Semantic Data Access to Relational Databases

Presenter: Li Ma (malli@cn.ibm.com)
Semantic Data Access (SeDA) Engine

- SeDA provides enabling technologies for exposing relational data as virtual RDF graphs, drives semantic federation and integration and realizes the linked data vision of semantic web.

- Applications and Status
  - Status: SeDA v0.2 is available at IBM community sources and will be released at IBM alphaworks around Oct.
  - SeDA is a core component for Metadata Web and Semantic Master Data Management projects.

- Technical Objectives
  - Develop high-performance and scalable RDF query rewriting techniques for the generation of virtual RDF views over existing relational data
  - Develop mapping based ontology and rule reasoning for semantic querying over existing relational data

![Semantic Data Access Engine Diagram]
Outline

- Value Proposition of SeDA
  - Driving information federation and integration
  - Semantic querying
- Technical Solutions
- Mapping Generation from Object Models
- Semantic Web Tools
Linked Data: A Global Database at Semantic Web

- A global database
  - Various resources (including documents) are linked together
  - Degree of structure in resources is high
  - Semantics of content and links are explicit
  - Designed for machines first, humans later

- Use URIs as names for resources
- Use HTTP URIs so that people can look up those names
- When someone looks up a URI, provide useful RDF information
- Include RDF statements that link to other URIs so that they can discover related things

Reference: Christian Bizer et al., Linked data presentation
The Data on Semantic Web

Sources: T.B. Lee's presentation
Semantic Querying over Relational Databases

Company info.

<table>
<thead>
<tr>
<th>ID</th>
<th>Name</th>
<th>Location</th>
<th>Business1</th>
<th>Business2</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>BAR</td>
<td>Bei Jing</td>
<td>Memory</td>
<td>Wireless software</td>
</tr>
<tr>
<td>3</td>
<td>ROL</td>
<td>New York</td>
<td>Banking Solut.</td>
<td>NULL</td>
</tr>
<tr>
<td>4</td>
<td>EDOX</td>
<td>New York</td>
<td>Memory</td>
<td>Main Board</td>
</tr>
<tr>
<td>5</td>
<td>GUC</td>
<td>Vancouver</td>
<td>NULL</td>
<td>NULL</td>
</tr>
</tbody>
</table>

Shareholding

<table>
<thead>
<tr>
<th>ID</th>
<th>Name</th>
<th>Shareholders1</th>
<th>Shareholders2</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>BAR</td>
<td>FOO</td>
<td>TIT</td>
</tr>
<tr>
<td>2</td>
<td>FOO</td>
<td>GUC</td>
<td>Null</td>
</tr>
<tr>
<td>3</td>
<td>ROL</td>
<td>BAR</td>
<td>TIT</td>
</tr>
<tr>
<td>4</td>
<td>EDOX</td>
<td>ROL</td>
<td>Null</td>
</tr>
</tbody>
</table>

Semantic query

1. Find Company EDOX’s all direct and indirect shareholders who are from Europe and are IT company.

2. A software company that has products about wireless telecom and is held by a Canada company
Semantic Querying over Relational Databases

Semantic query
1. Find Company EDOX’s all direct and indirect shareholders who are from Europe and are IT company.

2. A software company that has products about wireless telecom and is held by a Canada company
Semantic Querying over Relational Databases

**Company info.**

<table>
<thead>
<tr>
<th>ID</th>
<th>Name</th>
<th>Location</th>
<th>Business1</th>
<th>Business2</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>BAR</td>
<td>Bei Jing</td>
<td>Memory</td>
<td>Wireless software</td>
</tr>
<tr>
<td>3</td>
<td>ROL</td>
<td>New York</td>
<td>Banking Solut.</td>
<td>NULL</td>
</tr>
<tr>
<td>4</td>
<td>EDOX</td>
<td>New York</td>
<td>Memory</td>
<td>Main Board</td>
</tr>
<tr>
<td>5</td>
<td>GUC</td>
<td>Vancouver</td>
<td>NULL</td>
<td>NULL</td>
</tr>
</tbody>
</table>

**Shareholding**

<table>
<thead>
<tr>
<th>ID</th>
<th>Name</th>
<th>Shareholders1</th>
<th>Shareholders2</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>BAR</td>
<td>FOO</td>
<td>TIT</td>
</tr>
<tr>
<td>2</td>
<td>FOO</td>
<td>GUC</td>
<td>Null</td>
</tr>
<tr>
<td>3</td>
<td>ROL</td>
<td>BAR</td>
<td>TIT</td>
</tr>
<tr>
<td>4</td>
<td>EDOX</td>
<td>ROL</td>
<td>Null</td>
</tr>
</tbody>
</table>

**Semantic query**

1. Find Company EDOX’s all direct and indirect shareholders who are from Europe and are IT company. **FOO is retrieved using transitive closure and subsumption inference.**

