Storing Large Scale Ontology in Relational Database

Song-Qing WANG, Zhuang MIAO*, Ying-Peng DU, Yang LI, Jia-Bao WANG

College of Command Information Systems, PLA University of Science and Technology, Nanjing, China, 210014
emiaoe_beyond@163.com
*Corresponding author

Keywords: Ontology, Relational Database, Storage Model.

Abstract. To overcome the limitations of the relational database for storing the ontology in four modes, by analyzing the various attributes common ontology, drawing vertical model stable structure, the advantages of high resolution model query efficiency, design and implement a storage mode, and finally according to LUBM storage time consuming to design experiments to test the storage model set, storage space consuming, query efficiency for verification. Experimental results show that the storage model can provide good support for large-scale ontology storage and query data.

Introduction

With the increasing complexity of the ontology to increase the number of instances and the ontology structure of the ontology reasonable storage and efficient query becomes the theory and application value has. Relational database development more mature technologies to improve storage capacity and less restricted circumstances, the ideal storage and query efficiency and data manipulation has guaranteed the accuracy and precision of the transaction system, based on a relational database storage is more suitable for handling large-scale ontology.

In this paper, the current storage model for relational database storage ontology were analyzed and compared, and then used statistical analysis of the properties of the ontology, combined with the advantages of existing models proposed storage model design principles as the basis for achieving a storage model ontology. Finally, the design of experiments for the proposed storage model was validated.

Related Works

Differences between ontology model and the relational model, resulting in a relational database can store multiple patterns ontology [1]. Existing storage model can be divided into four kinds, namely, horizontal mode, vertical model decomposition and mixed mode.

Level Model

Level model is stored in the ontology of the simplest, most easy-to-operate model. This storage model in the database requires only a common data table to store all the data. The data in
the table, in the instance, the instance name, instance attributes as the column of the table. How many object properties, there are that many correspondingly field. Existing relational database system, the number of columns in the data table is limited, far unable to meet the storage requirements of large-scale ontology. Since the horizontal model own limitations, the number of attributes for a difference of more ontology, the sparse structure of the data table, there are a lot of empty field, resulting in a waste of storage space.

**Vertical Model**

Vertical model contains a triplet table, its data model is essentially a binary relation. Because multiple simple binary relationship can be expressed in any complex relationship, so that the data model can be used as the basis of any other models of complex relational model. In the data model, the ontology needs all the information represented in the form of triples become available for storage. The advantage of this model is designed to be simple, common good, etc., resulting in a lot of advantages because they could not appear to attribute entries listed in the table, will not change with ontology patterns change. When the ontology changes only need to update the corresponding tuples to be modified, so its stability and scalability is good. But this pattern for each query ontology must search for all entries, you need to connect a lot of the data table operation, so the disadvantage vertical model is the query efficiency is relatively low.

**Decomposition Model**

Decomposition model is based on multiple data tables for ontology storage, which is above the main differences between the models. Decomposition pattern database schema decomposition, the following two methods.

The first method is based on the class of the pattern decomposition. The main idea of this method is for each class appearing in the ontology create a separate data table. Although this method has a clear structure, but once the ontology type or attribute changes, the structure of the table will change. This change leads to system instability.

The second method is based on the properties of model decomposition. This method is to create a separate data table for each attribute in the ontology. It builds a data table for each attribute in the ontology. With the change of the properties, stored data table in this way will change, which leads to changes in the data structure, time-consuming, affect stability.

Mixed model of the three modes in combination, to achieve the purpose of storage, this model is more flexible, there is no fixed structure. Existing hybrid mode, Pan [2] proposed a hybrid model DLDB, create a data sheet for each ontology type, and create a data sheet for each attribute ontology; Lehman [3] also proposed a hybrid model, the ontology contains all the classes to create a data table, and based on the properties of the OWL language is created that contains the data table.

**Performance Comparison**

In the four modes, horizontal model structure is unstable and a waste of storage space, vertical model query efficiency, decomposition model structure is unstable and limited number of data tables, the existence of complex queries consuming higher problem existing mixed mode. These shortcomings make the existing model is not suitable for large-scale ontology
storage. We compare the existing storage models, from the query efficiency, comprehensibility and scalability three aspects [4]. The comparison of four models is shown as Table 1.

**Table 1. The Comparison of Four Models.**

<table>
<thead>
<tr>
<th>model</th>
<th>shortcoming</th>
<th>advantage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Level model</td>
<td>Storage space is huge and the structure is unstable.</td>
<td>The speed of querying the same type is high.</td>
</tr>
<tr>
<td>Vertical model</td>
<td>Querying costs much time.</td>
<td>The structure of triples is stable.</td>
</tr>
<tr>
<td>Class-based decomposition model</td>
<td>The number of tables is limited and the structure is unstable.</td>
<td>The speed of querying the same type is high.</td>
</tr>
<tr>
<td>Attribute-based decomposition model</td>
<td>The number of tables is limited and the structure is unstable.</td>
<td>The speed of simple querying is high.</td>
</tr>
</tbody>
</table>

**Design of Large Scale Ontology Storage Model**

During storage model ontology design, the impact of different attributes of the ontology will be different. Literature [5] pointed out, property type and sub-type relationship of the ontology to store an impact, and be verified by experiment. Based on this idea, we designed the model, through statistical analysis of some common ontology of each attribute, combined with the advantages of vertical model decomposition patterns, draw the appropriate design principles as the basis for design of a storage model for ontology storage, and finally realized this storage mode.

