Assessing the Viability of an OBDA Approach to Integrating Databases into a Virtual Knowledge Graph for Research Management at ITC

Phearun Ol[1,*], Deperias Kerre[2], Christophe Debruyne[2,*], Dona Valy[3] and Sotheany Nou[3]

[1]Graduate School, Institute of Technology of Cambodia, Phnom Penh, Cambodia
[2]Montefiore Institute, University of Liège, 4000 Liège, Belgium
[3]Research and Innovation Center, Institute of Technology of Cambodia, Phnom Penh, Cambodia

Abstract

Institutional research management depends on data that is often distributed across independent operational systems. At the Institute of Technology of Cambodia (ITC), relevant information about students, employees, departments, academic records, researchers, projects, profiles, and publications is spread across several databases. This paper reports an early-stage, applied assessment of an ontology-based data access approach for integrating these sources into a virtual knowledge graph. The implementation uses Trino to federate PostgreSQL databases and Ontop to expose RDF views through a SPARQL endpoint, while preserving the existing source systems. The ontology models core institutional entities and relationships, including cross-system links between employees and researcher identities. Initial mapping demonstrates integration of student academic records and research management. Preliminary SPARQL tests show that the approach can answer integrated questions over academic and research data without copying operational data into a new repository. The work suggests that OBDA is a viable and cost-conscious foundation for institutional knowledge graph development at ITC, while also revealing challenges in identity alignment, schema harmonization, endpoint performance, and long-term governance.

Keywords

Research Management System, Ontology-based Data Access, Virtual Knowledge Graph, Data Integration

1. Introduction

Research institutions increasingly need integrated views of their activities, outputs, expertise, and resources. For universities and institutes, this information is rarely contained in a single system. At the Institute of Technology of Cambodia (ITC), this problem appears in the coexistence of several information systems. ITC runs three main systems: SMIS (students, staff, departments, academic records), the Student Portal (student-facing academic data), and RMIS/RIC (research profiles, projects, and publications).[1] These systems are valuable individually, but their separation limits the visibility and reuse of institutional knowledge.

This work explores whether Ontology-Based Data Access (OBDA) [1] can provide a practical integration layer for these systems. In this way, a virtual knowledge graph (VKG) can support integrated access while keeping the operational systems in place. This is applied, systems-oriented work rather than a methodological contribution: the paper does not propose new OBDA techniques, but reports a concrete, real-world feasibility assessment of an existing OBDA/VKG stack (Trino, Ontop) deployed over ITC's institutional databases. It presents the source integration architecture, the mapping scope, and early evaluation results from SPARQL tests over student academic records and SMIS-to-RMIS research links, offering evidence and lessons for similar deployments at other institutions.

This work is part of the ARES Institutional Support Program 2022–2027 at the Institute of Technology of Cambodia (ITC). A key objective is to make ITC research outputs more visible, reusable, and valuable

Posters, Demos, Blue Sky, and Tutorials at SEMANTiCS 2026, Sep 2026, Ghent, Belgium
*Corresponding authors
ol.phearun@itc.edu.kh (P. Ol); c.debruyne@uliege.be (C. Debruyne)
0009-0005-3355-0134 (P. Ol); 0000-0002-7437-6735 (D. Kerre); 0000-0003-4734-3847 (C. Debruyne); 0009-0000-6181-5551 (D. Valy); 0009-0006-3659-3305 (S. Nou)

[1]The first two are not accessible from outside Cambodia.

for Cambodian society. The program emphasizes the reuse and coordination of existing institutional structures, careful budget monitoring, and sustainable strengthening of ITC capacities. In this context, VKGs offer a "cost-conscious" approach to data integration, as they can provide a conceptual layer over existing databases without requiring the replacement of current systems. At the same time, the study contributes to upskilling ITC staff in semantic modeling, data integration, and KGs.

