International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 07 Issue: 08 | August - 2023 SJIF Rating: 8.176 ISSN: 2582-3930

DATA FEDERATION

Author: Ms. Vanshika Shahaji Nikam

Abstract

Data federation addresses the problem of uniformly accessing multiple, possibly heterogeneous data sources, by mapping them into a unified schema, such as an RDF(S)/OWL ontology or a relational schema, and by supporting the execution of queries, like SPARQL or SQL queries, over that unified schema. Data explosion in volume and variety has made data federation increasingly popular in many
application domains. Hence, many data federation systems have been developed in industry and academia, and it has become challenging for users to select suitable systems to achieve their objectives. In order to systematically analyze and compare these systems, we propose an evaluation framework comprising four dimensions:
(i) *federation capabilities*, i.e., query language, data source, and federation techniques; (ii) *data security*, i.e., authentication, authorization, auditing,
encryption, and data masking; (iii) *interface*, i.e., graphical interface, command line interface, and application programming interface; and (iv) *development*, i.e., main development language, deployment, commercial support, open source, andrelease. Using this framework, we thoroughly studied 51 data federation systems
from the Semantic Web and Database communities. This paper shares the results of our investigation and aims to provide reference material and insights for users, developers and researchers selecting or further developing data federation systems.

Introduction

The convenience of digitization, the variety of data descriptions, and the discrepancy in personal preferences have led large enterprises to store massive amounts of data in a variety of formats, ranging from structured relational databases to unstructured flat files. According to the prediction by Riesel et al., the global data volume will reach 163 zettabytes by 2025 and half of that data will be produced by enterprises.

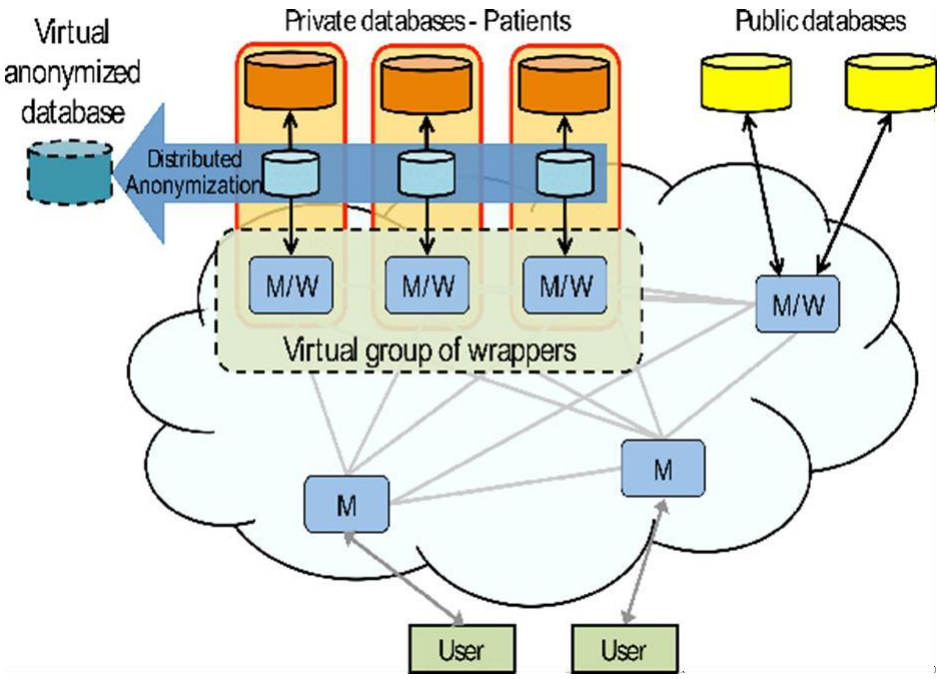
Since data becomes more valuable if enriched and fused with other data, decision- makers need to consider data distributed in different places and with different formats in order to get valuable insights that support them in their daily activities.

