentence-BERT, are applied to the different datasets to show their capacity to leverage the domain-specific semantic representation of the datasets to boost the capability to learn the correspondence in the heterogeneous datasets. EasyBDI is a development of Schmidt et al. [23], which adds logical data that is distributed throughout the machines. It is an extension of the schema matching, integration and mapping to a global schema and analytical queries to distributed sources.

The following are just a couple of examples of how to decouple the logical integration from the physical locations and organization of data. Beyond being an academic question, C.K. Dehury et al. [24] pointed out that explainable and interpretable machine learning is imperative in the face of other impacts an ML system can have on real-world decision-making, regulatory and operational considerations.

2.4. Machine Learning-Powered Schema Matching & Migration Intelligence

One of the reasons for solving the schema matching problem with machine learning methods is that if the attributes were not consistently named, or if they had complex relationships, then traditional schema matching techniques based on similarity would not work well. If the schema-matching network requires more than two schemas to be matched, some machine-learning techniques are suggested by T. Sahay and A. Mehta [25]. To enhance the quality of the matches, they take a user feedback and bootstrapping approach, which exemplifies the possibilities of user validation in automated learning.

In soft-schema matching, which was proposed by Nozaki, K. [26], the set of attribute relationships is not defined one-to-one, but rather in a probabilistic way. One of these approaches is helpful in the case of heterogeneous migration: The real world has attributes that can have one-to-many or many-to-many relationships. Ellison, M. et al. [27] proposed a self-adapting migration methodology for NoSQL databases which can adapt the migration strategy dynamically based on the workload, schema evolution, latency, migration cost, precision and recall.

This work is proof that the migration strategy choice may be an adaptive decision problem in itself and not an engineering decision problem. Alrehamy, H. et al. [28] studied explainable artificial intelligence and pointed out the significance of explainable ML decisions given by the explainability mechanisms. It is particularly relevant when forecasting migration, as migration engineers may wish to understand why one part of the database is considered to be high risk but not just a number.

2.5. Graph-based Data Integration and Dependency Representation

A new trend is the use of graph-based methods, especially since the relations between the elements of a heterogeneous information source are not necessarily easily represented in a relational form or similarity scores at the attribute level. Not only are there two schemas matched, but some of these schemas are connected in some way, as has been shown in the schema-matching-network formulation of Rodrigues and da Silva. Similarly, P.G. Kolaitis et al. [29] demonstrate the relationship between schema evolution and costs of workload and migrations to select a migration strategy. There has also been explicit research on data integration from the graph point of view. M. Fahmideh and G. Beydoun discussed data integration aspects such as schema matching, entity matching and data fusion, and pointed out that the different data sources exhibit structural and semantic mismatches [30]. Data integration plans for graph data have even been assessed, and have revealed that graph representation can be used to simplify data integration across heterogeneous data sources while maintaining the connectivity of the data and allowing for an alternative architecture with regard to data integration. These studies have inspired the application of graph representations in STRIDE-DB: the relationships that are known to exist in the database are converted to a graph prior to the criticality analysis based on the migration. This is particularly true given that migration impact might grow beyond the individual table to dependency chains.

2.6. Complexity of Transformations, Migration Risk and Deployment in the Cloud

The amount of data in the database is not enough to assess the complexity of the migration. The magnitude of effort and risk to the transformation will vary based on the amount of work, schema structure, dependencies, semantic ambiguity,

and data quality and constraints on the target platform. There is good evidence in the literature to support this multi-dimensional view of migration.

F.J. Bermúdez Ruiz et al. [31] pointed out technical and organizational issues related to the migration of a legacy environment to a cloud environment, and migration can fail due to poor preparation and understanding of the migration.

M.F. Gholami et al. [32] have done a systematic study on the cloud-migration processes and identified a list of migration activities, techniques and concerns that should be considered while migrating legacy Applications to the cloud. The survey is a good illustration of demonstrating that migration is not a technical process, but rather a series of interrelated processes.

M.F. Gholami et al. in [33] investigated word vectors and semantic schema matching and showed that semantic information could help to identify correspondences between attributes that are represented in various ways. The approach of E. Ferragut et al. [34] It is a workload-model-based approach, which is an attempt to evaluate all possible migration options of a cloud-based database. They propose a framework to compare various cloud database options in terms of workload and models, and the importance of understanding the workload behaviour when deciding on a migration target.

He, Jie, and Wei Wang et al. [35] also identified the migration issues based on the empirical investigation with the domain experts, and demonstrated that there are technical and non-technical issues that influence migration decisions. In the literature, the study conducted by T. T. Quan et al. [36] on cloud-migration risk, and the literature on migration as a whole, reveals that migration risk may originate from a multitude of sources, such as security, data access, provider selection, compliance, financial or management.

A systematic review was conducted to identify types of cloud migration risks, which prompted the need to explicitly identify risks in cloud migration planning—lastly, Y.-H. Wang et al. [37] did systematic research on the factors that impact cloud adoption and migration, highlighting that many organizations are often driven to consider cloud environments due to legacy system limitations of maintainability and scalability.

2.7. Critical Analysis of Existing Literature

The literature study results indicated that there are five research directions, namely:

1. The second approach that can also deal with data and schema evolution automatically is more oriented towards data and schema integrity (e.g., EvolveDB, MoDevo), which is not specifically predisposed to be able to deal with schema evolution.
2. The majority of the recently developed ones are based on

the word embedding, a pre-trained language model or a mixture of word embedding and other similar matching methods [13, 17, 21, 22, 33] to accomplish their semantic schema matching task.

3. Today, some portions of the migration plan are modeled based on the workload and schema change, latency, cost, etc. of a scenario [9, 16, 27, 34].
4. Without a comprehensive view of the risks, the technical, financial, security, organizational and operational risks involved in the cloud migration have been identified [14, 15, 31, 32, 36]. Risk-aware cloud migration – without a perspective of schema transformation (technical and operational risks), especially regarding transitioning to cloud computing.
5. Explainable intelligent decision making — Although there is much literature out there on how to use research to explain the output of our ML Models, it may not be easy to fit that into the end-to-end DB migration process.

2.8. Identified Research Gap

Based on the literature, there were several gaps and missing linkages identified with the relationship between the SDA, SSEM, BRI, TCE and M2M; the ability to integrate EAI with a single database modernization framework and the ability to integrate EAI with the hybrid cloud. In the current study, an attempt is also made to do the same, and a significant amount of the work is done under the dimension analysis approach. In other words, a schema-evolution approach can provide an ability to generate migration scripts, however, it does not necessarily have to be able to assess the migration risk well; schema-matching can increase the semantic match, but does not typically have the ability to explain and analyse the uncertainty of the match as a migration risk; cloud-migration can analyze the general migration risk, but does not typically model the detailed database-level dependency; explainable-AI can provide mechanisms to explain and analyse, but does not address the uncertainty of the match as a migration risk at the database level.

3. Problem Formulation and Stride-DB System Model

A direct schema-to-schema conversion may be able to replicate tables and records effectively, but it causes some semantic differences or breaks some dependency relationships between the related database objects. However, before the generation of the transformation decisions, a representation of the system that uses the database to produce the transformation must be created in order to produce a reliable one. In order to meet this requirement, this paper proposes a database transformation process model, which takes into account the structure of the database, dependency relationship, semantic correspondence, complexity of the database transformation, risk of database migration and compatibility with the target environment. The proposed STRIDE-DB framework is designed to make the information from various

types of legacy databases available in a structured format to be utilized for database migration planning, automated database transformation, risk analysis and validation after transformation.

The first step of the proposed modelling process is to define a set of tables, attributes, data types, keys and inter-table relationships for the source database. This dependency information is then presented as a graph, and the graph structure is then used to model the association structure between the entities in the database. An attribute name similarity criterion is not the only one used to perform semantic mapping between source and target attributes; multiple evidence sources are used. The mapping confidence and dependency characteristics derived lead to an estimate of the complexity of the transformation and the risk of migration. Last but not least, these attributes serve as the foundation of smart migration planning and automated transformation of migration. The overall formulation keeps the representation of the source system, the decision making about transformation, the prediction of risk and the execution of the transformation separate. This separation allows for a single assessment of each stage, as well as for the whole system to be considered a single migration pipeline.

3.1. Problem Formulation

Suppose that the source legacy database is modeled as:

$$D_s = (T_s, A_s, R_s)$$

It is expressed as (T_s, A_s, R_s) , where T_s is a set of source tables, A_s is a set of source attributes, and R_s is the relationships between the entities in the database.

Similarly, the target database is displayed as:

$$D_t = (T_t, A_t, R_t)$$

The main goal of STRIDE-DB is to find a trustworthy mapping between the source database D_s to build database D_t .

$$\mathcal{M}: D_s \rightarrow D_t$$

While retaining the salient structural and semantic aspects of the source system.

This overall change can then be written as:

$$\max Q(\mathcal{M})$$

(M) is the complexity of the migration. The transformation quality is represented by $Q(M)$, the transformation complexity is represented by $C(M)$, the migration risk is represented by $R(M)$ and the complexity of the migration is represented by $C(M)$.

and

$$\mathcal{C}(\mathcal{M}) \leq C_{\max}$$

$$\mathcal{R}(\mathcal{M}) \leq R_{\max}$$

The acceptable complexity and the acceptable risk limits are represented, respectively. The problem that STRIDE-DB aims to solve is: how to transform a legacy heterogeneous database into a target database that maintains the structural relationships, identifies reliable semantic correspondences, determines the complexity of the transformation, foresees the risks of the migration, and validates the target database.

3.2. Structural Representation of Database Systems

The first modelling stage is to represent the legacy database by its structural metadata. Each table is described in terms of its properties, data types, important fields and connections with other objects in the database.

The structural representation of a table T_i is:

$$T_i = (A_i, K_i, D_i, R_i)$$

where A_i represents the attributes of table T_i , K_i represents the attributes of the primary key and foreign-key, D_i represents the attributes of the datatype, and R_i represents the attributes of the identified relationships.

The whole source database can then be represented as:

$$D_s = \{T_1, T_2, \dots, T_n\}$$

It is a representation that transforms the metadata from the database into a structured form that can be used by the following dependency and semantic analysis steps. The structural representation also helps STRIDE-DB differentiate between attributes based on their functional and relational roles in the design.

The name of an attribute is thus not the only thing that is kept, but also its key property, the type of the table which contains it, and its relationships.

3.3. Dependency Graph Modelling

Relationships between tables and attributes can impact database transformation. This results in a dependency graph of the source database, known as a STRIDE-DB:

$$G_D = (V, E)$$

V represents the names of database entities, and E represents the relationships between these entities.

A node can be a table or an attribute, and an edge is a recognized structural dependency. An example of this is a foreign-key association between two tables, such as:

$$v_i \rightarrow v_j$$

Where v_i is the referenced entity, and v_j is the dependent entity.

The dependency graph gives structural information that is needed to make further transformation decisions. It allows the framework to decide whether the attribute or table can be transformed in isolation or whether the transformation of the attribute or table is connected with other database elements. The dependency representation is then used for the transformation-complexity estimation and migration-risk assessment. Therefore, dependency analysis is not considered as a standalone visualization tool, but rather as a piece of information that is used to inform subsequent decision-making at the downstream. In this way, dependency analysis is not seen as a standalone visualization tool, but rather as part of the information that is utilized in the downstream decision-making process.

3.4. Semantic Schema Mapping

It is not always enough to only have structural correspondence to decide which attributes are equivalent in heterogeneous database schemas. Hence, STRIDE-DB brings semantic schema mapping to set up relations between the source attributes and the target concepts.

Let:

$A_s = \{a_1, a_2, \dots, a_m\}$ represent source attributes and:

$A_t = \{b_1, b_2, \dots, b_n\}$ represent target attributes or canonical target concepts.

The semantic mapping function is shown as:

$$M: A_s \rightarrow A_t$$

STRIDE-DB is not just based on literal attribute-name matching, but includes a number of forms of evidence for each source-target pair.

The generally accepted mapping confidence is shown as:

$$MC = f(N_s, S_s, D_c, V_c, R_c)$$

where:

N_s indicates evidence indicating name similarity.

The similarities of semantics are represented by S_s .

D_c is a datatype compatibility.

V_c is the value-profile compatibility,

The " R_c " stands for relationship compatibility.

Candidates are mapped, and the resulting maps are categorized based on their confidence. High-confidence mappings can be developed further to automated transformation; medium- and low-confidence mappings can be further verified. With this mechanism, STRIDE-DB can differentiate between a confident semantic correspondence and potentially ambiguous correspondence, which helps minimize the risk of uncertain mappings propagating to the transformation stage [39].

3.5. Transformation Complexity Estimation

The complexity of the planned transformation is estimated using its structural and semantic properties that are formed in the previous stages [40]. The complexity of transformation varies by the number of attributes to be transformed, dependency relationships, ambiguous mappings, datatype conversions, rules for transforming data and data volume.

The transformation-complexity function can be represented as:

$$TC = f(A_m, D_c, M_a, V_r)$$

where:

- A_m is the number and type of attributes mapped,
- D_c = dependency complexity,
- M_a is an ambiguity in mapping,
- V_r is a transformation requirement that is volume-related.

The goal of this model is to get an idea of the level of difficulty to transform before executing the migration. If the mappings and dependencies of a large number of attributes or records are well defined, then the transformation is not necessarily high complexity. However, a smaller transformation [41] can be complicated in the case of a large amount of semantic ambiguity or dependency relations.

Thus, the complexity of transformation in STRIDE-DB is defined by the structural, semantic, relational and data-volume characteristics, and not just by the size of the database.

3.6. Migration Risk Modelling

The transformation-complexity estimate is an important input in migration-risk assessment, but does not fully capture the overall migration risk.

The migration-risk representation is then enhanced by a number of attributes in the STRIDE-DB:

$$RI = f(M_u, D_c, TC, Q_d, V_v)$$

where:

- M_u is used for the mapping uncertainty,
- D_c = the dependency complexity,
- T_C is a measure of the complexity of the transformation,
- Q_d are the data-quality characteristics,
- V_v is the volume of data on migration.

