Proceedings of the International conference on Computing Technology and Information Management, Dubai, UAE, 2014

MUSYOP: Towards a Query Optimization for Heterogeneous Distributed Database System in Energy Data Management

Zhan Liu[1], Fabian Cretton[1], Anne Le Calvé[1], Nicole Glassey[1], Alexandre Cotting[1] and Fabrice Chapuis[2]
[1]Institute of Business Information Systems
University of Applied Sciences and Arts Western Switzerland, Sierre, Switzerland
{zhan.liu, fabian.cretton, anne.lecalve, nicole.glassey, alexandre.cotting}@hevs.ch
[2]Institute of Business Information Systems
University of Applied Sciences Western Switzerland, Neuchâtel, Switzerland
fabrice.chapuis@he-arc.ch

ABSTRACT

The integration of data from multiple distributed and heterogeneous sources has long been an important issue in information system research. In this study, we considered the query access and its optimization in such an integration scenario in the context of energy management by using SPARQL. Specifically, we provided a federated approach - a mediator server - that allows users to query access to multiple heterogeneous data sources, including four typical types of databases in energy data resources: relational database Triplestore, NoSQL database, and XML. A MUSYOP architecture based on this approach is then presented and our solution can realize the process data acquisition and integration without the need to rewrite or transform the local data into a unified data.

KEYWORDS

Heterogeneous distributed database system, query optimization, mediator, federation, semantic web, database integration

1. INTRODUCTION

Information systems usually consist of multiple database systems, which may be stored on different computer systems, use different data models, etc. it is also very common to find that many of these databases contain overlapping and inconsistent data. In fact, the real world of databases is far from the ideal world of an integrated database where all of the data relevant to an organization would be stored and managed in one single unified and integrated database. Rather, databases are non-integrated, distributed and heterogeneous [1]. This is especially evident in the context of an energy database management system that not only requires storage of massive amounts of information every day, but also needs to be integrated with existing data applications like temperature management systems and a geographic information system.

Today's complex and increasingly globalized world which has encouraged waves of mergers and acquisitions, presents new difficulties for companies as they have to continue to handle huge amounts of complex and disparate information across regions. Simply exchanging basic information today may involve accessing and interpreting a wide variety of formats, data language, data models, and protocols that go beyond just text. Consequently, information integration is becoming increasing important and it consumes "a great deal of time and money" for large enterprises [2].

As a result, integrating and querying data from heterogeneous sources has become a hot research topic among information researchers. In general, there are two possible approaches to the architecture of a heterogeneous distributed database: namely warehouse approach (e.g., [3]) and federated approach (e.g., [4]). The separation is sometimes called centralized and decentralized systems. The first method typically provides a uniform interface to materialize the integrated view. The latter approach, on the other hand, is a form of virtual integration – the data are brought together as needed [5]. In this study we focus on the federated approach, as under this architecture local databases can continue their local operations and transactions without changing the features of local databases; but at the same time participate in the federation. Therefore, this approach is more stable and reliable in our case.

Companies must systematically manage energy use and handle as much energy information as possible to get deep and quantitative knowledge of the process of energy consumption [6]. As an important part of information resource, energy information resource supports energy efficiency and influences the direction of future performance.

However, similar to other manufacturers, energy companies also involve heterogeneous database system problems due to several reasons. First and foremost, an energy database management system was built according to the characteristics of the energy usage in a specific region. Thus, the requirements are diverse and as a consequence, database systems in the field of energy are rather distributed and complicated. For example, electricity consumption function requires its information system to quickly convert and store a large body of non-relational data; thus, a NoSQL database like MongoDB [7] fits very well in this case. However, in other functions where the data are stable with low variability, and if such data are related to data sources, a relational database should be a good choice. Moreover, Triplestore is selected if the consumption data are necessary to integrate with other remote data resources, like geographic and weather information systems. In the case of a semi-structured data model, XML serves well and it is usually used to store and exchange information of configuration for different systems. Second, an integrated system was not the main goal at the time the database systems were built [1]. Third, energy database systems that differ from each other may be caused by changes in technology. Last but not the least, in contemporary urban environments and at a household level, energy management requires that the design of systems be able to integrate remote and spatially distributed monitoring data while being open, low cost, easy to use and flexible [8]. All these characteristics indeed set barriers to getting accurate energy information in a global perspective. Only using the existing tools cannot solve these problems.