2. Background

Related work on VKG [2] and OBDA [1] shows that semantic integration can be achieved without physically materializing all source data as Resource Description Framework (RDF). Systems such as Ontop [2] translate SPARQL queries into SQL using ontology mappings, allowing existing relational databases to be queried through a semantic layer. This approach is suitable for institutional environments where data is distributed across operational systems such as student management, academic portals, and research management platforms. In parallel, federated SQL engines such as Trino provide a practical way to query multiple relational databases through a single SQL interface. However, previous work also highlights that performance in federated and OBDA systems depends on query rewriting and optimization [3], mapping complexity [4], network latency [5], and source database performance.

3. Method

This study follows a VKG methodology to integrate institutional data from SMIS, Portal, and RMIS/RIC without physically transforming all records into RDF.[2] The method consists of four main steps: source analysis, ontology design, mapping implementation, and query validation.

Source analysis In SMIS, employee and academic data are stored in tables such as `users`, `employees`, `departments`, `students`, and `studentannuals`. Portal student data is conceptually linked to SMIS primarily via the student identity card number, `id_card`. RMIS/RIC research data is connected to SMIS through the bridge table `ric_pro`, which contains `smis_user_id`, `smis_employee_id`, and `ric_user_id`.

Instead of defining one global canonical identifier for all systems, the integration uses source-specific identifiers and bridge columns. For example, SMIS employee resources are identified by `employee.id`, Portal students are matched to SMIS students by `id_card` and linked to SMIS employees through `smis_employee_id` or the fallback key `smis_user_id`.

Ontology design The ontology was designed to provide a shared model across the different institutional systems. The ontology covers `Person`, `Employee`, `Student`, `Researcher`, `Department`, `Branch`, `AcademicRecord`, `ResearchProject`, `ResearchProfile`, and `ResearchArticle`, linked by properties such as `hasDepartment`, `hasAcademicRecord`, `hasResearchIdentity`, and `participatesInProject`, with datatype properties for names, emails, and source-system identifiers. Although an OWL 2 ontology was developed for this study, it does not extend beyond RDFS semantics. The focus of this study was assessing the viability of the OBDA approach itself rather than ontology engineering; a purpose-built ontology allowed the mappings to be developed and iterated quickly during the feasibility assessment.[3]

Mapping implementation Mappings were authored in the native Ontop mapping language (OBDA mappings) rather than R2RML. We initially tried R2RML mappings, but the OBDA syntax proved faster to write and iterate on for this study, particularly for the string templates and conditional joins needed across SMIS, Portal, and RMIS/RIC. For SMIS employee data, mappings create `Employee` resources and

[2]The ontology, mappings, and benchmark queries are available at https://github.com/Phearun2020/itc-rmis-kg.

[3]Reusing an established vocabulary such as VIVO, which already models similar concepts (e.g., `Person`, `Student`), was out of scope at this stage and is left for future work, contingent on the approach proving viable.

link them to departments and branches. For Portal and SMIS student integration, the mapping joins Portal students with SMIS students using `id_card`, then links the resulting student resources to the academic records from `studentannuals`. For RMIS/RIC integration, mappings use the `ric_pro` bridge table to connect SMIS employees with RMIS/RIC researcher identities, projects, profiles, and articles.

All identifiers used in IRI templates were explicitly cast to a string value to avoid datatype conflicts during Ontop [2] query rewriting and Trino union operations.

Query and validation workflow The validation workflow was performed at two levels: SQL source validation and SPARQL endpoint validation. First, SQL test files were executed directly through Trino to confirm that the source joins produced the expected records. These tests included SMIS employee mapping, Portal-SMIS student joins, and SMIS-RMIS/RIC researcher links. Second, SPARQL test queries were executed through the Ontop endpoint [2]. The queries validated that RDF resources and relationships could be retrieved correctly from the VKG. Separate SPARQL tests were created for SMIS-only data, SMIS with Portal academic records, and SMIS with RMIS/RIC research data.