This value for migration risk is then converted into qualitative values:

$$\mathcal{L}(RI) = \begin{cases} Low, & RI < \tau_1, \\ Medium, & \tau_1 \leq RI < \tau_2, \\ High, & RI \geq \tau_2, \end{cases}$$

It allows the estimation of risks in advance of the actual transformation. The feature set for the Machine-Learning (ML) component, discussed later in Section VI, is the risk

representation. Thus, STRIDE-DB separates the generation of the risk features, the modeling of the risk and the interpretation of the model, allowing the machine-learning model to run on a set of features created in the previous steps of the DB-analysis process.

3.7. Hybrid-Cloud Compatibility Modelling

STRIDE-DB is an application for Database Transformation in Hybrid Cloud environments. The transformed Database should be checked for the Hybrid Cloud environment where it is to be deployed.

The compatibility model takes into account the structural, semantic, dependency, datatype and deployment features:

$$HC = f(S_c, S_m, D_c, D_t, P_c)$$

where:

- The S_c reflects structural compatibility,
- S_m : semantic compatibility,
- The dependency compatibility is denoted by D_c ,
- D_t is the datatype compatibility,
- Platform and deployment are compatible with P_c .

The role of this component is to see if the changed database representation can be deployed in the desired hybrid environment. Cloud deployment is not a stand-alone task after migration; instead, STRIDE-DB takes into account the compatibility of the cloud deployment in the decision process of the cloud transformation. This will allow any potential incompatibilities to be identified prior to deployment [42].

3.8. STRIDE-DB System Modelling Workflow

The whole system model is presented as a sequential transformation process as follows:

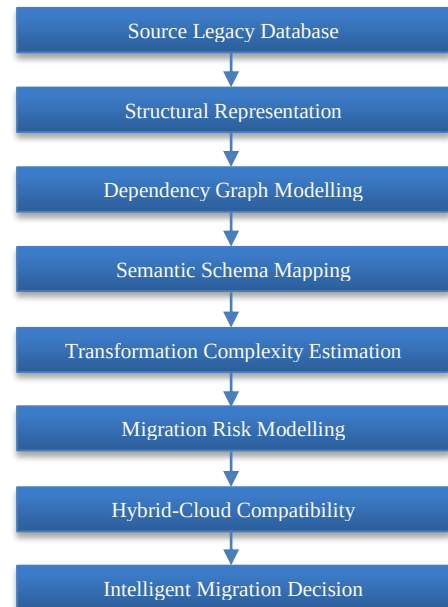


Fig. 1 STRIDE-DB system modelling workflow for intelligent database migration

4. Dataset Preparation, Metadata Discovery and Structural Characterization

A vital part of the value of an intelligent database transformation framework is the quality and completeness of the representation of the source system passed on to the downstream parts of the system. The source database is then fed into a controlled preparation pipeline (STRIDE-DB), followed by dependency analysis, semantic mapping and complexity estimation and modelling of migration risks. In the preparation phase, the data in the source files are converted to a common representation for the database and the attribute-level information and the evidence of the relationships are extracted incrementally [43-45].

The proposed pipeline is made up of 5 core activities: Dataset Ingestion, Source Schema Manifesto, Column-level Inventory, Candidate Relationship Discovery, and Relationship Verification. STRIDE-DB does not assume that any database relationships with similar names are valid, but rather identifies candidates for the relationships and verifies their validity by calculating the evidence of datatype compatibility, uniqueness and value-overlap. The relationships that meet the confidence criterion are sent to the confirmed relationship repository. This design makes it possible to make a controlled transformation from the raw input data to the structure it needs to pass on to the subsequent STRIDE-DB modules.

4.1. Data Sources and Preparation Methodology

All the data used in this research were obtained from the Adventure Works 2022 Data set, which is publicly available on Kaggle from Algorismus. The data set is a tabular representation of Adventure Works' business data, for offline analytical and Power BI usage. Source data is from the Adventure Works data provided by the SQL Server data source that has been mapped to CSV-based tables for offline use, as outlined in the data set [44].

Link for Adventure Works 2022 Dataset [38]:

<https://www.kaggle.com/datasets/algorismus/adventure-works-in-excel-tables>

The source tables used in this study were retrieved from public sources and were then modified for later use in the proposed STRIDE-DB framework. Tables downloaded from the website were studied and selected for their suitability for the experimental setup, and then presented in a standardized tabular format. The output files were then fed into an SQLite database called STRIDE_DB.db, which served as the source database for the next step of metadata discovery, schema analysis, relationship validation, data quality evaluation and data migration experiments. There are 7 source tables in the database set up for the experiment: Product, Region, Reseller, Sales, Salesperson, Salesperson Region and Targets. All of these tables have a total of 59825 records and 38 attributes. The Region table has the fewest records (10), and the Sales table has the most (57,851).

Table 1. Shows the entire structural distribution of the prepared data

Source Table	Number of Records	Number of Attributes
Product	397	8
Region	10	4
Reseller	701	6
Sales	57,851	10
Salesperson	18	5
Salesperson Region	39	2
Targets	809	3
Total	59,825	38

The output of the source tables was not considered as the final experimental output. Instead, they were seen as the past representation of sources, for which internal dependency and metadata representation in STRIDE-DB were developed. While ingesting, all 7 source files were read successfully in the SQLite environment. This resulted in a database, which was used in all the following experiments. The implementation includes the successful creation of the seven tables and loading them into the database.

4.2. Dataset Ingestion and Database Construction

The source files were added to tables and then added to a SQLite database. All files are ingested, normalized and converted to tables in the database. These 7 source relationships were a result of the process implemented:

$T = \{\text{Product, Region, Reseller, Sales, Salesperson, SalespersonRegion, Targets}\}$

The second of these STRIDE-DB modules is all set to export a common relational interface, via CSV to DB. It is able to get metadata from the original file representation as well as from the inspection of the database. All 7 files have been loaded to the SQLite environment, and the SQLite operation succeeded. This is an experimental environment that was reported to be running this version of SQLite: 3.37.2

The ingestion method also allows for the recreation of the source database in the event that experiments are repeated on the same structured representation, ensuring that the following dependency, quality, mapping and migration operations are done on the same source database.

4.3. Source Metadata Discovery

After the data set has been ingested, STRIDE-DB can retrieve the structural data from the source database. It passes through all the user-defined tables and captures the columns, the data type that they have a definition for, primary keys, foreign keys, uniqueness constraints, check constraints and indexes. The information retrieved will be put in a source schema manifest, the first level of the metadata repository that will be used to further process the information. Implementation makes a separate column for the columns, primary keys, foreign keys, unique constraints, check constraints and indexes.

The source-schema inspection found 7 tables, each having 2-10 attributes. Importantly, these imported tables were not explicitly represented as primary-key or foreign-key constrained tables in the initial SQLite representation. The 7 tables that were imported were then reported as empty lists of primary keys, and were not listed as having foreign keys in the schema manifest. The observation is crucial for the proposed framework as it shows that dependency inference is needed for legacy database transformation. If they did not have partial or missing dependency info with no relationship info from the source system, then they would not.

4.4. Source Column Inventory and Structural Characterization

STRIDE-DB also creates a column inventory, which includes the attributes and their declared data types of the exact source.

The Product relation contains attributes including ProductKey, Product, Standard Cost, Color, Subcategory, Category, Background Color Format and Font Color Format are the attributes of Product relation. SalesTerritoryKey, Region, Country and Group make up the Region relationship. Reseller relation is composed of the following elements: ResellerKey, Business Type, Reseller, City, State-Province and Country-Region.

The column names in the Sales table are: SalesOrderNumber, OrderDate, ProductKey, ResellerKey, EmployeeKey, SalesTerritoryKey, Quantity, UnitPrice, Sales and Cost. The Salesperson, SalespersonRegion and Targets relations all have five attributes.

This inventory is at the basis of the levels of dependency analysis and semantic analysis of attributes. Also emphasizes schema name differences and datatype declaration that might have an effect on automated schema correspondence.

4.5. Candidate Relationship Discovery

This implies that when building the confirmed dependency representation using the STRIDE-DB tool, the dependency relationship needs to pass a candidate relationship discovery process to confirm the dependency relation.

The first step in the candidate-generation process is to strip any common separators (such as spaces, underscores and hyphens) that might exist between the components of an attribute name, and give it a unique name. In addition, it removes frequently used "suffixes" from attribute names, such as Key and ID. If there is a correspondence on attributes that are normalized in different tables, then candidate relationships will be created.

If there is only the first similarity to the structure, there will be 16 relationships possible. The appropriate links were included in these statements, such as:

product.ProductKey → sales.ProductKey
 region.SalesTerritoryKey → sales.SalesTerritoryKey
 region.SalesTerritoryKey → salespersonregion.SalesTerritoryKey
 reseller.ResellerKey → sales.ResellerKey
 sales.EmployeeKey → salesperson.EmployeeKey
 sales.EmployeeKey → salespersonregion.EmployeeKey
 sales.EmployeeKey → targets.EmployeeID

The “foreign-key” correspondences do not have to be used during the candidate-generation phase. This is because there can be “false” relationships due to lexical similarity. There were some candidates that were generated from the implementation that were rejected during verification because they were Product and Reseller attributes. So, the candidate generation is deliberately a discovery phase with a high recall rate, and the verification of the relationships is a high precision filter phase.

4.6. Data-Level Relationship Verification

Each candidate's relationship is then looked at in the light of the data-level evidence. STRIDE-DB calculates an aggregated verification score, value uniqueness, value overlap, datatype compatibility and referential coverage. When the candidate pair $C = (a_i, a_j)$ is considered to be first tested, it is determined if the type of the target attribute and the type of the source attribute are compatible. This is referred to as Datatype Compatibility. This is then followed by the removal of the nulls and calculation of unique and value/overlap properties.

To determine the uniqueness of an attribute, it is done based on the ratio of the number of values that are not null to the number of different values of the attribute (non-null). The following test might be helpful for making a decision if an attribute is likely to be a P/K-Q attribute. It is then that the candidate attributes that are calculated have an overlap. It can be used to find out, for a given value of one attribute, the distribution of values of the other attribute of the other table.

Finally, these attributes are tested against the relationship, and a new category is created: relationship-confidence and a new attribute will be created: verification_score. The following fields exist in the verification repository that is run: datatype mismatch, parent uniqueness, referential coverage, verification score and relationship status.

This verification is necessary, as a result of the similarity in names of the attributes; there was no way in the old system to know whether there was a dependency or not.

4.7. Confirmed Relationship Repository

Only the relationships of HIGH_CONFIDENCE will be placed in the STRIDE-DB confirmed relationship repository after verifying a relationship. The candidates with medium and low confidence are pulled apart for review to make sure

no evidence that did not meet the criteria for one of these confidence levels was overlooked.

After execution, 8/16 relationships were found to be of high confidence. The following associations were made: Product to sales; Region to sales; Region to salesperson-region; Reseller to sales; Sales to sales; Sales to salesperson; Salesperson to sales; Salesperson to salesperson-region; Salesperson to targets. The scores of verifications ranged from 0.8515 to 1.0000. The candidate relationships that were left were retrieved and stored in another review repository. With this implementation, there will be a separation that allows for an audit trail for use when deciding to automate migration and unknown relationships.

This dependency-graph module is the relationship repository-confirmation dependency relationship—both lexical similarity and explicit verification guide the construction of the dependency graph.

4.8. Data-Quality & Structural Readiness

The next characterization of the data set was to test the data set to make sure that the data are complete and unique when it is migrated to the new data set, along with being valid and consistent. The quality assessment shows that there are 6 tables with a completeness of 100% and 98.24% for the product table. The scores of the indicators of uniqueness, validity, consistency and referential integrity for the evaluated tables were found to be 1.0. The data-quality score was about 0.9995, and the data-quality score of the product table (0.9824) was also the lowest among all tables that had the lowest completeness score.

The following are described as properties of the data set used for the experiment, and are deemed a relatively "pure baseline data set" which would be used as a baseline in the proposed migration framework. Concurrently, the marginal missingness of the product relation prevents one from considering the experimental set as being completely homogeneous and allows parts of the framework, which rely on the quality of the source, to be run on a sub-optimal quality condition.

4.9. Section Summary

In the dataset-preparation stage, the structure that is needed by STRIDE-DB is set up. A SQLite version of 7 source tables, containing 38 attributes and 59825 records, has been built. The metadata extraction led to the development of the structured schema manifest, which then provided detailed attributes/datatypes representation at a column level that can be used for further analysis. For the lack of explicit relational constraints from the schema, STRIDE-DB could find 16 candidate relationships, and then test to determine if they are valid by using type evidence, unique evidence, value-overlap evidence and referential-coverage evidence. Eight relationships were moved to the high-confidence

relationships' repository, and eight relationships were held on for review. The "staged" preparation phase transforms all the legacy sources into a "structured" and auditable format that will be leveraged by the dependency graph, semantic mapping, transformation complexity and migration risk modules that are included in STRIDE-DB.

These well-prepared structural and semantical repositories can then be introduced in the next logical manner – the STRIDE-DB Architecture and Intelligent Migration Framework in the overall architecture.

5. STRIDE-DB Architecture and Intelligent Migration Framework

STRIDE-DB is a sequence of steps that can be followed to transform the heterogeneous legacy database to a structured target database and then to migrate and deploy it to a hybrid cloud. The combination of metadata discovery, dependency analysis, schema mapping, architecture transformation assessment, architecture migration risk characterization and execution of architecture migration is wrapped up in one workflow: Architecture. Although the source database is not schema converted, the evidence of structure and semantics is stored in intermediate repositories incrementally using the STRIDE-DB software to store evidence for future decisions.