The burgeoning semantic web technology has provided new methods for integration of heterogeneous distributed database management systems. According to Tim Berners-Lee et al. [9], the Semantic Web "provides a common framework that allows data to be shared and reused across application, enterprise, and community boundaries." While rapidly evolving, it is only recently that semantic web technologies are becoming available and stable, and practical solutions emerge and flourish in many fields. The idea involves the concept of Linked Data, which aims at enabling the same kind of possibilities for data, as well as creating a universal medium for exchanging information based on the meaning of content on the Web [10] in a way that is usable directly by machines. Resource Description Framework (RDF) is a general language to describe resources, especially on the web, and SPARQL is a query language for RDF that can join data from different databases, as well as documents, inference engines, or anything else that might express its knowledge as a directed labeled graph [11].

To this end, we proposed a uniform approach for SPARQL querying a heterogeneous distributed database system named MUSYOP. This federated method provides transparent query access to multiple heterogeneous data sources, including relational database, Triplestore, NoSQL database and XML, thus realizing the process data acquisition and integration without the necessity to rewrite or transform the local data. Most extant studies in heterogeneous distributed database systems only consider a single language (e.g., [1]) or only focus on relational data (e.g., [11]). Our approach is different from them in two ways: on one hand, we do not only look at one specific database model (e.g., relational database), but also provide solutions to integrate other database models. On the other hand, our mediator server does not require local databases to translate or transfer to a unified language; rather, all local sources remain at the original level thus it has cost advantages. Moreover, our solution fulfills the energy information calculation from the integrated data: e.g., daily, monthly, quarterly, etc. In addition, MUSYOP uses query optimization to speed-up search executions.

The remainder of this paper is structured as follows: we start with a discussion of related work in Section 2. Section 3 describes the architecture of MUSYOP for heterogeneous distributed database system. We conclude with a description of ongoing and future work in Section 4.

2. RELATED WORKS

Data discovery, date mining, and date integration have been important research topics in the field of heterogeneous distributed database management systems for years. Two possible approaches are briefly described as follows:

The warehouse and RDFizer [12] approaches usually consolidate data from multiple sources. The advantages of this method are the high efficiency and the capability of extracting deeper information for decision making [6]. However, the warehouse database must set up the “data cleansing” and “data standardized” areas before its actual use. Overlapping and inconsistent information may exists among local sources; thus, it must be cleansed. Moreover, each local database may adopt different models from the warehouse’s (e.g., schema, data type); thus, local sources need to be reshaped and transformed into a common one, that is, “data standardized”. As a consequence, it typically would take months of planning and effort to create [5].

In contrast, the federated approach provides a single interface to many underlying data sources without the user explicitly specifying the target data source in the query. The advantages of data federation are the high adaptability to frequent changes of data sources, and the support of large numbers of data sources and data sources with high heterogeneity [6].

A large of variety of federated queries has been proposed recently for heterogeneous distributed databases (e.g., [13]; [14]). SPARQL, as a query language for RDF, has been well accepted to support querying of multiple RDF databases. It aims to find matching resources from a graph-like connected web for the database community [15]. For example, both [16] and [17] described the approaches for SPARQL queries over a catalogue of remote endpoints from multiple distributed relational databases. Moreover de Laborda and Conrad [18] introduced a SPARQL query mechanism for mapping relational databases to an ontologies approach. Contrary to other approaches, they took the complete schema of the database into account, creating a database specific ontology. To the best of our knowledge, no existing research addresses SPARQL federated query to support a heterogeneous distributed database system including the most current and popular databases, such as relational, Triplestore, NOSQL, and XML. MUSYOP provides transparent query access over mapped RDF data sources. Our approach offers a standard SPARQL query interface to retrieve the desired distributed data in RDF format.

Most studies on SPARQL query optimization for a heterogeneous distributed database system include two aspects: minimizing communication cost and optimizing execution localization. According to [19], communication cost is reflected in the number of contacted data sources. It directly influences the performance of the query execution due to the communication overhead. The approach of query rewriting identifies the complex elements and proposes specific rewriting rules; therefore, it could be used to resolve the cost of communication among different databases ([20]; [21]; [22]). From [19] point of view, optimizing execution localization is represented by identifying optimal index structure and join ordering in order to execute queries in parallel and reduce query execution time. However, there is still little discussion of SPARQL query optimization across multiple heterogeneous databases. We aim to fill this research gap.