2. A software company that has products about wireless telecom and is held by a Canada company. **BAR is retrieved using classification and subsumption inference.**
Data Management and Semantic Web

Main Motivations are in capturing Data Semantics, achieving Data Integration and Reasoning

- RDF and OWL ontologies are good in capturing data semantics
  - Can be used to define a “semantic” model of the underlying relational data that can be tailored to different domains or applications, and that hides the actual layout of data across different tables
- Allow use of additional domain knowledge in OWL ontologies while answering queries to the relational DB
- Allow use of DL reasoning while answering queries to the relational DB to improve recall
- Allow Semantic Web applications (that use an RDF/OWL data model) to have access to relational data, without having to deal with a different data model
- Virtual RDF views facilitates data federation and integration
Major Challenges

- Ontology, rule and mapping
  - Expressive mapping: semantically complete
  - Easy to use semantic modeling: more choices to users

- Query processing and reasoning
  - Effective integration of reasoning with query processing. It is impractical to materialize all inferred results in advance, which causes the data synchronization problem in an operational store.
  - Efficient SPARQL-to-SQL query rewriting based on the ontology mapping
  - Scalable query evaluation by leveraging well-developed database optimizations for query processing
Query Evaluation with Reasoning

Extended SPARQL Query

Parser & Lexer

D2RQ Mapping

MetaData Info.

SQL Generator & Executor

DataLog Engine

Relational DB

Type Of Table Ontology Table

Temp Tables
SPARQL Query Expansion

SPARQL tree will be translated into ONE SQL.

Key idea: Maintain tree structure as much as possible! This is to give DBMS more information to do query optimization.

When the tree structure cannot be kept (recursive rules), trigger the datalog evaluation.

During the evaluation, call the SQL generator repeatedly.

The evaluation result is stored in the temporal table.

SPARQL Pattern Tree

Inference SubTree

Datalog evaluation

Invoke SQL Generator
An Example of Rules and Query

/* Use rules to define that sub-contract property is transitive */
Sub_Contract(?x,?y):-
 WCC:CONTRACT_Relationship_FROM(?u, ?x),
 WCC:CONTRACT_Relationship_TO(?u, ?y),
 WCC:hasContract_Relationship_Type(?u, ?z),
 WCC:Contract_Relationship_FROM_TO_NAME(?z, 'Sub Agreement');;.

Sub_Contract(?x,?y):-Sub_Contract(?x,?z),Sub_Contract(?z,?y);;.

/* The relationship Large_Claim_Agreement indicates that a contract has a claim with the amount more than $5000 */
Large_Claim_Agreement(?clm, ?contract, ?amount):-
 WCC:CLAIM_PAID_Amount(?clm,?amount),
 WCC:CLAIM_CONTRACT_Relationship_To(?u, ?clm),
 WCC:CLAIM_CONTRACT_Relationship_From(?u, ?contract);
 ?amount > "5000"^^xsd:double;.

/* A contract is involved in the large_claim_Agreement relationship if its sub-contract has large claims */
Large_Claim_Agreement(?clm, ?contract, ?amount):-
 Large_Claim_Agreement(?clm, ?subcontract, ?amount),Sub_Contract(?subcontract,?contract);;.

/* Compute the contract and the total number of large claims of the contract */
Contract_NumOfLClms(?contract, ?Agg_numclm):-

/* A contract owner is the party that plays the role of “owner primary” in the component of a contract */
Contract_Owner(?pty, ?contract):-
 WCC:Assemble(?component ?contract),
 WCC:CONTRACT_ROLE_Relationship_To(?u ?component),
 WCC:CONTRACT_ROLE_Relationship_From(?u, ?pty),
 WCC:hasContract_ROLE(?u,?typeCD),
 WCC: CONTRACT_ROLE_NAME(?typeCD 'Owner Primary');;.

Query:

Find the party that is the owner of the “costly” contract
Select ?pty
Where {
  FILTER (?numofClaims >2) 
}
Evaluation Process

Select ?pty
Where {
  FILTER (?numofClaims >2) }

SPARQL Query Tree

```sparql
AND

NAray
Contract_NumOfLClms

NAray
Contract_Owner

FILTER
?numofClaims > 2
```
Evaluation Process

Select ?pty

Where {
  FILTER (?numofClaims > 2)
}

SPARQL Query Tree
Evaluation Process

Select ?pty
Where {
  FILTER (?numofClaims >2)
}

SPARQL Query Tree

Dependency graph

Key | Temporal table object
--- | -------------------
n1  | Sub_contract
Evaluation Process

Select ?pty
Where {
  FILTER (?numofClaims >2) }

SPARQL Query Tree

AND

NAray
Contract_NumOfLClms

AND
NAray
Contract_Owner

FILTER
?numofClaims >2

AND

Evaluate: R1, R2

Dependency graph

Key | Temporal table object
--- | ---------------------
n1  | Sub_contract
Evaluation Process