**Attribute Analysis of Common Ontology**

We choose four common ontologies for attribute analysis, namely CYC [6], SUMO [7], WordNet [8], Teknowledge [9]. We analyzed several attributes that appear in the ontology, including label, comment, type, different from, inverse of, same as etc. The relationship between the ontology of a statistical properties appear, as shown in Fig. 1. We can see, in a different ontology, although the proportion of each property is different, but the predominant roughly the same proportion of the property, including the type, subclass of, subproperty of, domain, range properties.
Figure 1. Attribute Analysis of Common Ontology.

For higher frequency properties appear as type, subclass of, subproperty of, domain, range, can be expressed by a histogram, which is more intuitive, as shown in Fig. 2.

Figure 2. The Main Attributes Proportion.

Property used by the ontology to carry out analysis shows that type, subclass of, subproperty of, domain, range and other properties appear more frequently, a high proportion; few other different from, same as, inverse of other attributes such as the number appeared, the proportion of the more low. For different ontology, different though related fields of expression and describe the different expertise, but in the description of the ontology of knowledge, the proportion of the main attributes appear roughly the same.

Design of Model

Through analysis and comparison of four models, a comprehensive analysis of the results of the common property of the ontology, in the design of the relational data model to store the ontology, should meet three principles, same as the storage structure clear and reasonable, model’s stability is good and the efficiency of querying ontology is high. We design and implement a relational database storage model ontology. The database schema is consisted of 10 tables. The connection relationship between the data tables as shown in Fig. 3.
In the schema above, the properties of the model are inseparable, and each of which is neither part of the non-primary attribute functional dependencies, nor the transfer function dependencies, therefore, by definition, the model is the third paradigm. The storage model takes the form of triples, with the evolution of the ontology, add or delete the information you need only to ontology data table tuples be modified without modifying the table field, maintaining a stable pattern.

**Experiment and Discussion**

In the experiment, LUBM is a test set [10]. LUBM test set is to use the ontology data generator UBA-1.7 generation college ontology. UBA-1.7 provides a number of parameters including: UNDER_COURSE_NUM, GRAD_COURSE_NUM, UNI_NUM, RESEARCH_NUM etc., by modifying these parameters can produce the ontology making UBA-1.7 file contains a different number of classes and properties.

Experimental verification of the relational database to store the ontology mainly from the following aspect. Time-consuming and the size of the storage space to store relational database experimental analysis. LUBM test set clear requirements for storage when tested, the need for time-consuming relational database storage to store the ontology and the size of storage space experiment analysis. We stored in different ontology sizes, time-consuming and space-consuming storage comparison.

Liu [11] compared existing Sesame-DB, Jena-DB, and other tools in storage time consuming and space-consuming and we compared with his works. In order to vary the impact ontology structure while minimizing the experimental result, the size of each group of three similar experiments are used, the structure of the ontology for storage vary consuming and space consuming experiments, the average value as the result of the set of experiments the final result. This difference is mainly manifested in the ontology structure of the UBA-1.7 on generator configuration parameters.

We adopts three experiments: a first pair of left and right ontology 2M stored, a second pair of left and right ontology 72M stores, a third group of stored around the main ontology 167M. In the experiment, the time of this cost calculation for storage space, storage space cost includes not only the size of the data file is stored in the database also includes data tables index size; storage cost and time-consuming file was started by reading the ontology, until all data stored in the database program running time. A first pair of left and right ontology 2M stored, three
experiments conducted, the first ontology size 2106kb, subsequently twice the size of the ontology, respectively 2124kb and 2130kb. The first set of experiments, three storage consuming storage space below, in megabytes. The comparisons of storage space cost and storage process time are shown in fig. 4 and fig.5 distinctively.

There are two main storage tool storage model this article is verified by comparing the storage model in this paper has some advantages, which are stored in the memory consuming model is superior to Sesame-DB, storage space consuming superior Jena-DB balance the relationship between the cost and time-consuming storage space between. The main reason for this phenomenon: the time-consuming aspects of storage, Sesame-DB during storage when using a policy storage node ID, the need for duplicate ID detection for each node, and this process is not used effectively measures that increase the storage time consuming. In terms of space-consuming, Jena-DB of a table when retrieving data in the storage will increase the amount of data increases rapidly, resulting in the overall amount of data to more general multi-table the amount of the generated data structure.

**Conclusion**

In this paper, with examples of current relational database to store ontology methods were introduced, summarizes the advantages and disadvantages of each model. Followed by the statistical properties of the common ontology analysis, the model design principles. Then according to design principles, design and implement a storage mode. Finally, the experimental program designed to validate storage under different conditions consuming, space-consuming and query time to prove the superiority of the proposed model.

**References**