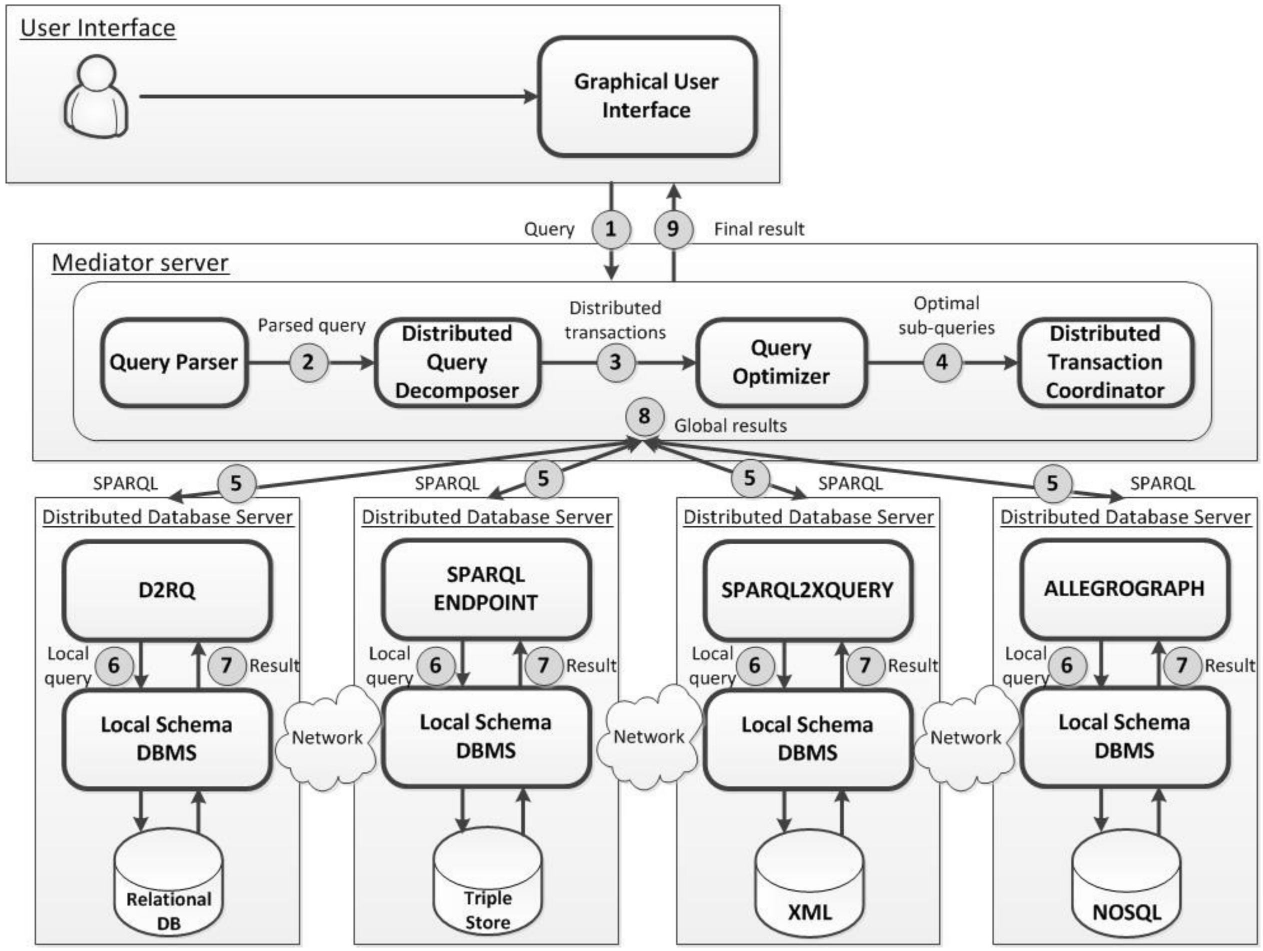

Figure 1. MUSYOP architecture of heterogeneous distributed database system

3. ARCHITECTURE OF HETEROGENEOUS DISTRIBUTED DATABASE SYSTEM

In this section, we first present a global view of our architecture; then, we will introduce in detail each component in the heterogeneous distributed database system for energy management.

