Data Partitioning for Semantic Web

Trupti Padiya
DA-IICT
Gandhinagar,India
trupti_padiya@yahoo.com

Mohit Ahir
DA-IICT
Gandhinagar,India
mohitaheer@gamil.com

Minal Bhise
DA-IICT
Gandhinagar,India
minal_bhise@daiict.ac.in

Sanjay Chaudhary
DA-IICT
Gandhinagar,India
sanjay_chaudhary@daiict.ac.in

***Abstract*-** Semantic web database is an RDF database. Tremendous increase can be seen in semantic web data, as real life applications of semantic web are using this data. Efficient management of this data at a larger scale, and efficient query performance are the two major concerns. This work aims at analyzing query performance issues in terms of execution time and scalability using data partitioning techniques. An experiment is devised to show effect of data partitioning technique on query performance. It demonstrates the query performance analysis for partitioning techniques applied. Vertical partitioning, hybrid partitioning and property table was used to store the RDF data and query execution time is analyzed. The experiment was carried out on a very small dummy data and now it will be scaled up using Barton library catalogue.

Keywords- *Data Partitioning, Semantic Web, Query performance, Query Execution time, Scalability*

I. INTRODUCTION

The Semantic Web is an effort by the W3C to enable integration and sharing of data across different applications and organizations [1]. It is a contemplation of combination of different content, information applications and systems across internet. There is a tremendous growth in semantic web data due to its application in real world. Application which relies on semantics in real life has perceived the importance of RDF (Resource description framework) data that is spread across and as a result we can find a rapid growth in semantic web data. Mostly all semantic applications require fetching the data efficiently which is spread over internet and hence different semantic web applications will often retrieve information dynamically from URLs and merge them into a storage system to make the data available. So data manipulation has to be carried out many times, on the scale of the web, and it should be done in efficient manner. Researchers are investigating usefulness of RDBMS tools for this purpose. Efficient management of RDF data is an important factor in realizing the Semantic Web vision. There is need for efficient query processing as performance and scalability issues are becoming increasingly important as Semantic Web technology is applied to real-world applications.

II. LITERATURE REVIEW

There are several data models available to store data. Here database model and semantic web model are of basic concern. These two data models are different from each other. The Semantic Web data model, called the "Resource Description Framework" or RDF. RDF structure of any expression is a collection of triples, each consisting of a subject, a predicate and an object. A set of such triples is called an RDF graph. This can be shown by a node and directed-arc diagram, in which each triple is represented as a node-arc-node link. The graph can be structurally parsed into a set of triples or statements in the form of < subject, predicate, object >. Now Query processing and optimization technologies and other important data management facilities are commonly found in a Relational DBMS, which are not available in an RDF engine. Querying RDF/RDFS documents is based on tree traversal and simple pattern matching. So SQL requests made on relational databases are considered simpler and take less time to formulate than using the RDF-based language such as SPARQL [3]. There are tools to convert RDF data to relational data. R2D aims to transform RDF data, at run-time, into an equivalent normalized relational schema, thereby bridging the gap between RDF and RDBMS concepts and making the abundance of existing relational tools available to RDF Stores [5]. The RDF data can be stored in relational representation as property tables used by Jena semantic web tool kit. This data representation, though flexible, has the potential for serious performance issues, since there is only one single RDF table, and almost all interesting queries involve many self-joins over this table. Property table technique used by jena2 semantic web toolkit de normalizes RDF tables by physically storing them in a wider, flattened representation more similar to traditional relational schemas. Property tables have several limitations which include problem of NULL values because not all properties

will be defined for all subjects in the subject cluster, wide tables will have possibly many NULLs. For very wide tables with many sparse attributes, the space overhead of these NULLs can be potentially large as compared to the data itself. Multi-valued attributes, many-to-many relationships are somewhat awkward to represent in flattened relation. Proliferation of union clauses and joins because the query does not restrict on property value, or if the value of the property will be bound when the query is processed, all flattened tables will have to be queried and the results combined with either complex union clauses, or through joins.[1]. To overcome this, data partitioning techniques can be used to store RDF data and query the data efficiently. Column data store and row data store are used to store this RDF data. Column oriented database systems have been shown to perform more than an order of magnitude better than traditional row-oriented database systems on analytical workloads such as those found in data warehouses, decision support, and business intelligence applications. RDF documents and schemas (RDFS) are used to describe information in the semantic web. Web researchers regard the RDF/RDFS documents as databases and have proposed data manipulations for them [2].