Select ?pty
Where {
  FILTER (?numofClaims >2)
}

SPARQL Query Tree

AND

NArray
Contract_NumOfLClms

NArray
Contract_Owner

FILTER

tmpPattern:n3

AND

n1
Sub_contract

n2
Large_Claim_Agreement

n3
Contract_NumOfLClms

n4
Contract_Owner

Evaluate: R1, R2
Evaluate: R3, R4
Evaluate: R5

Dependency graph

Temporal table object

<table>
<thead>
<tr>
<th>Key</th>
<th>Temporal table object</th>
</tr>
</thead>
<tbody>
<tr>
<td>n1</td>
<td>Sub_contract</td>
</tr>
<tr>
<td>n2</td>
<td>Large_Claim_Agreement</td>
</tr>
<tr>
<td>n3</td>
<td>Contract_NumOfLClms</td>
</tr>
</tbody>
</table>
Object Relational Mapping (ORM) has been a more and more popular technique for objects persistence.

Sources: Jing Mei's presentation
Object Relational Mapping (ORM) has been a more and more popular technique for objects persistence.
Object Relational Mapping (ORM) has been a more and more popular technique for objects persistence.
Mapping Generation from Object Models

**OO Models**

- OR Mapping Generator
- Model Transformation

**OR Mapping**

- ORM engine
- OO programming

**Relational Schema**

- OR Mapping
- D2R engine

**OWL Ontology**

- Semantic Mapping Generator

**Semantic Mapping**

- SW style access

Sources: Jing Mei’s presentation
Semantic Web Tools

- **SOR@IBM Alphaworks** ([http://www.alphaworks.ibm.com/tech/semanticstk](http://www.alphaworks.ibm.com/tech/semanticstk))
  SOR is a high performance Semantic Web data management system on top of DB2, including efficient schema and indexes design for storage, practical ontology reasoning support, and an effective SPARQL-to-SQL translation method for RDF query.

- **SHER@IBM Alphaworks** ([http://www.alphaworks.ibm.com/tech/sher](http://www.alphaworks.ibm.com/tech/sher))
  Scalable Highly Expressive Reasoner (SHER) is a breakthrough technology that provides ontology analytics over highly expressive ontologies (OWL-DL without nominals). SHER does not do any inferencing on load; hence it deals better with quickly changing data (the downside is, of course, that reasoning is performed at query time). The tool can reason on approximately seven million triples in seconds, and it scales to data sets with 60 million triples, responding to queries in minutes. It has been used to semantically index 300 million triples from medical literature.
  SHER tolerates logical inconsistencies in the data, and it can quickly point you to these inconsistencies in the data and help you clean up inconsistencies before issuing semantic queries. The tool explains (or justifies) why a particular result set is an answer to the query; this explanation is useful for validation by domain experts.
A Pluggable Semantic Object Repository

- RDF documents
- TBox Translator
- Query Adaptor
- DL Reasoner
- Membership and relationship query
- ABox Summarizer
- ABox Filters
- Reasoning
- SHER
- SPARQL Processor
- Query Adaptor
- Storage
- Persistent Store
- OWL Parser
- Import
- DB Translator
- Insert Tbox
- Retrieve Tbox
- Insert Abox assertions into DB
- Generate EODM models from documents
- Load and traverse EODM Tbox
- Load and traverse EODM Abpx
- Return results by SQLs
- SPARQL queries and results
- Reasoning task
- Generate reasoning task
- Insert data for reasoning
- Retrieve data for query answering & reasoning
- Insert Tbox
- Retrieve subsumption
- Return results
- Generate SPARQL memory model
- Import
- Storage
- Database
- Query Answering
- Users
- Reasoning
- Insert data for reasoning
- Generate EODM models from documents
- Load and traverse EODM Tbox
- Load and traverse EODM Abpx
- Return results by SQLs
- SPARQL queries and results
- Reasoning task
- Generate reasoning task
- Insert data for reasoning
- Generate SPARQL memory model
- Return results
- Generate SPARQL memory model
- Return results
Recent Publications


- Li Ma, Chen Wang, Jing Lu, Feng Cao, Yue Pan, Yong Yu, "Effective and Efficient Semantic Web Data Management over DB2", SIGMOD 2008, pp. 1183-1194.

- Ying Yan, Chen Wang, Aoying Zhou, Weining Qian, Li Ma, Yue Pan, "Efficiently querying rdf data in triple stores", poster, WWW 2008, pp. 1053-1054.

- Robert Lu, Feng Cao, Li Ma, Yong Yu and Yue Pan, "An Effective SPARQL Support over Relational Databases", VLDB2007 Joint ODBIS & SWDB workshop on Semantic Web, Ontologies, Databases.