The architecture MUSYOP, as shown in Figure 1, contains three principal layers. The first layer includes users' interfaces and in it, user could send one or multiple queries via a Graphical User Interface (1) to a Mediator Server layer. The mediator server is on the second layer of our architecture. It is a middleware system containing a global schema that describes the data throughout the network, and it is used to support and coordinate the distributed transaction management. The mediator is designed to integrate any kind of component database. Four important components are stored in this layer: Query Parser, Distributed Query Decomposer, Query Optimizer, and Transaction Coordinator. Once a user's query is received by the mediator, the query will be scanned and parsed into a graph structure of SPARQL. If no error is found, the generated transactions corresponding to the query are sent to the Distributed Query Decomposer (2), which can interpret the query received from the user's interface and generates a distributed query context containing several transactions and their associations (i.e., joins) [23]. Then, Query Optimizer takes all distributed transactions (3), and generates optimal sub-queries to build an optimal SPARQL query execution plan. The optimization of such sub-queries is a key factor concerning the performance of the overall system.

For each distributed transaction (4), the Distributed Transaction Coordinator looks up the corresponding distributed database schema at which the accessed relation of the transaction resides from the definition of endpoint addresses. However, SPARQL queries require an explicit definition of endpoint URIs. Our system allows execution of queries without the necessity to specify target remote endpoints. After that, the Transaction Coordinator generates the navigation information in the form of SPARQL for all sub-transactions according to the associations among them, and sends then separately to the related heterogonous database server (5). The third layer of our architecture contains four different heterogonous distributed database servers: relational database, Semantic Web TripleStore, XML database, and NoSQL database. In order to encapsulate the details of component databases, free RDF-wrappers such as D2RQ [24], SPARQL2XQUERY [25] and AllegroGraph [26] are associated and placed on the top of distributed database systems. Therefore, when a SPARQL query arrives at the heterogonous distributed database servers, the query does not directly refer to distributed database. Instead, it contains graph patterns adhering to a virtual RDF data set. In addition, RDF-wrappers also participate in query optimization. Then, the corresponding RDF-wrapper generates the SPARQL query into a local query to the local schema DBMS (6). The execution result of the local transaction is returned back to the same wrapper (7) and then, the local result is converted to a uniform format (e.g., XML or JSON) and is collected by the Mediator Server (8). Finally, the client receives the global results in the form of HTML on their interfaces (9).

Now, we will present each component in our architecture in details as follows:

3.1 Graphic user interface

A type of full screen user interface allows users to issue queries and to receive the returned results.

3.2 Query parser

Query parser is used to scan and parse query statements to check syntactic errors, such as query references, names of relations, and attributes.

3.3 Distributed query decomposer

Distributed query decomposer generates a number of transactions to match the underlying remote data sources. These distributed transactions are submitted and executed in parallel with heterogeneous databases over remote connections. Moreover, the distributed query decomposer assembles transaction results and returns a final result to the end user.

3.4 Query Optimizer

SPARQL query optimizer in mediator layer provides an approach of the query execution plan to minimize the communication and processing costs to transmit query and result between mediator and heterogeneous distributed databases. In fact, the join order has a significant influence on the cost-effective query execution plan. Therefore, the join order optimization is usually the main focus of SPARQL query optimization. In our architecture, we proposed two steps for query optimization, namely data source optimization and join order optimization.

The data source optimization is represented by the precision of the data source selection and building sub-queries. The idea is to determine all return results from different data sources. Specifically, the data source selection would identify whether the return result to the SPARQL query is empty and which data source does not need to be accessed. Therefore, we send SPARQL ASK queries [27] including the triple pattern to all the federation databases and eliminate sources that fail to match the pattern. This refining of data sources is more efficient than accepting no results in regular SPARQL SELECT queries. The results from source selection are then used to build sub-queries. Each sub-query contains triple elements: triple patterns, value constraints and data source that can answer the sub-query. One sub-query could be matched to one or multiple data sources. The join order optimization is implemented in our solution to determine the numbers of intermediate results, since all query execution plans for heterogeneous distributed databases are based on the sub-queries generated by the data source selection. It is possible to use sub-queries joining larger result sets in a nested loop join after the smaller result set has been received completely. This mode of join order is called “Mediator Join” [19]. It executes the joins in the mediator after comparing the intermediate result sets from the data sources, and only the smaller results set will be returned to the mediator. This approach of join order is used in our query optimizer to deal with large result sets and it will drastically reduce the transfer costs.