However, data explosion in volume, variety, and velocity – *i.e.*, the "3Vs" of Big Data – increases complexity and makes the traditional ways of data integration, such as data warehousing, not only more costly in terms of time and money but also unable to guarantee the freshness of data. Integration solutions developed in a more agile way are thus demanded especially in the Big Data context. *Data federation* is a technology that makes this possible today, that is becoming more and more appealing in both industry and academia, and that has been studied for a long time in different communities such as the Database and (more recently) the Semantic Web ones.

WHAT IS DATA FEDERATION?

D*ata federation systems* (also known as *federated database systems*) are traditionally defined as a type of meta-database management system that transparently maps multiple *autonomous database systems* into a single federated database. The key task of data federation systems is *federated query answering* that is to provide users with the ability of querying multiple data sources under a uniform interface. Such an interface usually consists of a *query language* over a *unified schema*, such as SQL over a relational schema or SPARQL over RDF(S) /OWL ontology, this interface being often closely related or restricting the query languages and schemas of supported data sources. Unlike in traditional pipelines for data extraction, transformation, and loading (ETL) often used in data warehouse systems, federated query answering is achieved by *data virtualization*, *i.e.*, all the data are kept *in situ* and accessed via a common semantic layer on the fly, with no data copy, movement, or transformation. As a result, federated query answering via data virtualization reduces the risk of data errors caused by data migration and translation, decreases the costs (*e.g.*, time) of data preparation, and guarantees the freshness of data. Compared to centralized solutions, though, accessing multiple data sources on the fly renders query answering more challenging and requires sophisticated optimization strategies to be devised. Besides federated query answering, modern data federation systems also offer a wide range of other important capabilities for data management, such as *read-and-write data access* for enabling users to both access and modify the data in the sources, *data security* for protecting the sensitive data of users and implementing secure data access, and *data governance* for managing the availability, usability, and integrity of the data.

ARCHITECTURE OF DATA FEDERATION.

