# ForBackBench: From Database to Semantic Web Mappings and Back

Afnan Alhazmi, Jaime Osvaldo Salas and George Konstantinidis

[1]*University of Southampton, University Road, Southampton, SO17 1BJ, United Kingdom*

**Abstract**

Data Integration/Exchange (DI/DE) has seen great developments both in the Databases (DB) and the Semantic Web (SW) areas. ForBackBench has been the first framework that facilitates the comparison of systems from both areas by introducing datasets, converters and algorithms that allow for integration and interaction of different query rewriting (OBDA) and forward-chaining (Chase) systems. In this paper, we discuss the most recent developments of the framework, which aim to completely bridge the gap between the DB and SW areas in DI/DE field by integrating RML (and R2RML) ontology materialisation systems. To facilitate translations between mapping languages (R2RML, RML, OBDA) and tuple-generating dependencies (TGDs), we introduce a special language of source-to-source TGDs with built-in functions. We integrate state-of-the-art ontology materialisation systems from SW as well as datasets and scenarios from that area. Our initial experimental results already shed light on the interaction of these areas and the properties of different algorithms and systems, exhibiting the merits of our framework.

**Keywords**

Query rewriting, Ontology Materialisation, Data integration, OBDA, Dependencies, Benchmark

## 1. Introduction

Recent years have seen an increase in the demand of systems for data integration and exchange (DI/DE), a problem extensively studied in the both the Databases (DB) and the Semantic Web (SW) areas and invaluable for a variety of domains, including biological research, industry, data marketplaces, and others [1]. Several systems have been introduced to enable data integration. Efforts such as ChaseBench [2] have compiled a variety of systems that implement the chase algorithm, one of the most well-known class of algorithms used to infer new data.

In our previous work [3], we introduced ForBackBench, a framework that creates a unified approach to compare several DI/DE systems. ForBackBench comes with a set of common forward- and backward-chaining systems, common testing scenarios from literature, and pre-integrated converter tools to simplify the comparing and evaluation process, allow automated testing, and allow for easier extension in the future. The query-answering systems that initially became available on ForBackBench were classified in three groups: query rewriting (Rapid, Iqaros, Graal, Ontop, OntopR, GQR), chase (RDFox, Rulewerk, Llunatic), and hybrid systems (ChaseGQR). Through a command-line interface and a web interface, the user of the framework can run end-to-end experiments on combinations of query-answering systems and scenarios,

*ISWC 2023 Posters and Demos: 22nd International Semantic Web Conference, November 6–10, 2023, Athens, Greece*

a.alhazmi@soton.ac.uk (A. Alhazmi); j.o.salas@soton.ac.uk (J. O. Salas); g.konstantinidis@soton.ac.uk (G. Konstantinidis)



build new scenarios by providing ontology or TGDs files, and generate data with different sizes. Additionally, the command-line interface allows the user to run any converter as a stand-alone tool. The full details of ForBackBench functionalities are available in [4].

Although the backward-chaining systems included as part of ForBackBench came mostly from the Ontology-Based Data Access (OBDA) area of SW, the forward-chaining systems came from the state-of-the-art data exchange technologies in the DB domain. In this paper, we close the loop by integrating in ForBackBench state-of-the-art ontology materialisation engines from the SW domain. This brings us a step closer to realising our vision of bridging the gap between the DB and SW areas in DI/DE. Our contributions include the following:

- We present end-to-end translation algorithms and implementation between the most common SW mapping languages (OBDA mappings, R2RML, and RML) and TGDs.
- We integrate three state-of-the-art ontology materialisation systems into ForBackBench: Morph-KGC [5] , RMLMapper [6], and SDM-RDFizer [7].
- We integrate a new scenario, GTFS-Madrid-Bench scenario [8], a recent OBDA scenario with real data that was used in the KGCW 2023 Challenge at the KGC Workshop [9].
- We present an initial round of experiments pointing out the different ways algorithms and engines are used and shedding light on the performance of these systems.

To the best of our knowledge, this is the first framework that includes all relevant scenarios, systems and algorithms, paving the way for a versatile platform where one can plug different systems to produce a customised end-to-end solution for a DI/DE task.