3.5 Distributed transaction coordinator

A distributed transaction coordinator is used in our architecture to manager optimal distributed sub-transactions. It detects and handles persistent records of the transactions, and manages the communications with the databases.

3.6 SPARQL endpoint

A SPARQL endpoint allows users to query a machine-friendly interface towards a knowledge base such as triple store via the SPARQL language. The results are returned in machine-processable formats, like XML and JSON. In our case, the four different RDF-Wrappers in the distributed database server could be considered as four SPARQL endpoints.

3.7 Mapping relational data to RDF

There are existing approaches for mapping relational data to RDF, such as Triply [28], R2O [29], and RDBToOnto [30]. In this study, we chose the approach of D2R to ease integration of our relational database and discover information without replicating the data into a dedicated RDF triple store. The D2R server uses D2RD mapping language to provide an automated process to generate the mapping file between specific relational database schemas and RDF schemas. This mapping file convers all tables from relational database to RDF classes, and it is used to identify resources, as well as access and generate property values into RDF format from database content.

The D2R server allows applications to query relational databases using SPARQL query language through the SPARQL protocol. Once the SPARQL requests arrive from the mediator, they are rewritten into SQL queries via the mapping and executed against a D2RQ-mapped relational database. Finally, the query results will be represented in XML and JSON formats and integrated into global results.

3.8 Mapping NOSQL data to RDF

There have been a considerable number of studies between NOSQL databases and relational databases in the past couple of years. However, these studies mainly focused on the conversion between these two formats, and which type of database is more effective and optimized for specific database management issues. Until now, little attention has been paid to the integration of NOSQL data and RDF. AllegroGraph is one of few tools that could help map NOSQL data to RDF. AllegroGraph server is developed to meet W3C standards for the RDF; therefore, it could be used as an RDF database. Similar to D2R for relational databases, the AllegroGraph server provides a mapping mechanism, and it allows query graph style linked data and document based data in MongoDB by using SPARQL. This mapping mechanism provides a “read-only” mode for the database content; thus, the requests of adding, updating, and deleting will not change any data in MongoDB. In order to map the data from two databases, the Mongo ID is used to create the connection variables for the subject of each triple. Then, we use the magic predicate from AllegroGraph to query MongoDB, and the results are returned in JSON formats.

3.9 Mapping XML data to RDF

XML has been widely successful for configuration information storage and information exchange on the Web. XML defines a set of rules to describe the content of structured and unstructured documents in a format that is both human-readable and machine-readable [31]. Several research works have clearly observed striking similarities between semi-structured data models and XML ([32]; [33]). These similarities are reflected in their irregular or often changing structure, as well as different attributes for different entities represented in a model that is often based on using tree or graph data structures.

A number of combinations of Semantic Web and XML technologies have been exploited. However, the objectives of these research works (e.g., [34]; [35]) only focus on data transformation from XML to RDF. SPARQL2XQuery represented a comprehensive framework that allows expressing semantic queries on top of XML data through the translation of SPARQL queries in XQuery syntax. SPARQL2XQuery proposed two types of scenarios to query CML data by using SPARQL. The first scenario is based on an automatically generated mapping ontology, and the second is based on an existing OWL ontology. In our framework, the first scenario is matched and used to generate the mappings between the ontology and the XML schema automatically, as well as to integrate and query the XML data from the Sematic Web environment. The query results are transformed into the desired formats (such as XML or RDF) and returned to the mediator layer.

4. CONCLUSION AND FUTURE RESEARCH

This paper describes the architecture of a heterogeneous distributed database system that we call MUSYOP. Contrary to other studies, we proposed a mediator server, a middleware that contains a global schema throughout the network and is used to support and coordinate the distributed transaction management. Based on this mediator, we showed how to query data among four widely used data sources, including relational database, Triplestore, NoSQL database and XML. With this approach, the system can integrate any kind of component database and it does not require any changes to local databases. We also proposed an approach for query optimization based on our architecture and, in the near future, we plan to experiment to enhance the flexibility and optimize the query to speedily retrieve data.