The paper focuses on the experiment carried out on small set of data, which is discussed in section IV. Details of the further research to be carried out and experimental setup are given in section V and VI respectively.

III. RELATED WORK

Partitioning data for semantic web ensures better performance is stated by Daniel Abadi where he shows that a vertical partitioned schema achieves similar performance to the property table technique while being much simpler to design [1]. They made Performance comparison of the triple-store schema with the property table and vertically partitioned schemas in C-Store. In order to easily manipulate the database, RDF/RDFS documents are transformed into relational database format so that relational languages, data management and business intelligence facilities which are readily available can be exploited [3]. After experimental evaluation for RDF data and it is stated that still room for optimization in the proposed generic relational RDF storage schemes and thus new techniques for storing and querying RDF data are still required to bring forward the Semantic Web vision [6]

IV. EXPERIMENT

An experiment was carried out to find out, in detail, how partitioning affects execution time. A database was created in which five subjects, five properties and five objects were considered. It was a dummy FOAF (Friend of a Friend) data set. The properties included in this experiment were name, email, homepage, knows and interest. In which the first three are single valued attributes where as rest of the two are multi valued attributes. Initially these triples were stored as property table in the database and the following set of queries were executed on it.

Query 1: Find person, the person he knows and 2nd person's homepage.

Query 2: Find person1, person1 email, and person2, person2 email where person1 knows person2.

Query 3: Find people in their circle with common interest.

After execution of the queries on property tables, data was partitioned to evaluate query performance. The data was partitioned using vertical and hybrid approach.

A. Vertical Partitioning

The data was partitioned using vertical partitioning. For every property, a table was created. As we have used five properties we created five tables. For each property, these tables were of the form:

Subject	Predicate

After creating tables and inserting data in these tables, the above queries were executed to evaluate the query performance.

B. Hybrid Partitioning

The data was partitioned using hybrid partitioning technique. For this, properties having single valued attribute were put in a single table with its subject, and rest of the two having multi valued attributes were having separate tables. So the three tables were:

Subject	Name	Email	Homepage

Subject	Knows

Subject	Interest

After creating and populating the tables, the same set of queries was executed on these tables to evaluate query performance. Results of query execution time for all the above discussed technique is depicted in Table 1.

Query	Vertical Partitioning (Execution time is in ms)	Hybrid Partitioning (Execution time is in ms)	Property Table (Execution time is in ms)
Query 1	14	16	16
Query 2	15	15	16
Query 3	15	16	16

Table 1: Query Execution time for various partitioning techniques

Fig 1 represents the column chart which compares the query execution time.

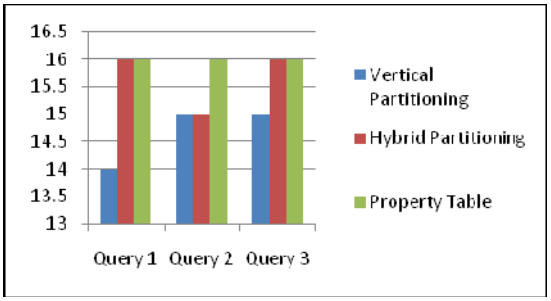

Fig. 1: Query execution time v/s data partitioning technique

It can be seen from the Fig.1 that after partitioning the data, the query execution time has decreased compared to flat representation of RDF data in property tables. In case of vertical partitioning it can be seen that number of records per table are less compared to property table. Secondly, number of joins is similar in both property table approach as well as partitioned approach, but the performance is better or equal due to less complexity and number of records. Even though the number of joins is same, comparison is reduced while retrieving query result and hence it results in better or equal performance. Partitioning approach can reduce number of joins and as a result, better performance can be achieved in terms of execution time and complexity.

The clear advantage of this approach is its simple design and less complexity. This results in simpler way of storing RDF data giving efficient or equally efficient result for queries. This approach can be used or studied more to improve query performance and deal with scalability issues.

The research intends to conduct experiment on Barton libraries dataset [4] as an RDF benchmark data. Partitioning techniques will be applied on this data set and query performance will be evaluated in terms of query execution time and scalability.

V. RESEARCH METHODOLOGY

Barton data set [4, 7], is to be used as benchmark experiment database which have 50,000,000 triples. Data is converted to triples using Jena parser. As Barton data set is huge in size that is around 6 G.B after decompression and goes till 750 GB after converting the data to triple. A tool can be created to insert data in postgres. Here postgres will be used as database tool to store or replicate data in vertical or horizontal partitioning. As the data is quite huge at present the work will be carried on FOAF dataset which contains about 201616 triple. The flow of the work is depicted in Fig 2.