4. Demonstration and Evaluation

The VKG was demonstrated through the Ontop SPARQL endpoint and Ontodia [6] (see Figure 1). The demonstration showed that the system can answer integrated queries across multiple institutional databases. E.g., a single SPARQL query can retrieve an SMIS employee, the corresponding RMIS/RIC researcher identity, research project, and research article. Similarly, student queries can connect Portal identities with SMIS academic records, departments, degrees, grades, and academic years.

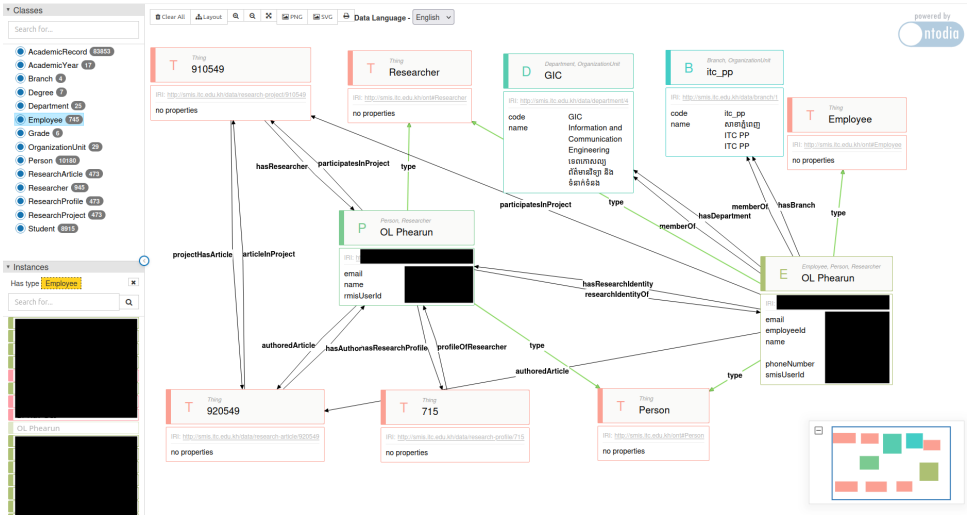

Figure 1: Ontodia visualization of the VKG, illustrating cross-system semantic links between an SMIS employee, their academic department (GIC), and corresponding RMIS/RIC research entities.

Experiment 1: Access via VPN

The experiment was designed to measure the effect of the number of Trino workers on SPARQL query performance. Three query groups were selected to represent the main integration scenarios: a query that focuses on SMIS-only data, a query that combines SMIS and Portal data, and a query that combines SMIS and RMIS/RIC (researcher, project, and article links) data. We executed three SPARQL query cases across 1 to 7 Trino workers in a Docker environment. For each worker configuration, warm-up runs were performed before the measured runs. The measured executions were randomized within each worker configuration, and each query was performed 40 times. The machine is an AMD Ryzen 7 5700G