This is the overall architecture that is utilized for the concept 'discover, represent, verify, assess, decide, transform and validate'. The metadata discovery part presents the structure of the source database, and the dependency and semantics part discovers dependency relationships and attribute-level correspondences. These outputs can be utilized to figure out the complexity of the transformation and the risk of migrating before conducting the migration. The architecture thus preserves the separation between analysis and transformation, and the risk that the uncertainty of structural and/or semantic information is directly propagated to the target database is minimized.

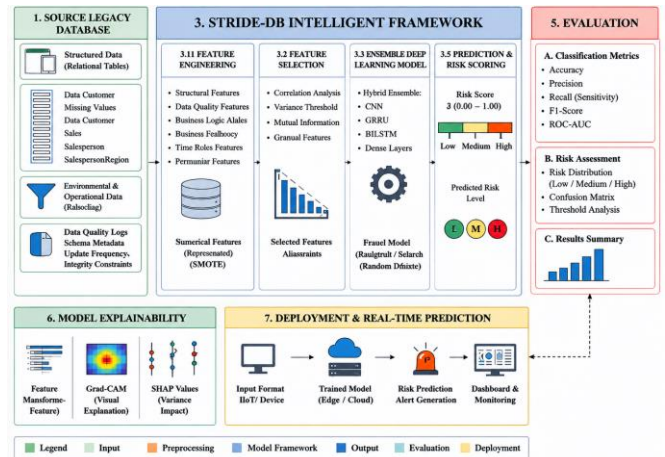


Fig. 2 Overall architecture of the proposed STRIDE-DB intelligent database migration framework

5.1. Metadata Discovery and Structural Representation

The first element of First architecture is the Metadata Discovery Engine that converts the source database into a machine-readable format. Once linked to the SQLite source database, STRIDE-DB can fetch tables from the source database, and table-level information such as column names, declared column data types, column nullability information, column default values and primary-key information. The implementation starts with creating a metadata repository that holds metadata about the database and a collection of column metadata, organized by table. The other components are organized according to the metadata that has been found. In the experimental implementation, the product table, the region table, the reseller table, the sales table, the salesperson table, the salespersonregion table and the targets table are the seven source tables.

This stage is intentionally considered separate from the semantic interpretation stage. So that the structure of the original will be preserved, and the high-level knowledge is acquired with the help of dependency analysis and semantic analysis.

5.2. Metadata Knowledge Repository

The metadata found is stored as an intermediate representation of the metadata, both between the source database inspection components and the downstream analysis components, in a Metadata Knowledge Repository. The repository stores the identity, type, and information about the tables and the metadata at the attribute level. This representation can be used as one representation of the source system, and other modules can use this one without making a number of contacts with the source system.

Also in the repository are the structural inputs for the dependency discovery and for the semantic mapping. So, it is not a metadata extraction "one-off" report; it is a piece of knowledge that can be reused throughout the pipeline at any time.

5.3. Dependency Analysis and Dependency Knowledge Graph

The higher-level of architecture is the structure of the relationship among the entities in the source database. STRIDE-DB is not dependent on any constraints in the database, but on the relationship repository created in the previous relationship-detection/relationship-verification stage. The dependency information is shown as a directed graph, with database entities being nodes and verified relationships being the directed edges. The implementation then constructs the dependency graph, based on the verified relationship manifest.

This is the graph that represents the structure of the database and illustrates the relationship between elements in the database. It also helps the framework to calculate the

dependency depth and find the components that have a higher level of importance in migration. The dependency-criticality component also analyzes some of the characteristics of the graph to identify which of them are structural in the database.

The dependency graph needs to have two functions: one to be displayed and the second to make migration decisions. With planning transformations, more attention can be paid to the more dependent components and/or those that have more downstream impact.

5.4. Business Rule and Integrity Analysis

Apart from these rules, there are other rules in both attributes and data values that affect the transformation of the database, but these rules are not explicit. STRIDE-DB contains a Business Rule Extraction Engine that will automatically detect the rules and integrity properties that can be extracted from the source database. Completeness, Pattern, Domain, Range, Uniqueness, Referential Integrity, and Nullability are the rules that are applied. A business rule analysis is then expanded to be the basis for the subsequent migration assessment.

This is because the dependency graph is the only model that can be used to describe the behaviour of the source database. It might not have any known relationship, but it might have a high attribute domain, uniqueness or attribute characteristic of completeness, which is considered to be retained in the transformation, for example. The outcome of this is that the information on the business rules is still retained as evidence of the transformation and migration phases.

5.5. Semantic Schema Mapping Engine

The Semantic Schema Mapping Engine (SSME) is the name of STRIDE-DB's semantic layer. The function of its is to find correspondences between the source attributes and the target concepts in the canonical domain. Implementation involves semantic attribute profiling and canonicalization, generation of candidates, semantic matching, confidence estimation and construction of the target-schema.

The mapping process takes into account the semantics of the source attributes, but doesn't just consider the equality of column names. This is particularly so if the database is a mix of data types and they have different names. The evidence for the semantics and structure is the only evidence that all candidates for mapping are evaluated. High confidence mappings can be automated, and Ambiguous mappings can be questioned. This design establishes an important difference between STRIDE-DB, in that semantic uncertainty is identified prior to the creation of a target database, as opposed to after.

5.6. Canonical Target Schema Generation

Then, STRIDE-DB constructs a canonicalized representation of the target schema in semantic analysis. The

implementation is clearly defined, it constructs a cloud-ready target representation that is normalized, and in which the semantic concepts that are captured in the semantic mapping phase are represented. The common representation is the canonical target schema that allows target attributes to be aggregated if there is more than one source attribute that represents the same concept.

For instance, a couple of attributes could be of the same identifier but in various tables, or even in the same table, but under different names. No need to save all of the source-specific representations, but rather a common concept aimed at is created during the process of canonicalization.

In this case, the semantic analysis stage will then be used as a fixed representation towards the next transformation process and to achieve the desired goal.

5.7. Transformation Complexity and Migration Risk Assessment

The architecture contains evidence as described above, which can be used to evaluate the complexity of the transformation and migration risk. The complexity will vary based on the characteristics of the source structure, dependency relationships, mapping results, requirement for review and properties of the data. Thus, it is possible to have a relative simplicity for the case of STRIDE-DB transformations compared to transformations with more structural and semantic ambiguity.

The characteristics are then taken into account to calculate the migration risk assessment. The new implementation includes the risk-feature engineering stage, where the pre-migration risk profile for each source table is created, including the mapping uncertainty, data-quality risk, dependency-related characteristics, and criticality-related information.

The risk assessment is then passed onto an intelligent migration decision process. Separating will allow you to maintain the difference between:

1. structural evidence,
2. semantic evidence,
3. transformation complexity, and
4. migration risk.

Sometimes, a transformation is structurally simple, semantically ambiguous or structurally complex and has highly reliable mappings, which is why the separation is important.

5.8. Intelligent Migration Decision and Transformation Pipeline

The final layer of architecture is the decision-making on the migration, which is based on the results of the previous layers. Before applying the mapping transformation,

STRIDE-DB evaluates all of the mappings, mapping needs, risk characteristics and representation of the target schema.

There are a number of migration scenarios to choose from: baseline, semantic-stress, data-quality-stress, dependency-stress, combined-stress and optimized. The scenarios are then walked through the migration path, which has been established in the framework, the migration pipeline. At the execution stage, the source records are turned into the target representation and the execution time, throughput, errors in transformation, warnings, and whether the transformation was successful or not are recorded. The new implementation will then run the validation after the transformation and check for the number of rows in the source table, comparing the number of rows in the target table, the characteristics of null cells and the completeness of the target table.

This architecture creates a closed migration workflow:

Source Database→Metadata→Dependencies→Semantic Mapping→Risk Assessment→Migration Decision→Transformation→Validation

The final validation phase is crucial since the successful implementation is not taken for granted as a successful migration in the framework. Both the Structural constraints and the data-preservation constraints need to be enforced on the target database.

5.9. Integrated STRIDE-DB Architecture

The idea of the STRIDE-DB is to be a 5-layered structure, with the inclusion of 5 different functional layers:

1. Structural identification: for the description of the metadata in the Source Understanding Layer.
2. The Knowledge Construction Layer is made up of the metadata repository, dependency graph and business-rule repository.
3. Semantic Intelligence Layer: semantic profiling, attribute mapping, confidence assessment and generating a target-schema.
4. Transformations: Complexity and migration risk assessment are easier with RiskAware DecisionLayer.
5. Smart Migration: Planning, Migration Transformation, Migration Execution Monitoring and Migration Post-Validation.

These layers enable STRIDE-DB to turn evidence-based and evolve from a simple schema-to-schema migration to one. The information that is gathered in each stage is passed on to the next stage, and problems or unclear mapping can be detected in each stage. The architecture thus offers the framework for the experiments on intelligent migration, which are described in the following sections. The next stage of the study will focus on the machine-learning-based migration-risk prediction method and explainable artificial intelligence mechanism, in which the risk characteristics generated by

STRIDE-DB will be used in predictive modelling and explanation.

6. Machine Learning-based Migration Risk Prediction and Explainable Analysis

The previous sections have described the analysis done by STRIDE-DB at the structural, dependency, semantic and transformation levels. However, the difficulty of migration alone is not enough to determine whether or not we expect operational problems with a database conversion. Factors that can contribute to migration risk include, for example, semantic uncertainty, degradation of data quality, dependence characteristics, transformation complexity, and criticality of the structure of particular database components. Thus, in this work, we extend the STRIDE-DB framework by adding a machine learning-based risk prediction layer that combines these different migratory features into one single risk assessment method.

The goal of this layer is not to assign a risk label to a part of a database, but to systematically correlate observable migration characteristics to the level of migration risk they induce. The predict risk has been made more interpretable by incorporating explainability. Therefore, the proposed approach can help migration planners to understand why a certain database component has been classified as low, medium or high risk instead of presenting an opaque prediction.

6.1. Building Migration Risk Feature

The first step of the predictive layer is the construction of migration-risk feature representation. The features are derived from the outputs of the previous STRIDE-DB components and are not generated by themselves from the raw database.

The feature construction process consists of combining information of five main dimensions: structural characteristics, dependency characteristics, semantic mapping characteristics, data-quality characteristics and transformation characteristics. The information provided by these dimensions complements the potential difficulty of migration.

Structural characteristics determine the basic properties of a database component, e.g. the number of rows, the number of attributes or the structural complexity. Dependency features indicate the level of participation of a table in upstream and downstream relationships. The semantic characteristics are the mapping confidence and mapping uncertainty, and the data-quality characteristic is the reliability of the source data. Transformation features are features that represent the expected difficulty of transforming the source representation into the canonical target representation.

The feature vector obtained for a database component T_i can be expressed as:

$$X_i = [X_i^{struct}, X_i^{dep}, X_i^{sem}, X_i^{quality}, X_i^{trans}]$$

where $X_i^{struct}, X_i^{dep}, X_i^{sem}, X_i^{quality}, X_i^{trans}$ represent structural, dependency, semantic, data-quality and transformation characteristics respectively.

6.2. Migration Risks Category of Features

The resulting representation of migration risk is grouped into interpretable features.

6.2.1. Structural Features

Structural characteristics define the physical and logical attributes of source tables. The main characteristics are the number of attributes, the structural complexity, the number of mapped attributes and the volume of rows. Large tables may require more processing resources.

Tables with large numbers of attributes may impose additional transformation requirements. So, structural complexity is considered to be one of the components that contribute to the migration-risk profile.

6.2.2. Features of Dependency

Then the dependency features are extracted from the validated dependency graph. These are dependency depth, upstream dependencies, downstream dependencies, graph degree and dependency criticality. A table that is part of many relationships can affect many other parts of the database. Then, migration errors in highly connected components can be propagated to the depending components. Thus, STRIDE-DB exploits dependency features in risk modeling.

6.2.3. The Significance of Semantic Maps

Semantic mapping is a projection of source features onto canonical target concepts, quantified. The properties of the mapping confidence and mapping uncertainty are calculated for the source tables.

As the confidence in the mapping increases, the transformation process is better adapted to be automated, and if the mapping is more ambiguous, then it needs more processing. This is significant since even with a well-defined database structure, ambiguity may be possible semantically.

6.2.4. Data Quality Attributes

The data quality assessment stage is responsible for the data quality attributes. It is based on the following criteria: Completeness, Uniqueness, Validity, Consistency and Referential Integrity.

The characteristics listed are incorporated into a data quality representation to assess the risk of migration. A database component can have poor-quality data and might need extra cleansing or transformation operations that can make it riskier to migrate.

6.2.5. Transformation of Features

The difficulty and effort required to transform the source representation to the target representation (usually the canonical form) are called transformation features. STRIDE-DB provides a profile of complexity for the transformations, mapped attributes, mappings to review, mapping confidence, data quality, criticality, row volume and complexity. The risk model is filled with an indicator of the source system status and the conversions to expect.

6.3. Migration-Related Risk Representation

Individual risk features were normalized and integrated into an overall risk representation for migration. Risk indicator can be considered as:

$$R_i = f(F_i^{struct}, F_i^{dep}, F_i^{sem}, F_i^{quality}, F_i^{trans})$$

The database component, T_i , and the evaluation of the risk function, $f(\cdot)$, are given by the migration risk indicator, R_i .

The risk indicator generated is explained in the pre-determined risk categories. The risk distribution available in the framework implemented is LOW, MEDIUM and HIGH. Such a 'categorical' representation is easier to use for planning migrations than is a number of migrants. The numerical risk indicator is still valid to order the migration components, but the categorical risk level gives a direct representation of the decision support.

6.4. Ensemble-based Migration Risk Prediction

An ensemble-learning approach is used to build STRIDE-DB and provides a stronger risk prediction. Rather than relying on a single predictive model to learn a single relationship between migration characteristics and risk, multiple alternative models can be used that are able to learn multiple relationships.

The overall ensemble forecast is:

$$\hat{y}_i = \mathcal{E}(h_1(X_i), h_2(X_i), \dots, h_K(X_i))$$

The feature vector of the component, a single predictive learner, the number of learners and the aggregation mechanism of the ensemble are indicated by where, $\hat{y}_i$, $h_1(X_i)$, $\mathcal{E}$, respectively.