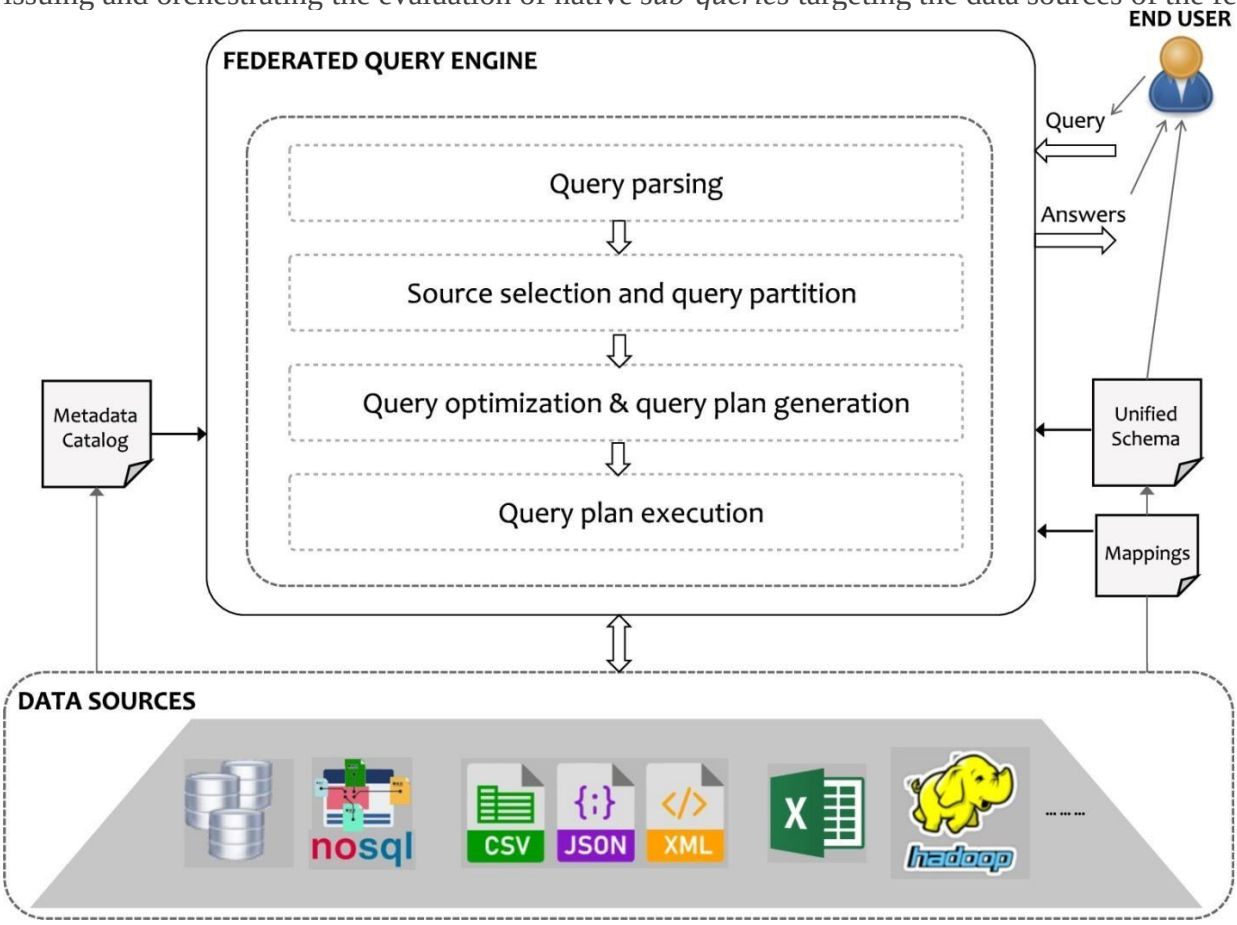


Virtual anonymized database: Data anonymization is the process of protecting private or sensitive information by erasing or encrypting identifiers that connect anindividual to stored data.

Outline of data federation

This section provides an overview of the main concepts underlying data federation that are addressed in this paper, for readers not already familiar with them.

The core task of data federation is federated query answering. For a set of autonomous and possibly heterogeneous data sources, the goal of federated query answering is to provide a uniform interface, typically as a unified query language over a unified schema, to access the data of these sources *in situ*, *i.e.*, without first copying the data to centralized storage. Given a user query over the unified schema, this task is carried out by issuing and orchestrating the evaluation of native *sub-queries* targeting the data sources of the federation.