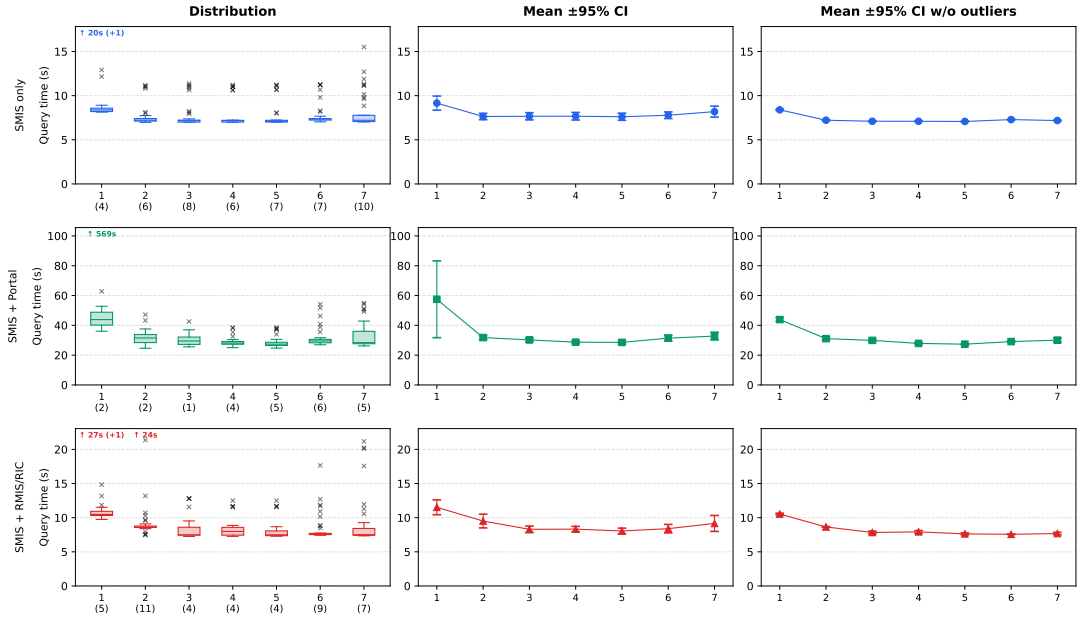

Figure 2: SPARQL query execution time as a function of Trino worker count for three query configurations. The number of outliers is shown in parentheses.

8-core/16-thread processor, 128 GB DDR4-3200 RAM, a 2 TB NVMe SSD, running Windows 11 Pro. The workers were connected to the real PostgreSQL databases through VPN.

In Figure 2, each row corresponds to one query case. The left column shows the full distribution as box plots (median, interquartile range, whiskers at $1.5 \times IQR$); the number in parentheses below each box indicates the number of statistical outliers for that configuration. The center column shows the mean with 95% confidence intervals computed over all runs. The right column shows the same statistic after removing per-group outliers.

The results show that increasing the number of workers improves query execution, but the improvement is not linear. As observed in distributed SQL architectures, network latency and data shuffling overhead can outweigh the benefits of parallelism [7]. Based on the benchmark results, three workers appear to perform best, capturing most of the parallelism gains in our setup. It is worth noting that in this setup, the available computing budget on an 8-core machine with 16 threads limits the ceiling for worker parallelism; beyond 3 workers, the marginal gain is negligible. What we observe here is likely a bottleneck due to the VPN connection.[4]

Experiment 2: Direct Access without VPN

To separate the effect of network latency from Trino's own distributed-execution overhead, we repeated the same three query cases and worker configurations (1–7 workers, 40 runs each) from a machine with direct, low-latency access to the source databases (average round-trip time to each PostgreSQL source was 0.9 ms with 0% packet loss). Figure 3 reports the results using the same layout as Figure 2.[5]

Removing the VPN reduced execution times by more than an order of magnitude: the SMIS + Portal case dropped from 30–40s to 1.1–1.2s for 1–6 workers. However, the scaling trend reversed. For the SMIS-only and SMIS + RMIS/RIC cases, one worker was consistently the fastest configuration, with execution time increasing monotonically up to seven workers (0.43s→0.55s and 0.53s→0.84s, respectively). The SMIS + Portal case stayed roughly flat across 1–6 workers (1.13–1.21s), with a single outlier at seven workers (22.5s of 40 runs), most likely a transient hiccup rather than a systematic effect.

[4]The extreme 569s data point for SMIS + Portal on one worker is almost certainly due to a VPN dropout.

[5]The experiments were conducted on a KVM (QEMU) virtual machine rather than a dedicated physical machine, so no specific hardware brand or model applies. The VM was configured with 16 vCPUs (host CPU: Intel® Xeon® Gold 6338 @ 2.00 GHz) and 31 GB RAM, running Ubuntu 24.04.4 LTS (kernel 6.8.0-136-generic).