The ensemble perspective would be helpful in the migration-risk problem, as the migration properties may have multiple relationships. For instance, if it is a transformation that is difficult to perform, the volume of that row may be more important; if the dependency criticality or the semantic uncertainty is more important, then those factors would be more important. It should be emphasized that these two should not be the only ones that will be used for a final model

selection procedure, but stability and interpretability of the models will also be important.

6.5. Risk Prediction in Controlled Migration Scenarios

To test this, controlled migration scenarios are tested in the predictive framework to determine if the risk model is sensitive to the stress in the individual migration scenarios.

There are 6 scenarios for the experimental implementation:

- S01 - This would be the original (baseline) data quality and mapping conditions.
- S02 - Semantic stress: the more ambiguous the semantic stress of the attribute mappings.
- S03 - Data-Quality Stress Deteriorated data-quality conditions.
- S04 - Dependency Stress: Dependency-aware migration in higher dependency influence.
- S05 - Combined Stress: data quality is not good, and mapping conditions are not good.
- S06 - Optimized risk-based migration with optimized transformation and resolved mappings.

The scenarios also enable the framework to validate the migration behaviour for each of the scenarios of controlled changes and the behaviour of the system for one operating condition. This is a scenario design and is important because we can now consider how the expected risk is affected when we incorporate adverse migration conditions into the system and optimal conditions when we reintroduce them.

6.6. Explainable Artificial Intelligence for Migration Risk

An important goal of the proposed structure is to be able to provide risk forecasts for migration that are interpretable. If it does have a risk classification, it is unlikely to be enough information for the actual migration - the migration engineers would need to understand what it means they have a low risk and what it means they have a high risk.

Thus, Explainable Artificial Intelligence (XAI) is considered a post-prediction analysis layer in the proposed framework. The explanation process links the output of the model to the features of migration that produce the output. A prediction is a statement of an idea or concept that can be made into a statement:

$$\hat{y}_i = \phi_0 + \sum_{j=1}^m \phi_{ij}$$

where $\hat{y}_i$ is the prediction of the database component for the baseline, ϕ_0 is the contribution of the feature to the prediction of the database component.

This is the explanation that renders the prediction understandable as a set of factors. For instance, the high-risk

prediction could be more relevant for dependency, less confident in their semantic mapping, more complex the transformation itself and the more they are uncertain of the quality of data. The individual contribution can be looked at in the XAI layer.

6.7. Global and Local Risk Interpretation

There are two levels to consider the explainability mechanism. The migration routes for a global interpretation are those that are most important to the set of data being interpreted. This gives an overview of what factors are most likely to affect the risk of migration. Local interpretation for predicting a particular database table or migration part. When considering 'real world' migration, it can be helpful when different tables are to have different risk values for different reasons.

Global interpretation and local interpretation are complete perspectives on the predictive model. Global explanations are used to communicate the general behaviour of the model, and local explanations are used to make local migration decisions for each part of the model.

6.8. Risk-Aware Migration Decision

The outcomes of the explainability and predictive layers will be input to the migration decision-making process. STRIDE-DB will not treat all of the components of the DB equally; instead, migration activities will be prioritized based on estimated risk.

A conceptual decision rule is given by:

$$D_i = \begin{cases} \text{Automated Migration,} & R_i < \tau_L \\ \text{Migration with Monitoring,} & \tau_L \leq R_i < \tau_H \\ \text{Review and Controlled Migration,} & R_i \geq \tau_H \end{cases}$$

where R_i is the estimated migration risk τ_L and τ_H are the lower and upper decision thresholds.

Using this risk strategy, limited intervention of low-risk components and medium-risk components will be migrated and monitored. An expert review may be done prior to transformation for high-risk items. Importantly, this does not take the place of human decision-making. Instead, it gives information that might be relevant to the migration engineer to take into account less certain or more influential aspects.

7. Automated Transformation and Migration Execution Pipeline

In previous sections, we have shown the structural, semantic, dependency and risk-aware analytical capabilities of STRIDE-DB. The next step is to do more than analyze that output and convert it to a migration technique that can be done. The automated transformation pipeline automates the transformation, transforming the validated source

representation to the target representation, while preserving structural and/or data properties of the source database that are relevant to the transformation.

The proposed pipeline did not return the result of the dependency verification operations, semantic mapping, estimate of the complexity of the transformation and evaluation of the migration risk prior to the transformation, but an operation of copying the database. This makes the migration process easier to handle as to which mapping can be migrated without any additional thought and which require more.

The migration process consists of selection mapping, the target schema being generated, the schema being prepared, the development of the transformation rules, the generation of SQL, running the SQL, monitoring the execution, and post-transformation schema validation.

7.1. Automated Transformation Strategy

The transformation process starts with the completed semantic mapping repository. There are target representations and target concepts for each source attribute. The information put into the mapping is passed on to the transformation engine to determine what attributes to map from the source to the target DB.

The transformation process can be expressed in the following manner:

$$S \xrightarrow{M} C \xrightarrow{T} D$$

The above shows S as a representation of the (source) database, M as the validated semantic mapping, C as the transformation operation and D as the target database. It is a gradual process to enable some changes to the representation without having to change the source schema.

7.2. Mapping-Driven Target Schema Generation

The data they're going to be pulling from is going to be based on a mapping, and that will be their basis for creating the ETL process. As defined by W3C, an ETL process is one that is based on a mapping.

The target schema is developed based on the target attributes that are retrieved in the semantic mapping phase. The target representation guarantees that the names and types of attributes are the same throughout the representation, yet that the meaning of the attributes is preserved from the time of the map creation.

The target attributes (id, measurement, temporal attributes and descriptive attributes) are created as a result of the implementation, using the canonical representation. If there are several attributes from the same source that have the same target concept, there is a possibility of merging them in the process of generating the target schema.

This mapping approach will reduce the need to rely on the source-specific naming convention and provide consistent images to support the subsequent migration operations. The dependency analysis identified structural needs that are also included in the target schema. In other words, the generation of schema is not dependency independent and dependency knowledge is also needed to maintain the dependency structure needed for the maintenance of the target representation structure.

7.3. Transformation Rule Generation

After the target schema is determined, the rules of transformation are then stated, given the transformation from source to target.

A transformation rule is a pictorial representation of:

$$r_j: A_j^S \rightarrow A_k^T$$

where A_j^S is a source attribute and A_k^T is the target attribute.

For each rule, there is a mapping, and this is used to fill the target representation. It can be any of the following operations, based on the mapping: direct transfer, attribute rename, datatype conversion, normalization/consolidation.

The explicit rules for transforming attributes give a traceability from the source attribute to the target attribute. This is especially relevant when considering migration auditing, as each attribute can have a decision made during the mapping process related to the production of the attribute.

7.4. SQL Generation and Execution

The transformation rules are then translated into actual database operations. The required SQL code to create the required structures and transfer the transformed data is automatically generated in the automated pipeline. In the execution phase, the actions are performed in a generic manner:

Target schema creation → source data extraction → transformation → target insertion → execution monitoring.

Operational indicators like number of rows processed, number of rows transformed, errors, warnings, execution time and success or failure of transformation are taken during the execution.

The implementation shows the feasibility of this automated execution process with the experimental database. The base execution was able to process and transform 59,825 rows, and there were no transformation errors, i.e., no errors in the transformation pipeline implemented.

Execution time: how long it took to execute, execution throughput:

$$\text{Throughput} = \frac{N_{\text{processed}}}{T_{\text{execution}}}$$

$N_{\text{processed}}$ is the number of records processed and $T_{\text{execution}}$ is the time it takes to execute. This measure will give an indication of the efficiency of the transformation process.

7.5. Multi-Scenario Migration Execution

STRIDE-DB evaluates the robustness of the transformation pipeline by executing the transformation in a set of controlled scenarios. The scenarios are based on the following: Baseline, Semantic-stress, Data-quality-stress, Dependency-stress, Combined-stress and Optimized migration conditions.

The scenarios aim to change the migration conditions, but leave the underlying database workload unchanged. This allows the researcher to investigate the impact of the semantic uncertainty, data quality, dependency conditions and migration strategy on the migration behaviour in a safe manner.

Table 2. Summarizes the six different scenarios

Scenario	Condition	Migration Strategy
S01	Baseline	Direct
S02	Semantic Stress	Direct
S03	Data-Quality Stress	Direct
S04	Dependency Stress	Dependency-Aware
S05	Combined Stress	Risk-Aware
S06	Optimized	Risk-Aware

The concept of scenario-based execution is that it is not possible to assess a migration framework in an ideal situation. The controlled scenarios enable exploring the sensitivity of STRIDE-DB to various migration uncertainties in both intensity and type.

7.6. Execution Monitoring

The framework gathers details of the execution of each scenario during the migration. These are the key:

- number of rows that have been processed;
- the number of rows which have been updated;
- transformation errors;
- execution time;
- throughput;
- the percent of students who are successful at transformations; and
- validation status.

Of all the pupils, how many are successful in their transformational process:

$$TSR = \frac{N_{\text{transformed}}}{N_{\text{processed}}}$$

Where *Ntransformed* is the number of records that were successfully transformed, and *Nprocesseed* is the number of records that were processed. The value of 1 means that all the records that were processed were successfully transformed. Prediction of migration risk is separated from execution monitoring, and the predicted risk can be compared to the actually observed migration behaviour subsequently.

7.7. Post-Transformation Validation

Executing SQL successfully is not enough to make a successful database migration. Thus, the target database is filled up, and afterwards, a post-transformation validation process is executed by the program STRIDE-DB.

These are some of the indicators used when validating the representation of the source to the representation of the target:

1. source row count;
2. target row count;
3. row preservation;
4. target-column count;
5. null-cell count;
6. target completeness; and
7. overall validation status.

RP_i is the ratio of the number of rows that are kept to the number of rows in the table is represented as

$$RP_i = \frac{N_{target,i}}{N_{source,i}}$$

$N_{target,i}$ is the row number of the source of the table and $N_{source,i}$ is the row number of the target of the table.

If $RPI = 1$, then there are the same number of records in the target as in the source table.

All the tables tested returned a PASS score for validation, and the new implementation returned a row preservation = 1.0 for the baseline migration as a whole. This is evidence that the transformation pipeline that is implemented was able to maintain the number of rows evaluated during migration of the baseline.

7.8. Migration Validation Criteria

Instead of using one execution indicator for the validation mechanism, multiple checks are used. In order to be successful, a migration can only be successful when the structural condition and data-preservation condition are met. Another validation condition is:

$$V = V_{rows} \wedge V_{columns} \wedge V_{nulls} \wedge V_{completeness}$$

Here, each V is the result of a unique validation condition. This way, you can be more confident that the migration is correct because, if you were able to execute without any of the

wrong data in the target data, you would be wrongly interpreting it as a successful migration.

7.9. Migration Artefact Generation

Another artefact that is captured on the automated pipeline, and is long-term, is the capability to reproduce and audit migration. These are: Migration Outcomes, Validation Results and Target Database after Migration.

These artefacts have been produced to be used as the baseline for their execution:

- baseline_migration_outcomes.csv;
- migration_validation_results.csv; and
- stride_db_baseline_target.sqlite.

In addition, the scenarios that are evaluated are combined in the consolidated migration-outcome repository with the outcomes of the multiple scenarios. These artifacts will serve to provide evidence throughout the analysis and throughout the implementation of the migration. They can also be used to check the experimental results without having to redo the migration process.

7.10. End-to-End STRIDE-DB Migration Workflow

To complete the whole pipeline for the automated transformation, it needs to be expressed as:

Source DB $\rightarrow$ *Metadata* $\rightarrow$ *Dependency Analysis*

→ *Semantic Mapping*

→ Risk Assessment → Target Schema

→ Transformation → Execution

→ *Validation*

This workflow is a definition of a link between the analytical part and the operational part of STRIDE-DB. Structural and semantic information is used for extracting migration knowledge, target-schema generation and transformation rely on the migration knowledge, and independent validation measures are used for migration assessment. The proposed model is one of the important characteristics, in which the analysis, decision, execution and validation are separated. It helps the system detect problems with migration prior to execution, and gives evidence of the results of the migration once it is done.

8. Experimental Results and Evaluation Methodology

In this part of the paper, the experiments that were set up in order to test the proposed framework for Sustainable and Resilient Development of a Smart Grid (STRIDE-DB) are discussed. This stems from 4 complementary elements of the migration analysis, risk assessment, migration predictive modelling and reliability of the migrations. The same legacy database as built from the Adventure Works database (Section 4) and the entire processing pipeline described in Sections 5–

7 was used for all the experiments. The experimental methodology is based on analysing the source-database, finding the dependency, identifying the business-rules, evaluating the data-quality, mapping the semantics, complexity estimation in the transformation, feature of migration-risk, prediction of the migration-risk by using machine-learning techniques and finally the feature of migration-risk. Finally, an element of change data preservation test and a change structural is carried out on the target representation.

Evaluation of not only a machine learning prediction metric, but also the migration behaviour in operation is considered.

8.1. Experimental Environment

A user-friendly and fun implementation of Python in Google's friendly environment - Colab. It utilized the source database engine SQLite, data-processing engines Pandas and NumPy to accomplish its job and used these to build the repository. The dependency-graph construction and analysis are performed via the Network library, and feature selection experiments and machine learning experiments are carried out with the help of the Scikit-learn library.

Some of the more advanced ensemble modelling techniques, as well as explainability methods (SHAP, CatBoost, LightGBM and XGBoost) were added to this as well. Data collected from the database is stored in a database called "STRIDE_DB.db" – it is a SQLite database. This information was retrieved from the database and saved into a database (SQLite) file, STRIDE_DB.db.STRIDE_DB.db: This is a file with the source of the database (SQLite).

All of the metadata, schema information, business rules, dependency graph, knowledge graph, feature engineering, models, risk prediction, transformation, validation and experimental results are automatically generated.

