# Chapter 19
# RDF Storage and Querying:
## A Literature Review

**Jingwei Cheng**
*Northeastern University, China*

**Z. M. Ma**
*Northeastern University, China*

**Qiang Tong**
*Northeastern University, China*

## ABSTRACT

*RDF plays an important role in representing Web resources in a natural and flexible way. As the amount of RDF datasets increasingly growing, storing and querying theses data have attracted the attention of more and more researchers. In this chapter, we first make a review of approaches for query processing of RDF datasets. We categorize existing methods as two classes, those making use of RDBMS to implement the storage and retrieval, and those devising their own native storage schemas. They are called Relational RDF Stores and Native Stores respectively. Secondly, we survey some important extensions of SPARQL, standard query language for RDF, which extend the expressing power of SPARQL to allow more sophisticated language constructs that meet the needs from various application scenarios.*

## INTRODUCTION

The Semantic Web (Berners-Lee, Hendler, & Lassila, 2001) is an extension of current Web, in which Web resources are given computer-understandable semantics, better enabling computers and people to work in cooperation. Resource Description Framework (RDF) (Manola, Miller, & McBride, 2004) provides a natural and flexible way to describe resources in the Web and how they are related. RDF data is essentially a set of triples of the form (subject, predicate, object), each of which states that the subject is related to object through the predicate. As more and more information is characterized with RDF, storing huge amounts of RDF data and efficiently evaluating queries over these data plays a central role in achieving the Semantic Web vision. The vigorous development of RDF has attracted the attentions of

DOI: 10.4018/978-1-5225-5191-1.ch019

researchers from database and Web communities. Different solutions and practical systems for efficient and scalable management of RDF data are designed and implemented. SPARQL (Prud'Hommeaux, & Seaborne, 2008) is the standard query language of W3C, SPARQL 1.1 is its latest version (Harris, & Seaborne, 2010). SPARQL can be used to express queries across diverse data sources, whether the data is stored natively as RDF or viewed as RDF via middleware. SPARQL contains capabilities for querying required and optional graph patterns along with their conjunctions and disjunctions. SPARQL also supports aggregation, subqueries, negation, creating values by expressions, extensible value testing, and constraining queries by source RDF graph. The results of SPARQL queries can be result sets or RDF graphs.

We discuss in this chapter mainly query processing of RDF data. However, the efficient query processing heavily depends on the storage strategy of RDF data. The storage strategy, or how a RDF store internally represents RDF data, is a central topic which influences every aspect of the source, from indexing, to planning and evaluation. We use the term "RDF Store" to refer to any RDF management system to provide a mechanism for persistent storage and access of RDF data, usually provide an endpoint for accepting queries and showing query results. We thus categorize these implementations around the storage strategies adopted by various RDF stores. They come in many different varieties. For small volume of RDF graphs, it is even possible to efficiently handle and manage data in computers' main memory. Larger RDF graphs render the deployment of persistent storage systems indispensable. RDF stores that make use of purpose-built databases for the storage and retrieval of any kind of data expressed in RDF are called "Relational RDF Stores" or "Relational Stores" in this chapter. In former literatures, the term "RDF Store" (Haslhofer et al., 2011) or "Triple Store" (Rohloff et al., 2007; Rusher, 2003) are frequently used to refer to this kind of systems. There are still other systems implementing their own native storage and indexing formats. We call these systems that do not make use of relational databases as "Native Stores" (Bizer & Schultz, 2009).

Currently, there have been lots of works concerning query processing of RDF datasets, but few comprehensive review papers can be found. Hertel, Broekstra, & Stuckenschmidt (2009) reviewed RDF storage and retrieval technologies as a common approach for accessing ontology-based data. Bizer, & Schultz (2008) introduced the Berlin SPARQL Benchmark (BSBM) for comparing the performance of these systems across architectures. They also presented the results of an experiment comparing the performance of D2R Server, a relational database to RDF wrapper, with the performance of Sesame, Virtuoso, and Jena SDB. Haslhofer et al. (2011) summarize the results of qualitative and quantitative study they carried out on existing RDF stores in the context of the European Digital Library project. The focus of these three works is to make a comparison of the performance of existing RDF storage and retrieval systems. Sakr, & Al-Naymat (2010) focuses on using relational query processors to store and query RDF data. They provided an overview of the different approaches and classify them according to their storage and query evaluation strategies. Kaoudi, & Kementsietsidis (2014) provided an overview of query processing techniques for the RDF data model using different system architectures. They focused on distributed RDF stores, including peer-to-peer, federated and cloud-based systems. This chapter reports the RDF storage and query processing of both RDBMS-based and native storage-based, i.e., Relational RDF Stores and Native Stores. We adapt the structure of (Sakr, & Al-Naymat, 2010) and (Kaoudi, & Kementsietsidis, 2014), with up-to-date literatures reflecting significant progresses in this area. Note that, however, it does not mean that this paper covers all publications in the research area and give complete descriptions.