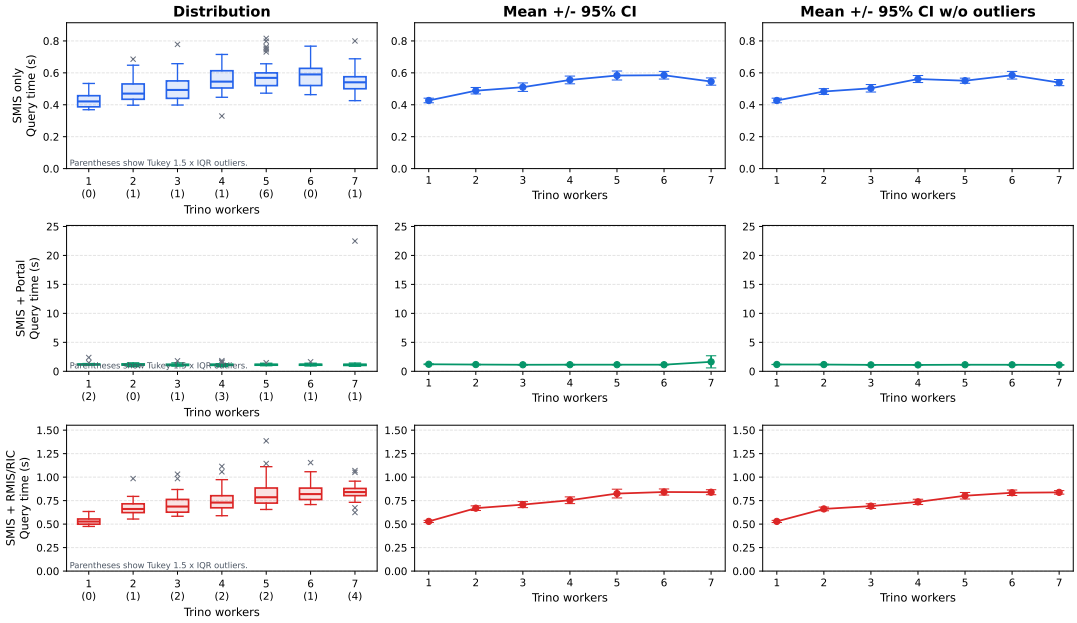

Figure 3: SPARQL query execution time as a function of Trino worker count, repeated with direct (non-VPN) access to the source databases.

4.1. Discussion

Important: Experiment 1 was conducted from Belgium, on hardware available to the authors there, and therefore required a VPN connection to reach the source databases in Cambodia. Experiment 2 was conducted with direct, low-latency access to the source databases, but the authors did not have access to the same or similar machine while in Cambodia, running instead on a KVM virtual machine. The two experiments therefore differ in network path, available computing power, and virtualization: hypervisor scheduling and virtualized I/O on the VM used in Experiment 2 are plausible additional contributors to the observed overhead of adding workers.

The evaluation shows that the proposed architecture is effective for integrating the databases with a VKG for ITC, but the two experiments reveal different bottlenecks. Under VPN, network latency dominated: adding workers helped up to a point, with three workers capturing most of the parallelism gains within the constraints of the benchmarking machine. Once the VPN was removed, overall latency dropped by more than an order of magnitude, but a single worker became optimal, and adding workers monotonically ”hurt” performance for two of the three query cases. This is consistent with findings from distributed SQL engines, where cross-node task scheduling and data-shuffling overhead can outweigh the benefits of parallelism [7]: once network latency no longer masks this overhead, it becomes the dominant cost for these small, sub-second queries.

The complexity of the SQL generated by Ontop’s mapping-based query rewriting is itself also a known contributor to VKG query latency, independent of data volume or network conditions; this has motivated alternative architectures, such as the intermediate triple table proposed by [8], aimed at reducing this rewriting overhead. VKGs are also usually slower than materialized RDF graph databases [9]. Therefore, the VKG approach is suitable for read-only integration and semantic access to live institutional data, but performance-sensitive use cases may require other solutions.