## 2. Extended ForBackBench Framework

As part of our extension of the ForBackBench Framework, we aimed to include forward-chaining systems from the SW community, and to provide complete translation between forward- and backward-chaining formats. Therefore, we extended the functionality of the Data/Mapping Generator component that we introduced in [4] to support conversion between TGDs and mapping languages (R2RML [10] and RML [5]).

The problem of SW mapping translation was first introduced in [11] to discuss its importance and the available mapping translation engines. Our work is complementary to the recent research [12], which focuses on mapping translations between mapping languages within the SW area, where these languages rely on triples' generation. In our work, we focus on the mapping translation between SW mappings (that use triples) and DB mappings (that use rules). In order to test these translations, we integrated the GTFS-Madrid-Bench scenario, which comes with R2RML and RML mappings used to generate TGDs for DB forward-chaining systems.

**Mappings to TGDs Translation** In some cases, mappings languages use "Skolem" or "built-in" functions that cannot be encoded into TGDs in a straightforward manner. For instance, Example 1 shows an OBDA mapping object that creates new values for the arguments of $ns{:}EmpInfo$ by combining multiple columns of the source tables. Therefore, we introduce a new sub-language of TGDs: *Skolem source-to-source TGDs.* A Skolem ss-TGD($\Sigma_{ss}$) is a TGD of the form $\forall \vec{x}, \vec{y}(\varphi(\vec{x}, \vec{y}) \rightarrow \psi(\mu(\vec{x}), \mu(\vec{y})))$ where $\mu$ can be an arbitrary user-defined function (here

we limit ourselves to string concatenation). In the following we focus on relations of arity two (so they are directly translatable to triples, without reification).

**Definition 1.** *For all TGDs $\sigma$ of at most arity two, $\sigma$ is a Skolem source-to-source TGD if (1) $\varphi$ and $\psi$ are over the source schema, (2) $\varphi$, the body, is an SQL query , (3) $\psi$, the head, is a ChaseBench query, where $\mu$, the arguments, are concatenation functions of multiple columns of source tables.*

For ontology materialisation scenarios we implement two phases: the offline phase, where we convert OBDA, RML, or R2RML mappings to TGDs (see Example 1) and the online phase, where we generate and load data for the new source tables. The choice of supported mapping languages was guided by the current scope of our framework, which is mostly limited to relational DBs. In the future we aim to support more non-relational mapping languages.

**TGDs to Mappings Translation.** For each source-to-target (st) TGD mapping (we assume/-transform TGDs to have a single head atom), we define a *triple map* $(S, P, O, \phi)$ where the TGD head atom translates (via the natural way) into $S$, $P$, $O$ and $\phi$ is a generated SQL query reflecting the TGD body. Every mapping language (OBDA mappings, R2RML, or RML) contains a "target" that can be constructed from $S$, $P$, $O$ and a logical source in SQL or CSV. Thus we appropriately translate our triple map to a chosen language. We can also support translating back to a mapping language from a scenario that includes both st-TGDs as well as our own Skolem ss-TGDs (thus reverting Example 1), in which case the SQL query $\phi$ is obtained from the Skolem ss-TGDs (since it appears explicitly). If we have Skolem ss-TGDs, we use the namespaces contained in them; otherwise, we invent an example namespace for the final mapping.

**Example 1.**
*OBDA Mapping:*
*mappingId* *Mapping00643*
*target* *ns:employee/{eID}/name/{eName} ns:EmpInfo ns:dept/{dName}*
*source* *SELECT DeptEmp.eID as eID, DeptEmp.eName as eName, DeptInfo.dName as dName FROM DeptEmp INNER JOIN DeptInfo ON DeptEmp.dID = DeptInfo.dID*
*Translated TGDs:*
$\Sigma_{ss}$ = { *SELECT DeptEmp.eID as eID, DeptEmp.eName as eName, DeptInfo.dName as dName FROM DeptEmp INNER JOIN DeptInfo ON DeptEmp.dID = DeptInfo.dID* $\rightarrow$ *src_nsEmpInfo_ID_*00643(*ns:employee/{eID}/name/{eName}, ns:dept/{dName}*)}
$\Sigma_{st}$ = {*src_nsEmpInfo_ID_*00643$(?X, ?Y) \rightarrow$ *EmpInfo*$(?X, ?Y)$.}