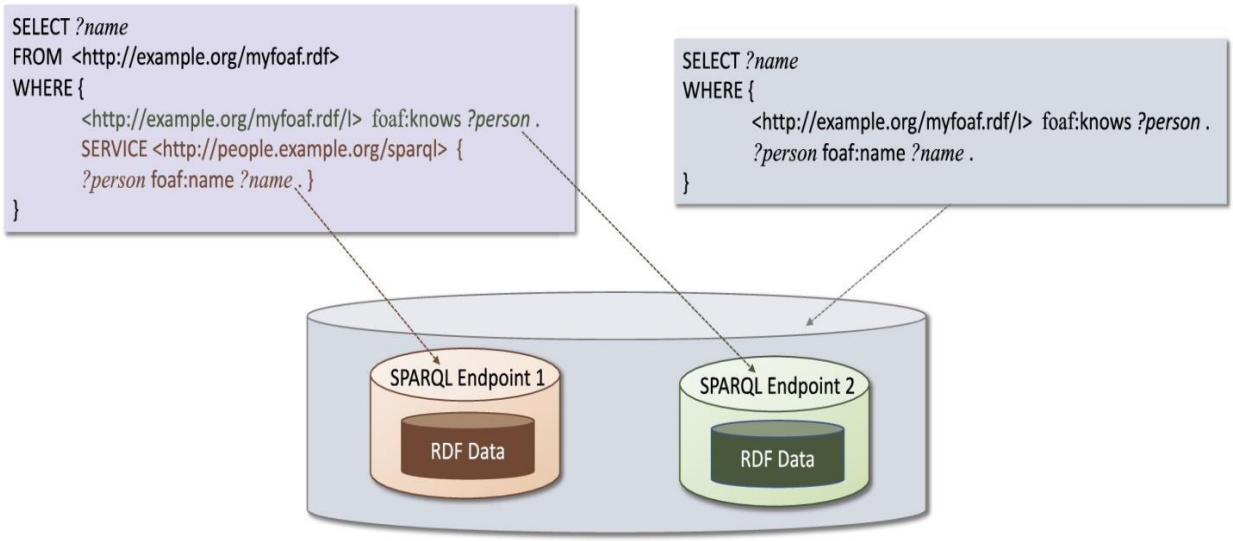


Query parsing. This step deals with the syntactic issues of *Q*, *i.e.*, checking whether the input queries are syntactically correct w.r.t. the adopted query language(s) as well as the unified schema. Some engines also transform *Q* into an algebraic form, such as a tree structure using internal nodes to denote operations (*e.g.*, join, union, or projection) and leaf nodes todenote accessed relations.

- *Source selection and query partition*. This step selects suitable data sources for each algebraic component of *Q*, and partitions *Q* into smaller sub- queries q1, qm1, (*i.e.*, query chunks) accordingly, based on the mappings from the data sources to the unified schema V. Approaches for source selection can be index-based, such as the "triple pattern-wise source selection" for SPARQL queries, and a way for query partitioning is to try to "push down" the evaluation of the operators to the data sources, rather than perform such evaluation at the level of the federation engine.

- *Query optimization & query plan generation*. This step computes an execution plan of the partitioned sub-queries q1,…,qm1,…, ??, establishing in which order to evaluate the sub-queries and which algorithms to use for joining their answers (*e.g.*, bind join, hash join, etc.), based on the metadata catalog. Existing approaches may be rule-based (*i.e.*, via predefined and deterministic heuristic rules) or cost-based (*i.e.*, choose the lowest-cost execution plan according to some heuristic cost function).
- *Query plan execution*. This step, finally, evaluates the decomposed sub- queries q1,…, qm1,…, over the corresponding data sources via the mappings and the metadata catalog, and generates the answers of the original query *Q*. Note that, if the query language that the data source supports is different from the query language of the federation engine, a translation based on the mappings is needed to translate the sub-query intothe one supported by the data source.

Transparent vs explicit federated query answering



From the perspective of whether the data source information is transparent for end users, federated query answering can be classified into *transparent federation* (the one we have discussed so far) and *explicit federation*. Transparent federation gives users the impression to query one single data source despite data being distributed and possibly coming from heterogeneous sources. Hence, it is recognized as a general and ideal22 solution.

A simplified setting is one where the unified schema is simply a merge of the source schemas, and the user explicitly states in the query the sources against which it should be evaluated. In such a scenario, we talk about explicit datafederation. This approach is built-in into SPARQL through its
dedicated SERVICE keyword, and therefore is supported by any SPARQL-based system fully compliant with SPARQL, including systems not primarily focusing on data federation. Figure, left-hand side, shows an example of query formulated under the explicit federation setting, asking for the data from a local RDF store and an explicitly specified remote RDF store. The right-hand side of the same figure shows the same query formulated under the transparent federation setting, assuming that foaf:knows and foaf:name are properties

belonging to the unified schema.