Table 3. Represent the principal software components

Component	Implementation
Programming language	Python
Execution platform	Google Colab
Source database	SQLite
Data processing	Pandas, NumPy
Scientific processing	SciPy
Graph analysis	NetworkX
Machine learning	Scikit-learn
Advanced ensemble models	XGBoost, LightGBM, CatBoost
Explainability	SHAP
Visualization	Matplotlib

The implementation uses a fixed random state of 42 for the principal machine-learning experiments to provide

deterministic partitioning and model initialization where supported.

8.2. Experimental Dataset

All of the experiments in this section will be run using the set-up database, and Adventure Works (section 4) will be used. There are 7 source tables, 59825 source records and 38 source attributes in the experimental database.

There are 7 types of relationship, which may be either:

- Product — 397 records;
- Region — 10 records;
- Reseller — 701 records;
- Sales — 57,851 records;
- Salesperson — 18 records;
- SalespersonRegion — 39 records; and
- Targets — 809 records.

For all of the analyses of the structure, semantic mapping and for the risk assessment and migration experiments, the same database was used. 25 different concepts for the migration period were revealed, and a set of these concepts, considered to be a standard (canonical) set, was chosen. These concepts are: concept of identifiers, concept of the product attributes, Geographical attributes, concept of the sales measures, concept of temporal attributes, concept of information of employees and the target concepts.

8.3. Structural and Dependency Evaluation

The first experimental phase, which involved studying the source database and dependency that was identified by the STRIDE-DB, is discussed.

An analysis of dependency was carried out that revealed 8 high-confidence 7 source table dependencies. Listed below are some relationships between the product, region, sales, reseller, salesperson, salespersonregion, target and salespersonregion: Product-Sales, Region-Sales, Region-Salesperson, Reseller-Sales, Sales-Sales, Salesperson-Sales, Salesperson-Salespersonregion, Salespersonregion-Sales.

It is known that there are 7 nodes and 8 directed edges. It is not found, and the graph consists of a Directed Acyclic Graph (DAG). Finally, the criticality of the dependencies was computed based on properties of the graph. This information was not only used to 'feed' the visualization, but also to represent the information related to the migration risk.

8.4. Business-Rule and Data-Quality Evaluation

Like in STRIDE-DB, the source database is evaluated, and, in this case, business rules and/or Data Quality Attributes (DQAs) of the source database are taken into account. Assisted by the business rule analysis, these properties of the business rules are discovered: Completeness, Pattern, Domain, Range, Uniqueness, Referential integrity, and Nullability. These quality aspects include evidence of the integrity of the source attributes before being transformed.

The data quality will then be evaluated based on completeness, uniqueness, validity, consistency and referential integrity will be evaluated. This leads to quality measurements, which are represented as a feature profile in the migration-risk.

The overall data quality of the revised implementation was good, but with some deviations in terms of completeness (a few) for the dimensions in the product table and completeness of dimensions in other tables in the observed data quality was not violated. However, it was not evidence as a descriptive statistic that was analysed as evidence; rather, it was evidence to be analysed to develop the risk feature.

8.5. Semantic Mapping Evaluation

The aim of the semantic mapping experiment is to determine whether or not the concepts being mapped are the ones actually being targeted (the “canonical” concepts) can be mapped as closely as possible, and whether or not the attributes of the source can be mapped as closely as possible to the source. In the beginning, 38 source attributes are taken into consideration.

The result of the semantic mapping process is a set of candidate correspondences which are then checked for a set of evidence dimensions such as name similarity, semantic similarity, datatype compatibility, value-profile information and relationship evidence.

The aim was to provide a semantic equivalent representation of the 25 different concept(s), and this was done through a new mapping. The experiments were quite significant in one of the experiments, in which it was called “Semantic-stress experiment”, participants were given an opportunity to assess the semantic-stress conditions while mapping the mechanism in the activity. When the threshold value was 0.85, there were 4 mappings that were ambiguous out of 38 mappings. These mappings involved:

- product.Background Color Format → product_color;
- product.Font Color Format → product_color; and
- reseller.Reseller → reseller_id.

The first two were considered low confidence mapping (required review), and the reseller mapping was medium confidence, but required review. Then, we have 35 maps left to automatically map and semantically ambiguous maps (not resolved), which are discarded.

This uncertainty does not have any correlation to the automatic mapping, as it is the type of design to be tested that is uncertain in this experiment.

8.6. Transformation-Complexity Evaluation

The properties that were acquired in the previous modules of the STRIDE-DB were used to calculate the complexity of

the transformation. The Complexity profile attributes are: Mapped attributes, attributes that need to be mapped, attributes that cannot be checked, quality characteristics of the data, criticality, number of rows and structural complexity.

Database overall Transformation complexity indicator (computed): 0.2184 – is an overall LOW complexity level at the database level.

However, it was observed that the complexity of the transformation of the tables at the table level had the highest transformation complexity (0.4550) in all the tables in the sales table. This is why its role in the dependency graph (and its size) is called that. This was followed by an entered pre-migration risk feature profile, which would lead to this transformation.

8.7. Pre-Migration Risk Assessment

The output of the three analyses (structural analysis, dependency analysis, semantic analysis and data-quality analysis) is all merged together in stage migration-risk feature engineering. This will form a feature profile which will include the following features:

- row volume;
- attribute count;
- mapped attributes;
- review mappings;
- review ratio;
- mean mapping confidence;
- mapping uncertainty;
- data-quality score;
- data-quality risk;
- criticality score;
- row-volume score;
- structural complexity;
- transformation complexity; and
- pre-migration risk.

Pre-migration risk assessment of the tables was carried out at the time, and all the tables were rated as 6 LOW and 1 MEDIUM. Pre-migration, there was only one table (sales table) with a pre-migration risk indicator MEDIUM (0.3535) before migrating.

The overall pre-migration risk indicator was 0.1717, which was considered an overall LOW risk. This is because the table that is about to go through the transformation process is known to have a certain number of migration properties (on a representation level) prior to the transformation process.

8.8. Controlled Migration Scenarios

The following 6 controlled migration scenarios were created to examine the Framework's response to various migration scenarios and these were clearly illustrated in Table 4.

Table 4. Controlled migration scenarios

Scenario	Scenario Type	Strategy	Transformation Mode	Data Quality	Mapping Condition
S01	Baseline	Direct	Minimal	Original	Original
S02	Semantic Stress	Direct	Semantic Heavy	Original	Ambiguous
S03	Data-Quality Stress	Direct	Minimal	Degraded	Original
S04	Dependency Stress	Dependency-Aware	Dependency Heavy	Original	Original
S05	Combined Stress	Risk-Aware	Full	Degraded	Ambiguous
S06	Optimized	Risk-Aware	Optimized	Original	Resolved

Time since last birth (if known) – time unknown, woman is aware she has an unknown loss of pregnancy. It is the women with pregnancy history, but whose pregnancy history was unknown, who should be included in the list of pregnant women who should be suspected of having undergone abortion in a previous pregnancy.

8.9. Scenario-Level Risk Evaluation

Once a scenario has been controlled, then an indicator(s) can be decided on the scenario, which can then be calculated as indicated in Table 5 below.

Table 5. Scenario-level risk evaluation

Scenario	Complexity	Uncertainty	Risk Indicator
S01 Baseline	0.2184	0.040	0.2172
S02 Semantic Stress	0.3145	0.048	0.2525
S03 Data-Quality Stress	0.2730	0.040	0.2614
S04 Dependency Stress	0.2457	0.040	0.2268
S05 Combined Stress	0.3800	0.048	0.3004
S06 Optimized	0.1262	0.034	0.1838

In the case of the scenario S06 (Optimized scenario), the stress scenario S05 (0.3004, 0.1838) is tested. If there are variations in the risk representation caused by variations in the migration conditions, then these changes in risk representation are evidence for controlled variation. In the scenarios calculated, ambiguous semantics of data, and poor quality of data proved to be the most difficult, complex and dangerous scenarios.

8.10. Machine-Learning Experimental Design

The table-level migration risk feature repository is applied at the machine learning stage to apply a set of features to the tables. 7 observations (7 tables of data sources) are the limited set of data. A number is translated to the risk level, prior to the model training.

Other properties which can be used to do refinement are: Mutual-information analysis, Recursive Feature Elimination (RFE). To Imple:100 trees, 8 features in the refined representation of the RFE.

The data has been divided into a train-test split with 80% to the train set, with the random_state = 42. Due to a lack of observations in the data set, stratified splitting was not used, and one observation in the data set was at MEDIUM risk. When implemented for the predictive experiments, it is implemented on the non-stratified partition.

While it is a research property, which is why it is included here, with the intention of making it as clear as possible as to the methodology.

8.11. Comparative Machine-Learning Models

Comparative evaluation of some of the common learning algorithms will be done, and ensemble learning algorithms will be implemented. These are some of the models that have been tested:

1. Random Forest;
2. Decision Tree;
3. Support Vector Machine;
4. K-Nearest Neighbors;
5. Logistic Regression;
6. Gradient Boosting;
7. XGBoost;
8. LightGBM;
9. CatBoost; and
10. Soft Voting ensemble.

In the Random Forest, there are 100 Estimators (RF) present in the forest. The maximum depth of 8 and minimum splitting of 5 and gini splitting were used as a criterion with the Decision Tree. To implement the SVM, the SVM feature standardization, SVM with RBF kernel and SVM with soft margin are implemented. 1000 iterations were used for the model Logistic Regression, and it has been normalized.

For Logistic Regression: Number of iterations = 1000, Normalised features = yes.

Gradient Boosting: This is a random search over the space of hyperparameters, and attempts to tune the number of estimators, learning rate, the depth of the trees, the

subsampling and minimum split hyperparameters. It is implemented with 15 configurations, shuffled k-fold cross-validation and 2 k-fold cross-validation.

In the last stage of the advanced modeling, the performance of all other models (XGBoost, LightGBM, CatBoost and soft Voting) will be tested.

8.12. Evaluation Metrics

Various performance measures of predictive models, such as accuracy, precision, recall and F1-score, can be used.

Accuracy is:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

Precision is defined as:

$$Precision = \frac{TP}{TP + FP}$$

Recall is defined as:

$$Recall = \frac{TP}{TP + FN}$$

The F1-score is defined as:

$$F1 = 2 \frac{Precision \times Recall}{Precision + Recall}$$

For Multiclass, the weighted Precision, as well as the weighted Recall and weighted F1-score, are shown. One of the metrics (MEDIUM-risk) only labels were provided, and for the other, only preliminary feasibility statistics were provided, and of course, there were no statistics of the generalization performance.

8.13. Migration Execution Evaluation

To evaluate the operation migration, all the sources (59,825 records) were used. All the errors and warnings were not observed in the baseline scenario (S01), and a total of 59,825 were processed and transformed. The total throughput was 1353.57 rows/second, all transformations were successful (value is 1.0), and 44.1979 seconds was the execution time. All the 59,825 records were processed for all the experiments in the multi-scenario experiments. In all six scenarios, transformation errors were not seen.

Time to execute the query was 39.35 s to 57.34 s while the rows per second were 1,043 to 1,520 rows/s.

8.14. Post-Migration Validation

Post transformation validation: validates that the data that is moved to the source is correct. As well as the completeness of the target is checked, the following properties of the source and target rows, of the columns in the target and of the null cells are checked in the validation:

The row-preservation value of all 7 of the tables in the baseline migration is 1.0. The result shows that all the tables were validated, and there are no nulls found in target representations, and all gave a PASS score. The average of the Rows' value of retention was 1.0 at the time.

When implementing, an end-to-end validation of the transformation is then done. There are multiple repository reports; they are created in the following ways:

- Table Preservation: 100%
- Column Preservation: 100%
- Schema Preservation: 100%
- Datatype Accuracy: 100%
- Constraint Accuracy: 100%
- Overall Transformation Accuracy: 100%

The structure properties would not be sensitive to any value levels since no changes to the structure of the pipeline were made in the experimental one.

8.15. Explainability Evaluation

Last but not least, the models' explanations are checked in the experiments, based on the assumption of SHAP. Implementation: It produces a 'Tree Explainer' of the XGBoost model, and a visualization of the SHAP summary of the test data. The purpose of this analysis is to gain an understanding of the model's prediction in terms of features that impact migration risk. The SHAP summary and SHAP bar plot (contribution of the features) are created. Note that the following explanation of feature importance is at the case level only since we do not have many experiments and data points for these experiments.

8.16. Experimental Reproducibility

The intermediate and final experiment results are saved as artefacts (csv & SQLite), which can be reproduced at implementation. These are made up of: Semantic mappings, Migration-risk features, Scenario results, Migration outcomes, Validation results and target database. Most of the essential ones are experiments, which are run on a particular seed, and in which all the parameters of a configuration are written into the framework, using machine learning.

Traceability can be followed along the way of the experiment from:

source database → metadata → dependency knowledge → semantic mapping → risk features → predictive modelling → migration execution → validation.

8.17. Summary of Experimental Design

The experimental design is based on three aspects, all of which are connected and utilize STRIDE-DB. Under SDLC View, a detailed analysis of the framework from a structural point of view is carried out – Dependency-wise, Business Rules-wise, semantically, Data Quality-wise & Data

Transformation Complexity-wise. There were 2 scenarios identified as risky behaviour, for which the risk was estimated in the 2nd risky scenario and the 6 migration scenario. Thirdly, it is evaluated from a practical point of view by actual transformation, real execution and real post-transformation examination. In the machine learning part, it makes the first guess about the probability of the migration and in the SHAP part, it gives an explanation of the machine learning model. When the data is relatively small in comparison to the size of the migration load (59,825 records), and the current experimental design (at the time of this writing) suggests this is a limitation of the data to be modelled. Experiments are discussed, explained, and the discussion/explanation is then an explanation/explanation of the outcomes of the experiments.