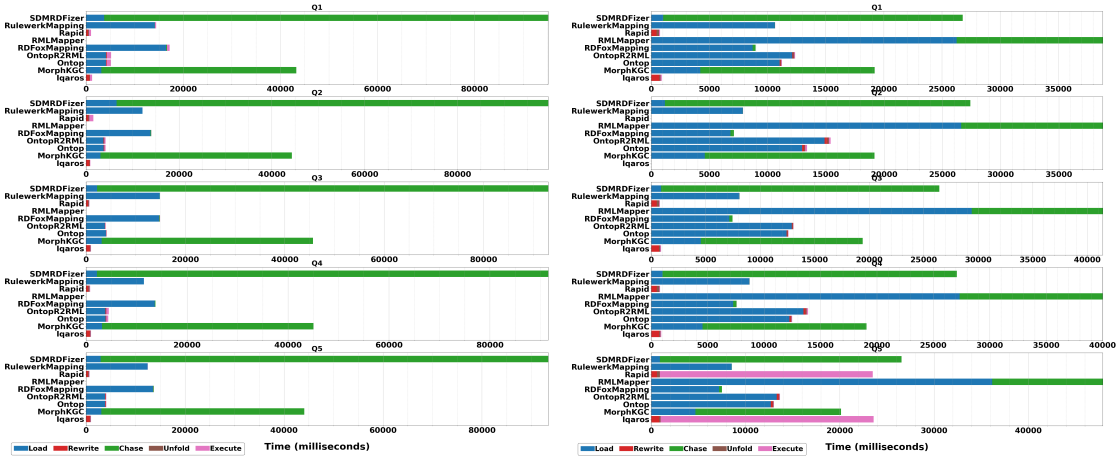
## 3. Experiments and Results

We are benchmarking materialization systems from DB (RDFox, Rulewerk), materialization systems from SW (Morph-KGC, RMLMapper, and SDM-RDFizer), and query rewriting systems (Rapid, Iqaros, and Ontop). Note that Ontop has two versions: (OntopR2RML) with R2RML mappings and (Ontop) with its native mapping language (i.e., OBDA mappings). Each system and query was run a total of six times, with the first being a 'cold-run'. We compare loading and chasing (generation of RDF graphs) times for forward-chaining, with loading, rewriting, and unfolding times (the unfolding time includes the conversion of the query to a valid SQL query) for backward-chaining. For DB systems we also include query execution times. All experiments were run on a MacBook Pro M1 chip with a 2.30GHz, 8-core processor, 8GB of memory, and a total of 1TB of disk space.

The SW forward-chaining systems (SDM-RDFizer, RMLMapper, and Morph-KGC) do not have out-of-the-box query answering functionality so we do not include execution times for these; we will extend the framework by including SPARQL query answering as future work. Thus, sometimes it is meaningless to compare an ontology materialisation system with a query rewriting system (that includes execution), especially when the former is faster. Nevertheless, in the GTFS-Madrid-Bench scenario seen below, the benchmarked SW materialisation systems are all slower than query rewriting systems that include execution, which is a very interesting result, pointing out that, possibly, virtual Knowledge Graphs – that do online query answering directly from sources – have not been investigated enough.

Figure 1a shows results for the GTFS-Madrid-Bench scenario [8] with its accompanying mappings, and a "small" data size (using the first scale of the VIG generator that comes with the scenario). Our choice for the 5 queries of GTFS-Madrid-Bench was based on their complexities, thus we chose both simple queries (without joins) and more complex queries (up to 8 joins). Note that systems' performance might vary up to a couple of seconds due to device resources allocation (including memory and CPU). Results show that DB forward-chaining systems, RDFox and Rulewerk, are significantly faster in chasing, in the current scenario, but slower in loading data. Overall, in this experiment, DB systems are faster than the ontology-materialisation engines SDM-RDFizer and Morph-KGC, while RMLMapper timed out (timeout was 2.5h).