The next step for this study would be to implement our solution in a real example in order to evaluate its performance. In particular, we intend to implement the MUSYOP to evaluate our approach by accessing real databases with a large amount of energy data in Switzerland.

5. ACKNOWLEDGEMENTS

The work described in this paper was supported by the University of Applied Sciences and Arts Western Switzerland (HES-SO) under grant number 34930.

6. REFERENCES

[1] J. M. Smith, P. A. Bernstein, U. Dayal, N. Goodman, T. Landers, K. W. T. Lin, and E. Wong, “Mutibase - Integrating Heterogeneous Distributed Database Systems,” In Proceeding of AFIPS of the May 4-7, National Computer Conference, pp.487-499, May 4-7, 1981.

[2] P. A. Bernstein, and L. M. Haas, “Information Integration in the Enterprise,” Communication of the ACM (51:9), 2008, pp.72-79.

[3] S. Chaudhuri, and U. Dayal, “An Overview of Data Warehousing and OLAP Technology,” ACM SIGMOD Record (26:1), 1999, pp.65-74.

[4] A. P. Sheth, and J. A. Larson, “Federated Database Systems for managing Distributed Heterogeneous, and Autonomous Databases,” ACM computing Surveys (22:3), 1990, pp.183-236.

[5] L. M. Haas, and A. Soffer, “New Challenges in Information Integration,” In DaWak 2009: Data Warehousing and Knowledge Discovery, T. Pedersen, M. Mohania, A. Tjoa, editors, Linz, Austria. Heidelberg: Springer, pp. 1-8, August 31 – September 2, 2009.

[6] B. Wu, J. Li, H. Liu, Z. Zhang, Y. Zhou, and N. Zhao, “Energy Information Integration based on EMS in Paper Mill,” Applied Energy (93), 2012, pp.488-495.

[7] 10gen, Inc., “MongoDB documentation”, 2012.

[8] K. Karatzas, A. Papadopoulos, N. Moussiopoulos, E. A. Kalognomou, and A. Bassoukos, “Development of a Hierarchical System for the Tele-transmission of Environmental and Energy Data,” Telematics and Informatics (17), pp.239-249, 2000.

[9] T. Berners-Lee, H. James, and L. Ora, “The Semantic Web,” Scientific American Magazine, 2001.

[10] S. A. Theocharis, and G. A. Tsihrintzis, “Semantic Web Technologies in e-Government,” World Academy of Science, Engineering and Technology, vol. 64. 2012, pp. 1237-1244.

[11] J. Wang, Z. Miao, Y. Zhang, and B. Zhou, “Querying Heterogeneous Relational Database Using SPARQL,” Eighth IEEE/ACIS International Conference on Computer and Information Science, pp.475-480, 2009.

[12] S. Mazzocchi, S. Garland, and R. Lee, “Simile: Practical metadata for the semantic web,” XML.com, 2006.

[13] V. Josifovski, P. Schwarz, L. Haas, and E. Lin, “Garlic: A New Flavor of Federated Query Processing for DB2,” In Proceedings of the 2002 ACM SIGMOD International Conference on Management of Data, Madison, Wisconsin, pp. 524-532, 2002.

[14] S. Schenk, C. Saathoff, S. Staab, and A. Scherp, “SemaPlorer – Interactive Semantic Exploration of Data and Media based on a Federated Cloud Infrastructure,” Journal on Web Semantics: Science, Services and Agents on the World Wide Web 7(4), 2009, pp. 298-304.

[15] M. Arenas, and J. Pérez, “Querying Semantic Web data with SPARQL,” In PODS, ACM, pp. 305–316, 2011.

[16] A. Langegger, W. Wöß, and M. Blöchl, “A Semantic Web Middleware for Virtual Data Integration on the Web,” In ESWC’08 Proceedings of the 5th European semantic web conference on the semantic web: research and applications. Springer Berlin, Heidelberg, pp. 493-507, 2008.

[17] H. Chen, Y. Wang, H. Wang, Y. Mao, J. Tang, C. Zhou, A. Yin, and Z. Wu, “Towards a semantic web of relational databases: a practical semantic toolkit and an in-use case from traditional chinese medicine,” In 4th International Semantic Web Conference (ISWC). LNCS, Athens, USA, Springer-Verlag, pp. 750-763, 2006.