9. Results and Comprehensive Evaluation

The experimental results of the proposed framework, STRIDE-DB, are presented in this section. The framework is presented at various phases of the migration process: Source data quality assessment, Semantic mapping, Complexity estimation for migration, Risk assessment, Migration under control, Migration based on machine learning and risk classification, Explainability, and post-migration validation. The goal is to know if the proposed framework is able to detect

migration-related uncertainties prior to the transformation, enable migration decision making for risk, and maintain the structural and data properties of the source database after the transformation. This evaluation was performed on the Adventure Works database that has 59,825 records in seven source tables and 38 source attributes. The experiments were then created in 6 controlled migration scenarios to evaluate the performance of STRIDE-DB for baseline, semantic, data-quality, dependency, combined-stress and optimized conditions.

9.1. Overall Migration Execution Performance

To check its ability to enable the transformation process without losing the source data, the first experiment is performed on STRIDE-DB. Records to be migrated from 7 source tables is 59825. The Success Rate of the transformation value will be "1.0" since there are no execution errors/warnings common to transformations. It took 38.900 seconds for the total migration, and a total throughput of approximately 1,537.9 rows/s. Following the insert/update, there were the same number of source rows, and from each of the seven tables, there were the same number of target rows. All the row-preservation criteria and all of the target completeness criteria were set to 1.0, as were all of the validation statuses PASS.

Table 6. Baseline migration and validation results

Table	Source Rows	Target Rows	Row Preservation	Validation
Product	397	397	1.00	PASS
Region	10	10	1.00	PASS
Reseller	701	701	1.00	PASS
Sales	57,851	57,851	1.00	PASS
Salesperson	18	18	1.00	PASS
SalespersonRegion	39	39	1.00	PASS
Targets	809	809	1.00	PASS
Overall	59,825	59,825	1.00	PASS

9.2. Multi-Scenario Migration Evaluation

In this regard, 6 scenarios are created and studied to understand the behaviour of the framework under various migration scenarios. All scenarios are baseline scenarios, a semantic stress scenario, a data-quality stress scenario, a dependency stress scenario, a combined stress scenario and an optimized risk-aware migration scenario. The baseline scenario (S01) data conditions are the data conditions that were used as the baseline for the direct migration. S02 had some ambiguous words, and some quality degradation in the

data is controlled in S03. In the cases of dependency-rich migration, S04 is for it, and in the case of degraded data, ambiguous mappings and complete transformation, S05 is for it. Then, optimized the condition (with migration S06) based on the risk, optimized transformations and resolved mappings.

The 5 scenarios were done without any problem (non-baseline). No transformation errors were reported, and execution time for the data was ~39.71-44.61 seconds and ~59,825 rows per scenario.

Table 7. Controlled migration scenarios

Scenario	Condition	Strategy	Mapping	Result
S01	Baseline	Direct	Original	Completed
S02	Semantic stress	Direct	Ambiguous	Completed
S03	Data-quality stress	Direct	Original	Completed
S04	Dependency stress	Dependency-aware	Original	Completed
S05	Combined stress	Risk-aware	Ambiguous	Completed
S06	Optimized	Risk-aware	Resolved	Completed

While the process for performing the migrations is similar to this experiment, different decision strategies and transformations can be performed by STRIDE-DB.

9.3. Semantic Mapping and Canonicalization Results

The results of this semantic profiling experiment were a list of 38 source attributes, of which 35 are associated with the semantic concepts from the canonical set. This means that the semantic coverage is 92.11%. Here are 3 attributes that needed to be specified: Background Color Format, Font Color Format and Reseller.

In the canonicalization process, these 38 source attributes were whittled down to 25 target attributes, and the possible ratio of consolidation was 0.3421. These were therefore grouped together under common concepts for the indicator, geographical information and product-related attributes in the common target concepts.

Table 8. Semantic mapping and canonical schema summary

Measure	Result
Total source attributes	38
Semantically resolved	35
Semantic coverage	92.11%
Unresolved attributes	3
Unique target attributes	25
Potential consolidation ratio	0.3421

This will be used to promote the use of semantic normalization prior to transformation, since attribute names can be mapped using a common language.

9.4. Transformation Complexity and Migration Risk Analysis

For each of the source tables, the Transformation Complexity Index (TCI) was calculated. The observed TCI values were 26.672 for Product, 24.150 for region, 25.700 for Reseller, 28.000 for Sales and 24.600 for Salesperson. The Sales table is the most complicated table to transform, and the Region table is the least complicated table to transform. The number of attributes and the number of transformation dependencies that are related to the Sales table are very different. The quality, dependency and semantic risk factors are then multiplied to get the Migration Risk Index. It is therefore not just the complexity of transformation that is a final indicator for migrations, but also the framework that provides a risk-based display.

Figure 3 depicts the Migration Risk Index obtained from STRIDE-DB at the table level. The MRI values are in the range of 9.61 and 11.44, indicating the variation in the migration risk among the components of the database. Product & Sales have a higher risk compared to Targets & Region. This variation is used to support table-level risk assessment for prioritizing migration activities and for choosing the appropriate migration strategy.

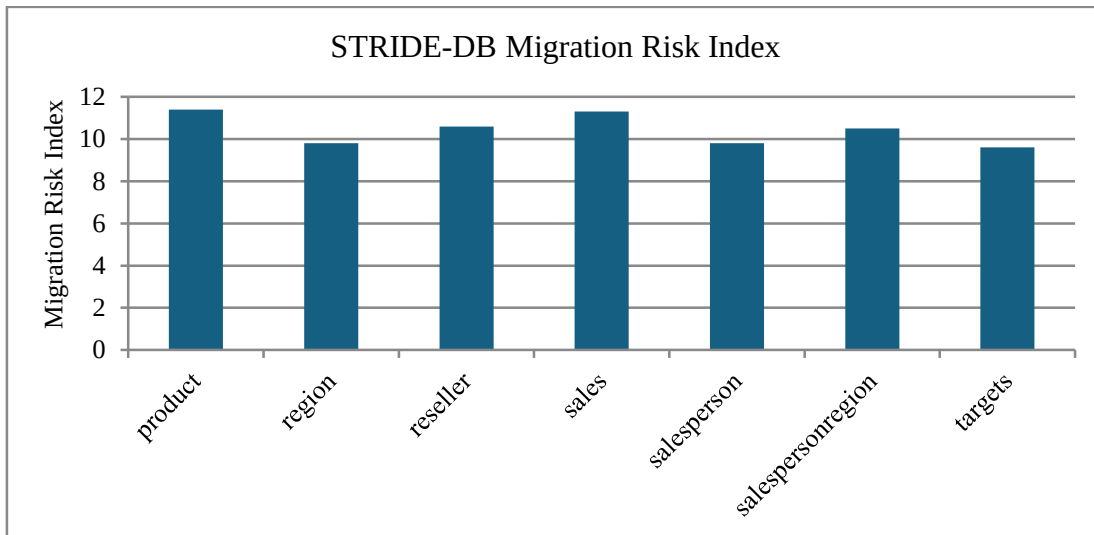


Fig. 3 Migration risk index across the source database tables

9.5. Hybrid-Cloud Compatibility Assessment

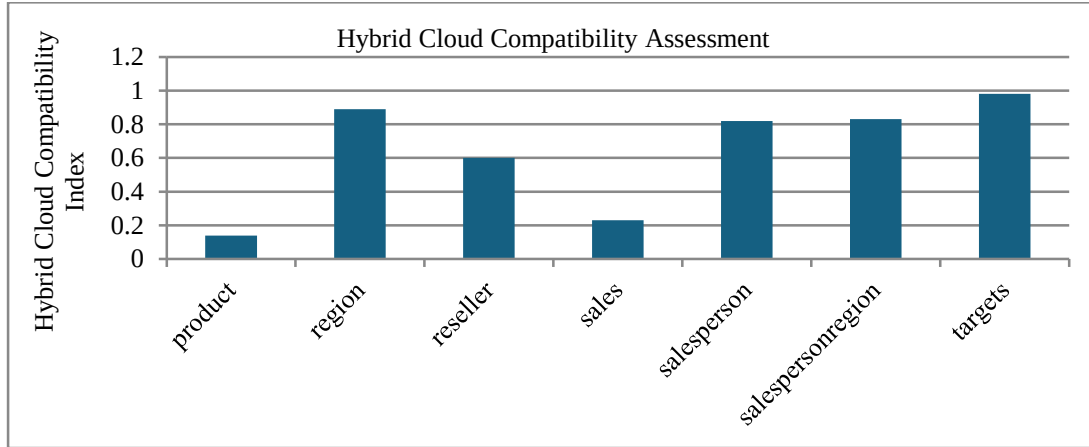
For complex clouds, measurements are taken on the individual tables to determine the quality of the transformations and the risk of migration, scalability and resource efficiency in hybrid clouds. The values returned for HCCI can be considered as an approximation of the 'deployment suitability' of the tables. The Region table was

given an HCCI of 0.8871 and was determined to be Public Cloud Ready. The value of the salesperson also was high, with a correlation value of 0.8241. This reseller was awarded a 0.5997 rating on the HCCI and was determined to be a Hybrid Cloud Ready reseller. Private Cloud Preferred was the same for both Product and Sales, having a compatibility of 0.1405 and 0.2234, respectively.

Table 9. Hybrid-cloud compatibility results

Table	HCCI	Deployment Recommendation
Product	0.1405	Private Cloud Preferred
Region	0.8871	Public Cloud Ready
Reseller	0.5997	Hybrid Cloud Ready
Sales	0.2234	Private Cloud Preferred
Salesperson	0.8241	Public Cloud Ready

This shows the evolution of the migration risk estimation method towards STRIDE-DB-based deployment decision support.

**Fig. 4 Hybrid-cloud compatibility assessment of the source tables**

The Hybrid Cloud Compatibility Index (HCCI) is shown in Figure 4 (for the database tables evaluated). Results reveal the tremendous difference between the deployability of clouds. The Region table is also a good candidate for public cloud employment, with an HCCI of 0.8871, and Salesperson is second with an HCCI of 0.8241. The Reseller gets an HCCI of 0.5997 and is thus considered to be Hybrid Cloud Ready. Product and Sales, meanwhile, have lower scores of HCCI, at 0.1405 and 0.2234, respectively, and would be classified as Private Cloud Preferred. The findings indicate that STRIDE-DB is not making the assumption of just one deployment environment for the database. It does not look at each component in isolation, but rather based on the complexity of the transformation, migration risk, and resource efficiency and scaling aspects of each component. Thus, having recommendations at the table level can assist in cloud deployment planning to be more focused.

9.6. Intelligent Cloud Recommendation

The cloud recommendation module integrates this data to come up with the cloud deployment recommendation, including HCCI, MRI and TCI. It will offer an alternative deployment plan across the entire database system, instead of per cloud environment. It is recommended that you deploy it in Public Cloud, Hybrid Cloud and Private Cloud based on the recommendation repository. Four recommendations of Public Cloud, two recommendations of Private Cloud and one recommendation of Hybrid Cloud were seen in the distribution.

It can be used at the table level, as there may be many different workloads in the legacy databases. This means that in the deployment selection, STRIDE-DB sees this as a table-aware selection, and not a single deployment target for all the database components.

9.7. Migration Validation and Consistency Analysis

The migration validation module validates the consistency of the dependencies, the consistency of mapping the semantics, the quality of the data and the success of the migration. All the dependencies identified were validated and confirmed the semantic validation records. All the tables returned a migration success score of 100 from the validation engine.

The transformation process was the result of validating the migration presented in Figure 5. The validation mechanism checks if tables that are migrated meet the consistency and completeness requirements set in it. The experimental results confirm the approach successfully for all the tested tables and for the validated components, 100% was observed. The result reveals that the following records were found as expected, and the validation checkpoints were met in the migration. This provides further corroboration that there was no discernable row level loss in the baseline experiment. The validation stage then allows the validation of the migration-risk analysis, accompanied by a clear and explicit post-migration validation mechanism.

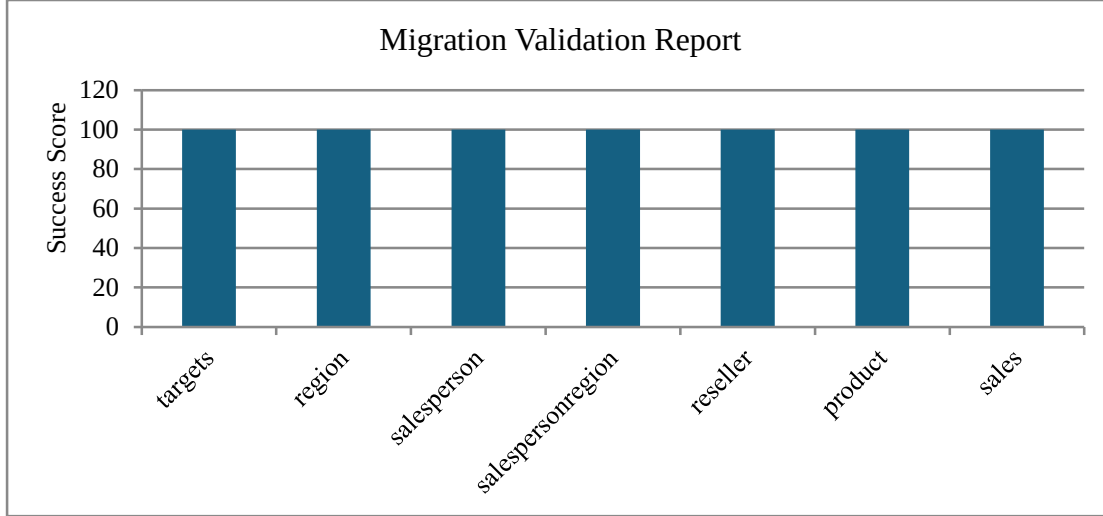


Fig. 5 Migration validation scores across source tables

Table 10. Migration risk prediction performance

Model	Accuracy	Precision	Recall	F1-score
Random Forest	0.50	0.25	0.50	0.3333
Decision Tree	0.50	0.25	0.50	0.3333
SVM	0.50	0.25	0.50	0.3333
KNN	0.50	0.25	0.50	0.3333
Logistic Regression	0.50	0.25	0.50	0.3333
Gradient Boosting	0.50	0.25	0.50	0.3333

9.8. Machine-Learning-based Migration Risk Prediction

Data collected from the machine learning is then cleaned, and a tabular format is created having 7 observations and 8 features. All the data was divided into five and two samples, used for training and testing, respectively. The models that have been tested are: Random Forest Classifier, Decision Tree Classifier, Support Vector Machine Classifier, KNN Classifier, Logistic Regression Classifier and Gradient

Boosting Classifier. The precision of the test results obtained is the same for these models (0.50) as is the weighted precision (0.25, 0.50 and the weighted F1-score is ~0.3333.