Figure 1b shows the results of our experiments in the OWL2Bench scenario with again a small data size. Once again, the SW systems present faster loading times but significantly slower chase times except RMLMapper which was slow in loading as well. Query-rewriting systems (Ontop, Rapid, Iqaros) appear to be the fastest in both figures, however in the experiment of Figure 1b, both Rapid and Iqaros timed out after unfolding in Q2 due to a huge rewriting size, while in Q5 Morph-KGC is faster than both. The results, although very preliminary, imply that query-rewriting might be penalised for larger/complex queries, and that forward-chaining systems from both the SW and DB areas are valuable solutions on large datasets and complex queries. We are currently running more experiments.




(a) GTFS-Madrid-Bench Scenario

(b) OWL2Bench Scenario

**Figure 1:** Experimental results for different systems on "small" data size and GAV mappings.

## 4. Conclusion

We presented the most recent developments of the ForBackBench framework. We introduced Skolem source-to-source TGDs, as well as translations between mapping languages (R2RML, RML, OBDA) and TGDs. Finally, we integrated state-of-the-art SW ontology materialisation systems into ForBackBench and provided experimental results on a wide range of scenarios.

## 5. Acknowledgements

This work was partially funded by the UKRI Horizon Europe guarantee funding scheme for the Horizon Europe projects RAISE (101093216101058479) and UPCAST (101093216101093216).

## References

[1] A. Doan, A. Halevy, Z. Ives, Principles of data integration, Elsevier, 2012.
[2] M. Benedikt, G. Konstantinidis, G. Mecca, B. Motik, P. Papotti, D. Santoro, E. Tsamoura, Benchmarking the chase, in: Proceedings of the 36th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems, 2017, pp. 37–52.
[3] A. Alhazmi, T. Blount, G. Konstantinidis, Forbackbench: A benchmark for chasing vs. query-rewriting, Proceedings of the VLDB Endowment 15 (2022) 1519–1532.
[4] A. Alhazmi, G. Konstantinidis, OBDA vs forward chaining: the ForBackBench framework, International Semantic Web Conference (ISWC) 2022: Posters, Demos, and Industry Track (2022).
[5] A. Dimou, M. Vander Sande, P. Colpaert, R. Verborgh, E. Mannens, R. Van de Walle, RML: A generic language for integrated RDF mappings of heterogeneous data., Ldow 1184 (2014).
[6] A. Dimou, T. De Nies, R. Verborgh, E. Mannens, P. Mechant, R. Van de Walle, Automated metadata generation for linked data generation and publishing workflows, in: LDOW2016, CEUR-WS. org, 2016, pp. 1–10.
[7] E. Iglesias, S. Jozashoori, D. Chaves-Fraga, D. Collarana, M.-E. Vidal, SDM-RDFizer: An RML interpreter for the efficient creation of RDF knowledge graphs, in: Proceedings of the 29th ACM Intl conference on Information & Knowledge Management, 2020, pp. 3039–3046.
[8] D. Chaves-Fraga, F. Priyatna, A. Cimmino, J. Toledo, E. Ruckhaus, O. Corcho, GTFS-Madrid-Bench: A benchmark for virtual knowledge graph access in the transport domain, Journal of Web Semantics 65 (2020) 100596.
[9] D. Van Assche, D. Chaves-Fraga, A. Dimou, U. Şimşek, A. Iglesias, KGCW 2023 challenge @ ESWC 2023, 2023. URL: https://zenodo.org/record/7837289.
[10] S. Das, S. Sundara, R. Cyganiak, R2RML: RDB to RDF mapping language, 2012. URL: http://www.w3.org/TR/r2rml/.
[11] O. Corcho, F. Priyatna, D. Chaves-Fraga, Towards a new generation of ontology based data access, Semantic Web 11 (2020) 153–160.
[12] A. Iglesias-Molina, A. Cimmino, E. Ruckhaus, D. Chaves-Fraga, R. García-Castro, O. Corcho, An ontological approach for representing declarative mapping languages, Semantic Web (2022) 1–31.