[18] C. P. de Laborda, and S. Conrad, “Bringing Relational Data into the SemanticWeb using SPARQL and Relational OWL,” Proceedings of the 22nd International Conference on Data Engineering Workshops (ICDEW'06), pp. 55, 2006.

[19] O. Gorlitz, and S. Staab, “Federated Data Management and Query Optimization for Linked Open Data,” In New Directions in Web Data Management. Springer, pp. 109-137, 2011.

[20] O. Hartig, and R. Heese, “The sparql query graph model for query optimization,” In 4th European Semantic Web Conference (ESWC), pp. 564-578, 2007.

[21] D. Kossmann, “The State of the Art in Distributed Query Processing,” ACM Computing Surveys 32(4), 2000, pp. 422-469.

[22] M. Schmidt, M. Meier, and G. Lausen, “Foundations of SPARQL query optimization,” In ICDT, pp. 4-33, 2008.

[23] D. Y. Yeh, M. C. Lee, and T. I. Wang, “Mobile Agents for Distributed Transactions of a Distributed Heterogeneous Database System,” Proceedings of the 13th International Conference on Database and Expert Systems Applications (DEXA 2002), Aix-en-Provence, France, pp. 403-412, 2002.

[24] C. Bizer, and A. Seaborne, “D2rq: Treating non-rdf databases as virtual rdf graphs,” In 3rd International Semantic Web Conference (ISWC2004 posters), 2004.

[25] I. Stavrakantonakis, C. Tsinaraki, N. Bikakis, N. Gioldasis, and S. Christodoulakis, “Sparql2xquery 2.0: Supporting semantic-based queries over xml data,” In Semantic Media Adaptation and Personalization (SMAP), IEEE, pp. 76-84, 2010

[26] J. Aasman, “Allegro Graph: RDF Triple Database,” Technical Report 1, Franz Incorporated., 2006.

[27] A. Schwarte, P. Haase, K. Hose, R. Schenkel, and M. Schmidt, “FedX: Optimization Techniques for Federated Query Processing on Linked Data,” In Proceedings of the 10th International Semantic Web Conference, Bonn, Germany, pp. 481-486, 2011.

[28] S. Auer, S. Dietzold, J. Lehmann, S. Hellmann, and D. Aumueller, “Triplify – Light-Weight Linked Data Publication from Relational Databases,” Proceedings of the 18th World Wide Web Conference, Madrid, Spain, pp. 621-630, April 20-24, 2009.

[29] J.B. Rodriguez, and A. Gomez-Perez, “Upgrading relational legacy data to the semantic web,” Proceedings of the 15th international conference on World Wide Web, Edinburgh, Scotland, pp. 1069-1070, May 23-26, 2006.

[30] F. Cerbah, “Learning highly structured semantic repositories from relational databases: the RDBToOnto tool,” Proceedings of the 5th European semantic web conference on The semantic web: research and applications, Tenerife, Canary Islands, Spain, pp. 777-781, June 01-05, 2008.

[31] T. Bray, J. Paoli, C. M. Sperberg-McQueen, E. Maler, and F. Yergeau, “Extensible Markup Language (XML) 1.0 (Fifth Edition),” World Wide Web Consortium, Recommendation REC-xml-20081126, 2008.

[32] R. Goldman, J. McHugh, and J. Widom, “From semistructured data to XML: Migrating the Lore data model and query language,” In ACM SIGMOD WebDB Workshop, pp. 25-30, 1999.

[33] D. Suciu, “Semistructured data and XML. In Information organization and databases,” Springer US, 2000, pp. 9-30.

[34] M. Droop, M, Flarer, J. Groppe, S. Groppe, V. Linnemann, J. Pinggera, F. Santner, M. Schier, F. Schopf, H. Staffler and S. Zugal, “Embedding XPATH Queries into SPARQL Queries,” In Proceedings of the 10th International Conference on Enterprise Information Systems (ICEIS), pp. 5-14, 2008.

[35] W. Akhtar, J. Kopecky, T. Krennwallner and A. Polleres, “XSPARQL: Traveling be-tween the XML and RDF Worlds and Avoiding the XSLT Pilgrimage,” In Proceedings. 5th European Semantic Web Conference, ESWC 2008, Tenerife, Canary Islands, Spain, pp. 432-447, June 1-5, 2008.