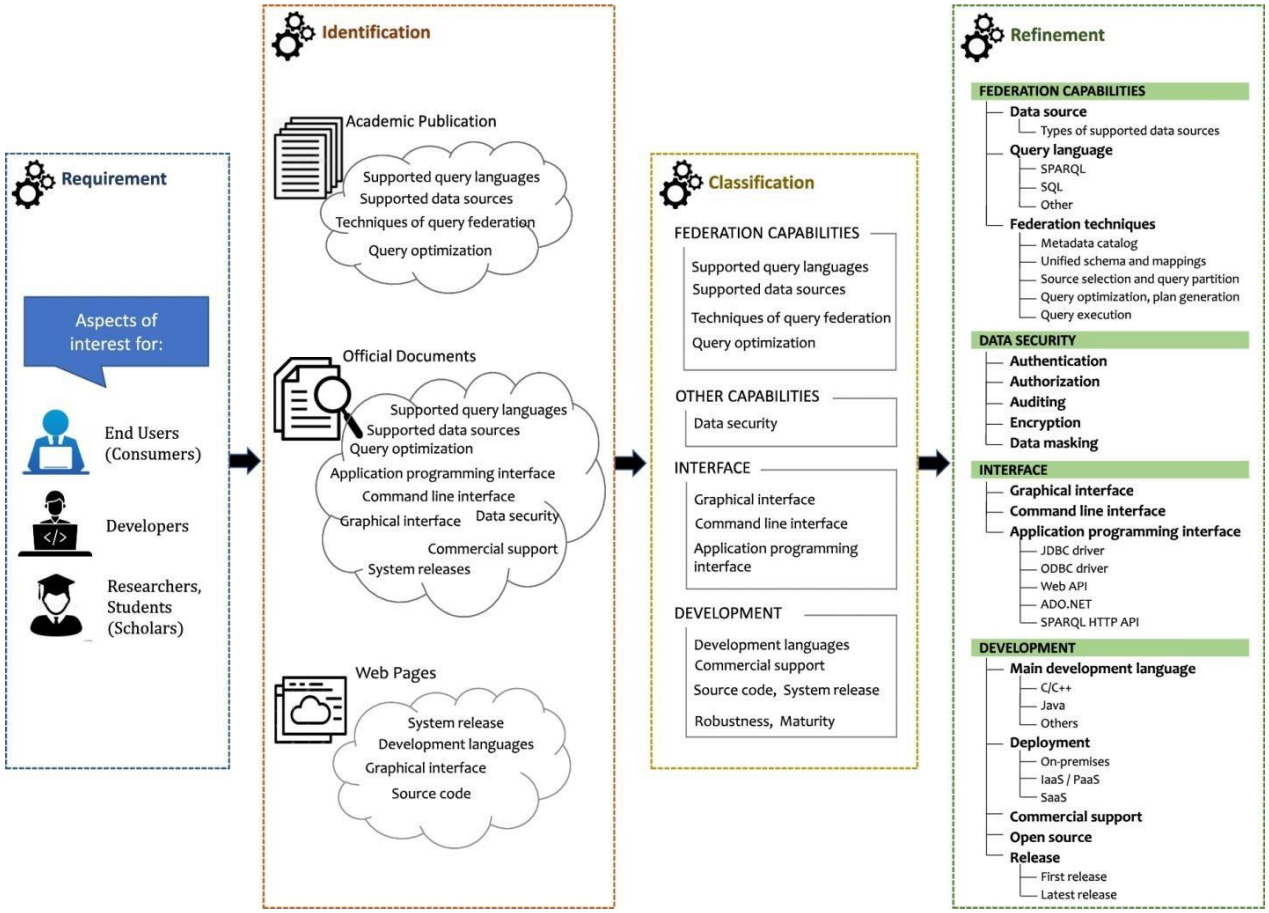
Compared with transparent federated query answering, the explicit scenario does not require a procedure of source selection for delivering its task of accessing and joining multiple data sources. However, the burden is placed on end users, and this might constitute a major hindrance in case they are not familiar with the data sources participating in the federation and the data therein contained.