Caution should be used with the results, as there are only a few observations. The models were on average 0.80 with 5-fold scores [0.5, 0.5, 1.0, 1.0, 1.0] that provide an insight into the sensitivity of the models due to the limited amount of data.

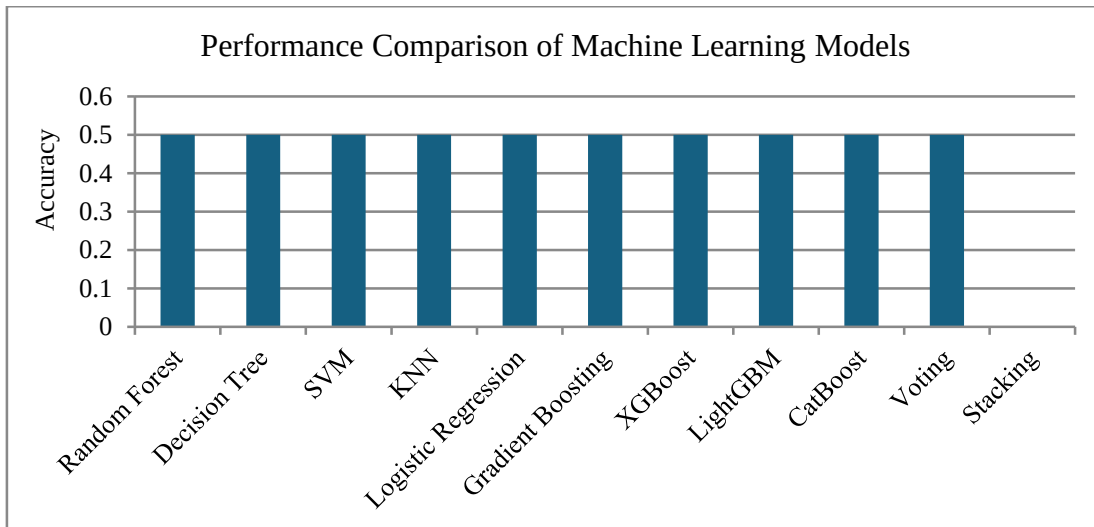


Fig. 6 Comparative accuracy of migration-risk prediction models

The test accuracy of the evaluated machine-learning models – Random Forest, Decision Tree, Support Vector Machine, KNN, Logistic Regression, Gradient Boosting and advanced ensemble models – is compared in Figure 6. The present experiment shows that the test accuracy of the evaluated models is equal to 0.5, which does not show any clear advantage in performance of the models on the present set of data. The interpretation of this result should be taken with a grain of salt, as the machine learning data set has only seven table-level observations, with five observations being used for training and two observations for testing. Thus, the figure mainly illustrates the potential for multiple machine-learning methods to be used in the same risk-prediction pipeline in the context of STRIDE-DB, and does not suggest any definite model superiority. More extensive and detailed migration data are needed to provide statistically sound comparisons.

9.9. Feature Importance Analysis

The model (Random Forest) can order the various factors influencing the migration risk in various ways. The most important features were the following: Number of Columns (around 0.2754), Nullable Columns (around 0.2441), Total Columns (around 0.2188) and Structural Complexity (around 0.2070).

This implies that the properties of structural size/complexity have a more significant influence on the model in the experimental data set analysed in the previous case than the graph-degree properties.

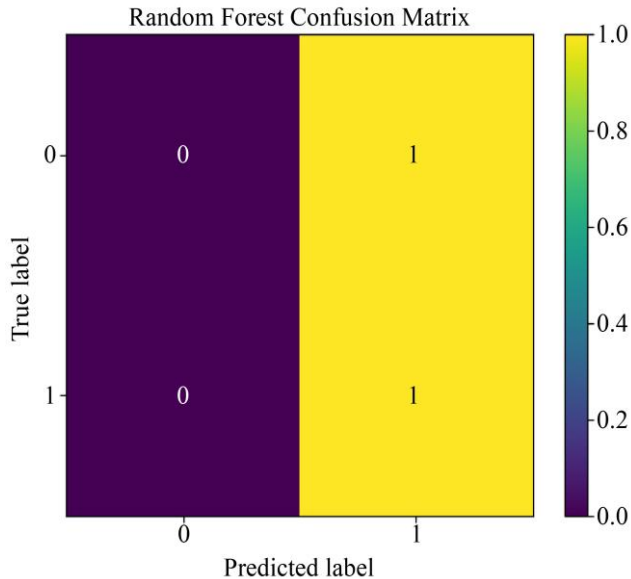


Fig. 7 Random Forest confusion matrix for migration-risk classification

Figure 7. Represent the confusion matrix for the Random Forest classification model for migration risk. The matrix demonstrates the association between the genuine risk and predicted risk labels.

Table 11. Random Forest feature importance

Feature	Importance
NumberOfColumns	0.2754
NullableColumns	0.2441
TotalColumns	0.2188
StructuralComplexity	0.2070
AverageColumnNameLength	0.0547
Degree	0.0000
OutDegree	0.0000
QualityScore	0.0000

9.10. Gradient Boosting Analysis

In addition, the “Gradient Boosting” method was also run along with Hyperparameter Search with random hyperparameters.

The model used had the following parameters: learning rate = 0.2, the number of estimators = 500, the maximum depth = 3, the minimum samples split = 10 and subsample = 0.7.

The test accuracy of the model was 0.50, the precision, recall and F1-scores were 0.25, 0.50 and 0.3333, respectively. All the above features will not be used to calculate the features' importance score for each stage of the Gradient Boosting model. This means that it is not recommended to write an opinionated paper on the significance of the feature, Gradient Boosting.

9.11. Advanced Ensemble Evaluation

The other models tested and analyzed are ensemble models (XGBoost, LightGBM, CatBoost, Voting, Stacking), which were also tested with the conventional models. The accuracy of the models is currently: 0.50, 0.50, 0.50, 0.50, 0.50, 0.50, 0.60, 0.59, 0.59 and 0.59, respectively, for Random Forest, Decision Tree, SVM, KNN, Logistic Regression, Gradient Boosting, XGBoost, LightGBM, CatBoost and Voting. If it is not able to find a successful stacking classifier, then the output of the Stacking will be left at 0.0 since the Stacking is initialized to 0.0.

Value of performance (Note: Do not use value of Stacking = 0.0). A value that is used for a program when it does not have a value. It appears to be more of an issue of what is being implemented, rather than what is the best one.

9.12. Explainable AI Analysis

Analysis based on the trained XGBoost model was performed by SHAP-based analysis, which helped to ease the interpretability of the migration-risk prediction process. Implementation: TreeExplainer is provided, and a standard SHAP summary representation, as well as a bar-type representation, is created.

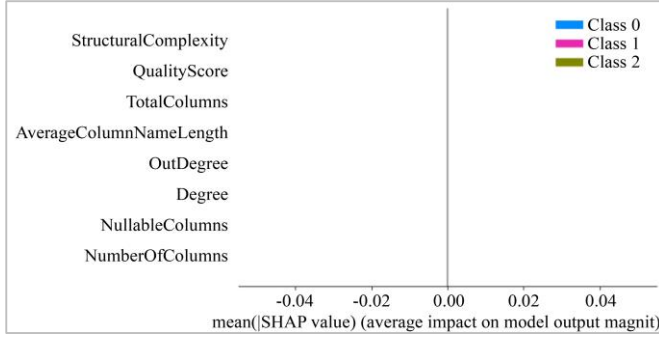


Fig. 8 SHAP-based explanation of migration-risk predictions

The SHAP analysis points out the structural/migration-related attributes to get insights into what contribution these attributes are making to the prediction. The explanation layer could also be used to determine the relative contribution of the input variables as opposed to the risk prediction being a 'black box' classification. The migration-risk prediction model's explainability analysis, based on SHAP, is shown in Figure 8. The following features are taken into account in the figure: StructuralComplexity, QualityScore, TotalColumns, AverageColumnNameLength, OutDegree, Degree, NullableColumns, and NumberOfColumns. These features are related to migration. To make the impact of these features on the model's output more understandable, SHAP analysis is used.

9.13. Summary of Experimental Findings

The experimental results validate the value of STRIDE-DB for analyzing the structure, semantic mapping, migration risk estimation, migration & validation in a single workflow and migration & validation for cloud compatibility. The most direct evidence in the experiments is the number of rows that were migrated (59825 rows) and the number of rows lost (0 rows) which means that there was no row being lost during transformation when migrating from the pre- to the post-transformation table. When it is canonicalised, the number of the source attributes is reduced to 25 target attributes and only 92.11% success rate in the semantic profiling. The various stages of cloud deployment of the tables were captured during the cloud assessment with the public cloud readiness, hybrid cloud readiness and private cloud preferred.

The outcome of the machine learning is a very simple proof of concept – Just 7 observations in the data set. However, it will take a lot of effort to get the ability of an intelligent-risk prediction pipeline to be more predictive, even if the data set is smaller, to prove the pipeline proposed.

10. Discussion

10.1. Interpretation of the Results

The results indicate that it is not enough to perform schema transformation to achieve good migrations. STRIDE-DB deals with this by introducing structural, semantic correspondence, dependency properties, complexity of the

transformations and deployment compatibility. The baseline migration result shows that the transformation pipeline can keep the source records as part of the pipeline as it performs the migration. A semantic analysis also reveals that there are many attributes that belong to the source, which can be represented by the canonical concepts, and would reduce the naming inconsistencies of the source if it were to be used in a transformation. One such example of the other important aspect of the proposed approach can be illustrated by the experiment of using a hybrid cloud. STRIDE-DB will not make a decision at the database level; instead, it will analyze each table and suggest deployment based on the compatibility calculation results of the tables.

10.2. Practical Implications

The framework can be used as a guide for the planning process when undertaking legacy modernization. Specifically created TCI and MRI can be an excellent tool to pre-identify potentially challenging components, BEFORE the migration, specifically. Elements that provide evidence that the re-used representation continues to carry the records in the expected way, and that they are consistent with this interpretation of the attributes. The cloud recommendation layer can then be utilized in the deployment planning procedure.

10.3. Role of Explainability

The other tool that can be utilized as a tool for decision-making is SHAP. The explanation mechanism may be used to learn the relationship between the structural features and the prediction, if the risk of migration is predicted by the machine learning model. Especially so when migrations are planned, a risk score based on no reason is not very helpful. However, there can be an explanation based on the complexity of the structure, and whether there are nullable attributes, the size of the schema, etc.

11. Limitations, Threats to Validity and Future Work

11.1. Dataset Size

The biggest problem is that there are not many observations at the table level to perform the machine learning-based risk prediction. There are only seven observations in the refined data set; five of which were used for training and two for testing. Thus, there is no guarantee that the use of this test will result in a predictor of the generalizable predictive accuracy. It is really a preliminary effort of the proposed risk-prediction pipeline at this point in time.

11.2. Dataset Diversity

In this experiment, a structures legacy database will be built, based on the Adventure Works legacy database. Other databases in other domains need to be used to evaluate the effectiveness of the framework in other schemas, names and workloads.

11.3. Machine-Learning Generalization

All models tested have the same test accuracy (0.50), so there is not enough data to determine which model is best. More detailed data is required – including multiple projects and a more detailed risk characterization for good model comparison.

11.4. Controlled Experimental Scenarios

Scenarios that are controlled are called ‘stress scenarios’ and may be used in experiments. This data-quality experiment “stress” changes only a portion of the data, rather than all the potential data-quality problems that may occur in the real world. Therefore, it is suggested that more research should be conducted in a real-world migration project, and different database platforms should be taken into STRIDE-DB.

11.5. Future Research

The future research directions will be focused on:

1. Incorporating more than one legacy database into the data set.
2. Gathering more observations of migrations.
3. Adding true failure of data migration.
4. Talk about alternate ways of access to the database.
5. improving dependency-aware transformation.
6. Explain what really takes place in real deployments to provide cloud advice.
7. The use of machine learning and explainability in more in-depth experiments.

12. Conclusion

As part of this paper, we propose an intelligent approach to database evolution of legacy databases, called structured

and risk-aware (STRIDE-DB). It includes the following features: metadata discovery, dependency analysis, semantic schema mapping, estimation of the complexity of migrations, risk assessment of migrations, compatibility analysis for hybrid-cloud and migration planning and validation.

The results of the experimental evaluation were successful: 59825 records were migrated, and there were no indications of any preservation errors or transformation errors, and the entire row was preserved. The semantic component accounted for a total of 92.11% of the attributes, and the canonicalization decreased the number of sources to 25 different target attributes. The cloud compatibility experiment also showed that it is possible to suggest that different parts of the database are cloud compatible and can be deployed separately, based on the compatibility properties of these parts, computed during the experiment. Migration of the tables has been done without any inconsistencies as per the validation module.

The machine learning explainability features come as another opportunity for further intelligent risk prediction on the migration journey. Preliminary results should be taken with caution, as there is a relatively small amount of data in the current ML data set, and larger and more diverse data sets on migration will increase the use of Machine Learning algorithms. Overall, the study is comprehensive, considering not only the structure of the database but also its semantics and the complexity of migration and the risk of deploying it as a database. This results in the basis for the development of more generic and well-tested intelligent database migration systems.