5. Conclusions

This project demonstrated how a Virtual Knowledge Graph can integrate heterogeneous institutional data from SMIS, Portal, and RMIS/RIC. Overall, the results show that a VKG approach is viable for read-only semantic integration of live institutional databases, providing a flexible and explainable way

to query distributed data through a shared ontology. Query latency is strongly dependent on network access to the source databases: over VPN, wider queries took 10–40s, but without a VPN the same queries completed in sub-second to low-second time. A proper comparison with a materialized graph is still needed; materialization was not allowed in this study. Future work should focus on improving performance, extending the ontology (including evaluating reuse of and alignment with established vocabularies such as VIVO), and further characterizing the workload-dependent trade-off between distributed and single-node execution observed in Experiment 2. A hybrid architecture that combines virtual access for fresh operational data and materialized graph storage for frequently accessed data [2] may provide a better balance between freshness and performance.

Acknowledgments

This work was supported by the ARES (Académie de Recherche et d'Enseignement Supérieur) Institutional Support Program 2022-2027 at the Institute of Technology of Cambodia (ITC).

Declaration on Generative AI

During the preparation of this manuscript, the authors used ChatGPT and Grammarly to assist with language polishing, text refinement, and shortening paragraphs. Claude was used to create the scripts to generate plots from CSV files produced by the experiment. After using these tools, the authors reviewed and edited the content as needed and take full responsibility for the publication's content.

References

[1] D. Calvanese, G. De Giacomo, D. Lembo, M. Lenzerini, R. Rosati, et al., Ontology-based data access and integration, in: Encyclopedia of database systems, Springer, 2018, pp. 2590–2596.

[2] G. Xiao, D. Lanti, R. Kontchakov, S. Komla-Ebri, E. Güzel-Kalaycı, L. Ding, J. Corman, B. Cogrel, D. Calvanese, E. Botoeva, The virtual knowledge graph system ontop, in: International Semantic Web Conference, Springer, 2020, pp. 259–277.

[3] D. Calvanese, B. Cogrel, S. Komla-Ebri, R. Kontchakov, D. Lanti, M. Rezk, M. Rodriguez-Muro, G. Xiao, Ontop: Answering SPARQL queries over relational databases, Semantic Web 8 (2017) 471–487.

[4] D. Hovland, D. Lanti, M. Rezk, G. Xiao, OBDA constraints for effective query answering, in: Rule Technologies. Research, Tools, and Applications - 10th International Symposium, RuleML 2016, Stony Brook, NY, USA, July 6-9, 2016. Proceedings, LNCS, Springer, 2016, pp. 269–286.

[5] B. Quilitz, U. Leser, Querying distributed RDF data sources with SPARQL, in: The Semantic Web: Research and Applications, 5th European Semantic Web Conference, ESWC 2008, Tenerife, Canary Islands, Spain, June 1-5, 2008, Proceedings, LNCS, Springer, 2008, pp. 524–538.

[6] D. Mouromtsev, D. S. Pavlov, Y. Emelyanov, A. V. Morozov, D. S. Razdyakonov, M. Galkin, The simple web-based tool for visualization and sharing of semantic data and ontologies, in: Proc. of the ISWC 2015 Posters & Demonstrations Track co-located with the 14th International Semantic Web Conference (ISWC-2015), Bethlehem, PA, USA, October 11, 2015, CEUR-WS.org, 2015.

[7] R. Sethi, M. Traverso, D. Sundstrom, D. Phillips, W. Xie, Y. Sun, N. Yegitbasi, H. Jin, E. Hwang, N. Shingte, C. Berner, Presto: Sql on everything, in: 2019 IEEE 35th International Conference on Data Engineering (ICDE), IEEE, 2019, pp. 1802–1813.

[8] J. Arenas-Guerrero, Ó. Corcho, M. S. Pérez, Intermediate triple table: A general architecture for virtual knowledge graphs, Knowl. Based Syst. 314 (2025) 113179. doi:10.1016/J.KNOSYS.2025.113179.

[9] G. Xiao, D. Calvanese, R. Kontchakov, D. Lembo, A. Poggi, R. Rosati, M. Zakharyaschev, Ontology-based data access: A survey, in: Proc. of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden, 2018, pp. 5511–5519.