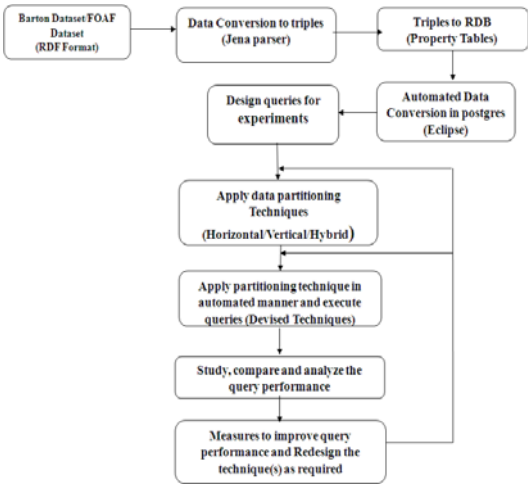

Fig. 2: Flow of activities for research work

VI. ABOUT TEST BED

For further experimentation, the benchmark dataset will be used as specified in the research methodology. Jena parser is used to convert RDF data into triples. Eclipse is used to write java code, for the experiment to be carried out and postgres is used as an RDBMS tool to store RDF data as well as partitioned data. To understand the nature of Barton library dataset [7], Simile group tools will be used which includes welkin [9] - RDF graphical browser, RDFizer [10] to convert data to RDF and Lonwell [8] is an RDF browser to visualize RDF data.

Table 2 represents the tools and technologies to be used for this experiment.

Data Set	FOAF Dataset, Barton Dataset[7]
Software Tools	Eclipse IDE, Java 1.6 SDK, Jena Parser 2.3, Postgres 8.2
Simile MIT Group Tools	Longwell 1.0.1 (Rdf browser), Rdfizer (converts marcmodsToRdf), Welkin (Rdf graphical browser) [8,9,10,11]

Table 2: Tools and technologies

VII. CONCLUSION

The work demonstrated effect of partitioning techniques using dummy data on query performance for semantic web. It has been seen from the experiment carried out, that partitioning data yields better query performance in terms of execution time. The partitioning approach is simpler and results in improved query performance. Future work includes experimentation on FOAF and Barton benchmark databases. Scalability and performance of the queries will be the main focus of the study so it will include techniques to partition the data such that performance of the queries can be evaluated.

REFERENCES

[1] Daniel J. Abadi , Adam Marcus , Samuel R. Madden , Kate Hollenbach, SW-Store: a vertically partitioned DBMS for Semantic Web data management, The VLDB Journal — The International Journal on Very Large Data Bases, v.18 n.2, p.385-406, April 2009

[2] Daniel J. Abadi , Samuel R. Madden , Nabil Hachem, Column-stores vs. row-stores: how different are they really?, Proceedings of the 2008 ACM SIGMOD international conference on Management of data, June 09-12, 2008, Vancouver, Canada

[3] Wajee Teswanich, Suphamit Chittayasothora, "A Transformation from RDF Documents and Schemas to Relational Databases", Proc.IEEE Symp. Computational Intelligence in Scheduling, IEEE press, Dec.2007, pp.38-41.

[4] Abadi, A. Marcus, S. Madden, and K. Hollenbach. "Using the Barton libraries dataset as an RDF benchmark", Technical Report MIT-CSAILTR- 2007-036, MIT.

[5] Sunitha Ramanujam , Anubha Gupta , Latifur Khan , Steven Seida , Bhavani Thuraisingham, “R2D: A Bridge between the Semantic Web and Relational Visualization Tools”, Proceedings of the 2009 IEEE International Conference on Semantic Computing, p.303-311, September 14-16, 2009.

[6] Hooran MahmoudiNasab , Sherif Sakr, An experimental evaluation of relational RDF storage and querying techniques, Proceedings of the 15th international conference on Database systems for advanced applications, April 01-04, 2010, Tsukuba, Japan

[7] Simile Group Home page, Available: http:// simile.mit.edu/wiki/Dataset:_Barton [Dec 3, 2011]

[8] Metadata Object Description Schema User guide, Available: http://www.loc.gov/ standards/mods/v3/mods-userguide.html [Dec 3,2011].

❑❑❑