References

- [1] Inwon Kang et al., “SemStruct: Contextualizing Semantic Embeddings with Structural Information for Schema Matching,” *Proceedings of the 32nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.2.*, Jeju Island Republic of Korea, pp. 2261-2272, 2026. [[CrossRef](#)] [[Google Scholar](#)] [[Publisher Link](#)]
- [2] Roque Lopez et al., “BDI-Kit: An AI-Powered Toolkit for Biomedical Data Harmonization,” *Patterns*, vol. 7, no. 4, pp. 1-11, 2026. [[CrossRef](#)] [[Google Scholar](#)] [[Publisher Link](#)]
- [3] Yunong Tian, Ning Wang, and Anshun Zhou, “CrossER: A Robust and Adaptable Generalized Entity Resolution Framework for Diverse and Heterogeneous Datasets,” *Information Systems*, vol. 135, 2026. [[CrossRef](#)] [[Google Scholar](#)] [[Publisher Link](#)]
- [4] Dogga Aswani et al., “Effective Cloud Data Management by Using AES Encryption and Decryption,” *Advances in Machine Learning and Big Data Analytics II*, Arunachal Pradesh, India, pp. 165-174, 2025. [[CrossRef](#)] [[Google Scholar](#)] [[Publisher Link](#)]
- [5] Yu Zhang et al., “SMUTF: Schema Matching using Generative Tags and Hybrid Features,” *Information Systems*, vol. 133, pp. 1-43, 2025. [[CrossRef](#)] [[Google Scholar](#)] [[Publisher Link](#)]
- [6] Beatriz Pérez, Ángel Luis Rubio Garcia, and María A. Zapata, “PROV-IDEA: Supporting Interoperable Schema and Data Provenance within Database Evolution,” *ACM Transactions on Software Engineering and Methodology*, vol. 34, no. 3, pp. 1-27, 2025. [[CrossRef](#)] [[Google Scholar](#)] [[Publisher Link](#)]
- [7] Bilal Abu-Salih et al., “Using Large Language Models for Semantic Interoperability: A Systematic Literature Review,” *ICT Express*, vol. 11, no. 4, pp. 819-837, 2025. [[CrossRef](#)] [[Google Scholar](#)] [[Publisher Link](#)]
- [8] Hakju Oh et al., “Employing Word-Embedding for Schema Matching in Standard Lifecycle Management,” *Journal of Industrial Information Integration*, vol. 38, pp. 1-29, 2024. [[CrossRef](#)] [[Google Scholar](#)] [[Publisher Link](#)]
- [9] Neha Reddy Palnati, Vijay Kumar Reddy Julakanti, and Nikhil Bayyavarapu, “Leveraging Machine Learning for Enhanced Database Integration,” *Procedia Computer Science*, vol. 235, pp. 1623-1633, 2024. [[CrossRef](#)] [[Google Scholar](#)] [[Publisher Link](#)]

- [10] I Made Putrama, and Péter Martinek, “Heterogeneous Data Integration: Challenges and Opportunities,” *Data in Brief*, vol. 56, pp. 1-23, 2024. [[CrossRef](#)] [[Google Scholar](#)] [[Publisher Link](#)]
- [11] Nishan Mills et al., “A Cloud-based Architecture for Explainable Big Data Analytics using Self-Structuring Artificial Intelligence,” *Discover Artificial Intelligence*, vol. 4, no. 1, pp. 1-13, 2024. [[CrossRef](#)] [[Google Scholar](#)] [[Publisher Link](#)]
- [12] Francisco Javier Maldonado Carrascosa et al., “Multi-Objective Optimization of Virtual Machine Migration Among Cloud Data Centers,” *Soft Computing*, vol. 28, no. 20, pp. 12043-12060, 2024. [[CrossRef](#)] [[Google Scholar](#)] [[Publisher Link](#)]
- [13] Harald Foidl et al., “Data Pipeline Quality: Influencing Factors, Root Causes of Data-Related Issues, and Processing Problem Areas for Developers,” *Journal of Systems and Software*, vol. 207, pp. 1-17, 2024. [[CrossRef](#)] [[Google Scholar](#)] [[Publisher Link](#)]
- [14] Md Asif-Ur-Rahman et al., “A Semi-Automated Hybrid Schema Matching Framework for Vegetation Data Integration,” *Expert Systems with Applications*, vol. 229, pp. 1-10, 2023. [[CrossRef](#)] [[Google Scholar](#)] [[Publisher Link](#)]
- [15] Rihem Nasfi, Antoon Bronselaer, and Guy De Tré, “A Novel Approach to Assess and Improve Syntactic Interoperability in Data Integration,” *Information Processing & Management*, vol. 60, no. 6, 2023. [[CrossRef](#)] [[Google Scholar](#)] [[Publisher Link](#)]
- [16] Pablo Suárez-Otero et al., “CoDevo: Column Family Database Evolution using Model Transformations,” *Journal of Systems and Software*, vol. 203, pp. 1-16, 2023. [[CrossRef](#)] [[Google Scholar](#)] [[Publisher Link](#)]
- [17] Yunjia Zhang et al., “Schema Matching using Pre-Trained Language Models,” *2023 IEEE 39th International Conference on Data Engineering (ICDE)*, Anaheim, CA, USA, pp. 1558-1571, 2023. [[CrossRef](#)] [[Google Scholar](#)] [[Publisher Link](#)]
- [18] Muhammad Hafiz Hasan et al., “Legacy Systems to Cloud Migration: A Review from the Architectural Perspective,” *Journal of Systems and Software*, vol. 202, 2023. [[CrossRef](#)] [[Google Scholar](#)] [[Publisher Link](#)]
- [19] Cinzia Daraio, Simone Di Leo, and Monica Scannapieco, “Accounting for Quality in Data Integration Systems: A Completeness-Aware Integration Approach,” *Scientometrics*, vol. 127, pp. 1465-1490, 2022. [[CrossRef](#)] [[Google Scholar](#)] [[Publisher Link](#)]
- [20] Diego Rodrigues, and Altigran da Silva, “A Study on Machine Learning Techniques for the Schema Matching Network Problem,” *Journal of the Brazilian Computer Society*, vol. 27, no. 1, pp. 1-29, 2021. [[CrossRef](#)] [[Google Scholar](#)] [[Publisher Link](#)]
- [21] Artur Kulmukhametov, Andreas Rauber, and Christoph Becker, “Improving Data Quality in Large-Scale Repositories through Conflict Resolution,” *International Journal on Digital Libraries*, vol. 22, no. 4, pp. 365-383, 2021. [[CrossRef](#)] [[Google Scholar](#)] [[Publisher Link](#)]
- [22] Erik Tute, Irina Scheffner, and Michael Marscholke, “A Method for Interoperable Knowledge-based Data Quality Assessment,” *BMC Medical Informatics and Decision Making*, vol. 21, pp. 1-14, 2021. [[CrossRef](#)] [[Google Scholar](#)] [[Publisher Link](#)]
- [23] Carsten Oliver Schmidt et al., “Facilitating Harmonized Data Quality Assessments. A Data Quality Framework for Observational Health Research Data Collections with Software Implementations in R,” *BMC Medical Research Methodology*, vol. 21, no. 1, pp. 1-15, 2021. [[CrossRef](#)] [[Google Scholar](#)] [[Publisher Link](#)]
- [24] Chinmaya Kumar Dehury, Satish Narayana Srirama, and Tek Raj Chhetri, “CCoDaMiC: A Framework for Coherent Coordination of Data Migration and Computation Platforms,” *Future Generation Computer Systems*, vol. 109, pp. 1-16, 2020. [[CrossRef](#)] [[Google Scholar](#)] [[Publisher Link](#)]
- [25] Tanvi Sahay, Ankita Mehta, and Shruti Jadon, “Schema Matching using Machine Learning,” *2020 7th International Conference on Signal Processing and Integrated Networks (SPIN)*, Noida, India, pp. 359-366, 2020. [[CrossRef](#)] [[Google Scholar](#)] [[Publisher Link](#)]
- [26] Kenji Nozaki, Teruhisa Hochin, and Hiroki Nomiya, “Semantic Schema Matching for String Attribute with Word Vectors and its Evaluation,” *International Journal of Networked and Distributed Computing*, vol. 7, no. 3, pp. 100-106, 2019. [[CrossRef](#)] [[Google Scholar](#)] [[Publisher Link](#)]
- [27] Martyn Ellison, Radu Calinescu, and Richard F. Paige, “Evaluating Cloud Database Migration Options using Workload Models,” *Journal of Cloud Computing*, vol. 7, no. 1, pp. 1-18, 2018. [[CrossRef](#)] [[Google Scholar](#)] [[Publisher Link](#)]
- [28] Hassan Alrehamy, and Coral Walker, “SemLinker: Automating Big Data Integration for Casual Users,” *Journal of Big Data*, vol. 5, no. 1, pp. 1-26, 2018. [[CrossRef](#)] [[Google Scholar](#)] [[Publisher Link](#)]
- [29] Phokion G. Kolaitis et al., “Limits of Schema Mappings,” *Theory of Computing Systems*, vol. 62, no. 4, pp. 899-940, 2018. [[CrossRef](#)] [[Google Scholar](#)] [[Publisher Link](#)]
- [30] Mahdi Fahmideh, and Ghassan Beydoun, “Reusing Empirical Knowledge during Cloud Computing Adoption,” *Journal of Systems and Software*, vol. 138, pp. 124-157, 2018. [[CrossRef](#)] [[Google Scholar](#)] [[Publisher Link](#)]
- [31] Francisco Javier Bermúdez Ruiz, Jesús García Molina, and Oscar Díaz García, “On the Application of Model-Driven Engineering in Data Reengineering,” *Information Systems*, vol. 72, pp. 136-160, 2017. [[CrossRef](#)] [[Google Scholar](#)] [[Publisher Link](#)]
- [32] Mahdi Fahmideh Gholami et al., “Challenges in Migrating Legacy Software Systems to the Cloud — An Empirical Study,” *Information Systems*, vol. 67, pp. 100-113, 2017. [[CrossRef](#)] [[Google Scholar](#)] [[Publisher Link](#)]
- [33] Mahdi Fahmideh Gholami et al., “Cloud Migration Process—A Survey, Evaluation Framework, and Open Challenges,” *Journal of Systems and Software*, vol. 120, pp. 31-69, 2016. [[CrossRef](#)] [[Google Scholar](#)] [[Publisher Link](#)]
- [34] Erik Ferragut, and Jason Laska, “Nonparametric Bayesian Modeling for Automated Database Schema Matching,” *2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA)*, Miami, FL, USA, pp. 82-88, 2015. [[CrossRef](#)] [[Google Scholar](#)] [[Publisher Link](#)]

- [35] Jie He, and Wei Wang, “An Integrated Heterogeneous Web Service Retrieval via Combination of Instance- and Metadata-based Schema Matching Method,” *Geo-Spatial Information Science*, vol. 18, no. 2-3, pp. 111-123, 2015. [[CrossRef](#)] [[Google Scholar](#)] [[Publisher Link](#)]
- [36] Tho Thanh Quan et al., “Argumentation-based Schema Matching for Multiple Digital Libraries,” *Online Information Review*, vol. 39, no. 1, pp. 81-103, 2015. [[CrossRef](#)] [[Google Scholar](#)] [[Publisher Link](#)]
- [37] Y.H. Wang, H.B. Zhang, and J. Xu, “A Survey of Applications and Researches on Schema Matching between GIS Spatial Data,” *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. XL-7/W4, pp. 175-179, 2015. [[CrossRef](#)] [[Google Scholar](#)] [[Publisher Link](#)]
- [38] Algorismus, Adventure Works 2022 CSVs, Kaggle, 2022. [Online]. Available: <https://www.kaggle.com/datasets/algorismus/adventure-works-in-excel-tables>.
- [39] Daomin Ji et al., “Table Integration in Data Lakes Unleashed: Pairwise Integrability Judgment, Integrable Set Discovery, and Multi-Tuple Conflict Resolution,” *The VLDB Journal*, vol. 34, no. 3, pp. 1-24, 2025. [[CrossRef](#)] [[Google Scholar](#)] [[Publisher Link](#)]
- [40] Fan Zhang et al., “A Multi-Strategy Ontology Mapping Method based on Cost-Sensitive SVM,” *Journal of Cloud Computing*, vol. 13, no. 1, pp. 1-14, 2024. [[CrossRef](#)] [[Google Scholar](#)] [[Publisher Link](#)]
- [41] Inès Osman et al., “Uncertainty in Automated Ontology Matching: Lessons from an Empirical Evaluation,” *Applied Sciences*, vol. 14, no. 11, pp. 1-19, 2024. [[CrossRef](#)] [[Google Scholar](#)] [[Publisher Link](#)]
- [42] Kareem S. Aggour et al., “Semantics-Enabled Data Federation: Bringing Materials Scientists Closer to FAIR Data,” *Integrating Materials and Manufacturing Innovation*, vol. 13, pp. 420-434, 2024. [[CrossRef](#)] [[Google Scholar](#)] [[Publisher Link](#)]
- [43] Huijie Cao et al., “Aion: Live Migration for In-Memory Databases with Zero Downtime and Reduced Redundant Data Transfer,” *Data Science and Engineering*, vol. 10, no. 2, pp. 212-229, 2025. [[CrossRef](#)] [[Google Scholar](#)] [[Publisher Link](#)]
- [44] John Agyekum, Somnath Mazumdar, and Christoph Scheich, “ADAPT: An Effective Data-Aware Multicloud Data Placement Framework,” *Cluster Computing*, vol. 28, no. 13, pp. 1-17, 2025. [[CrossRef](#)] [[Google Scholar](#)] [[Publisher Link](#)]
- [45] Vikas Hassija et al., “Interpreting Black-Box Models: A Review on Explainable Artificial Intelligence,” *Cognitive Computation*, vol. 16, no. 1, pp. 45-74, 2024. [[CrossRef](#)] [[Google Scholar](#)] [[Publisher Link](#)]