However, the transparent setting is not devoid of drawbacks. For instance, users lose the ability of communicating with specific data sources directly. Moreover, the transparent situation needs to maintain a unified schema mapped to multiple data sources, which means that it is more sensitive to schema updates: when the schema of a source is updated, the unified schema and the mappings may also needto be updated.

The methodology for designing the evaluation framework

To design a framework for evaluating data federation systems in a uniform and qualitative way, also considering the intended audience of this survey, we focus onanswering the following question:

- *What aspects of data federation systems are relevant for end users, developers andscholars?*
While in principle answers can be obtained by interviewing these three groups (*e.g.*, via questionnaires), this approach presents two main difficulties: (i) it is hard to identify a representative sample to interview; and (ii) it is hard for interviewees to answer the question in a general and comprehensive way. Instead, we rely on the fact that data federation is an established domain that has been studied for decades in both the Database and the Semantic Web communities, leading to a large body of information from which to extract the aspects of interest that answer our question. Concretely, we consider three information sources:
- – *Academic publications*. We look for aspects deemed important by other surveys on data federation, or that are frequent in academic publications referring to data federation systems.
- – *Official documents*. We look for aspects commonly present in official documents of data federation systems, such as user and developer guides.
- – *Web pages*. We look for aspects that are often considered when comparing datafederation systems.



Data Federation vs. DataVirtualization

Data virtualization is a broad set of capabilities that *includes* data federation. Therefore, all federalized data is also virtualized data. But since data virtualization includes capabilities other than data federalization, not all virtualized data isfederated.

For example, data virtualization may abstract the technical details of a data source so you can query the data without requiring advanced technical knowledge.
Virtualizing data in this way is not the same as federating it.

The other differences between data virtualization and data federation include:

- Data federation implies multiple data stores; data virtualization doesn't.
- Federated data is always virtualized; virtualized data is not always federated.
- Data federation is a subset of data virtualization; data virtualization's featuresinclude federation.
- Data virtualization includes abstracting the peculiarities of specific data sources; data federation is virtualizing autonomous data stores into one largedata store.
- Data federation tools are often limited to integrating relational data stores. Data virtualization can connect data between any flavor of RDBMS, data appliance, NoSQL, Web Services, SaaS, and enterprise applications.

The Drawbacks and Benefits of Data Federation

Data federation is an excellent tool for short-term use cases that don't need to scale. It enables business users to forgo more resource-intensive, technical dataintegration for small-scope, one-off requests.

But data federation is not meant to build an enterprise-level data strategy.

You can liken data federation technology to an egg slicer. It's perfect for slicing eggs, but for anything else, you'll need a more versatile tool, like a set of kitchenknives.

Drawbacks

- Lack of historical data makes disaster recovery impossible
- Inability to recycle, reuse, and share code
- Virtually integrating becomes more difficult with more complex data
- No ability to track or take action on data changes

Benefits

- Ease of use for non-technical business users
- Quicker access to data
- No need for additional storage hardware since data isn't replicated

Conclusion

Data federation is one of the most critical data management strategies in today's data-driven world. With so much data being generated and collected every minute, it's necessary to have a way to manage all of this information right when you needit.
Businesses across many industries are using data federation for better search results and analytics as well as improved customer interactions. Customers expect companies to be able to provide them with relevant information that pertains specifically to their interests or preferences; businesses can do just that by connecting different sources into one system where they can then use data integration techniques such as de-identification, masking, and anonymization effectively and efficiently. In a federated data model, all of this can be done without creating and storing redundant copies of data.

Data federation helps solve many of the problems that businesses and organizations face when it comes to raw data, whether it has to do with largeamounts of data that need storage or a lack of consistency among the data.

References

1) https://federation.data.gov/
2) https://resources.data.gov/data-federation/
3) https://www.tibco.com/reference-center/what-is-a-data-federation
4) https://www.g2.com/articles/data-federation
5) https://docs.devexpress.com/XtraReports/400917/detailed-guide-to-devexpress-reporting/bind-reports-to-data/data-federation