RDF Stores
and Graph Databases

Mustafa Jarrar
Birzeit University
Watch this lecture and download the slides

Online Courses: http://www.jarrar.info/courses/
RDF Stores

Part 1: Querying RDF(S P O) tables using SQL

Part 2: Practical Session (RDF graphs)

Part 3: SQL-based RDF Stores

Part 4: MashQL (Graph Query Formulation)

Keywords: RDF, RDF Stores, querying SPO Table, Graph Databases, Querying Graph, self joins, RDF3X, Oracle Semantic Technology, C-Store, vertical patronizing, MashQL, Graph Signature, Semantic Web, Data Web,
RDF (Resource Description Framework) is the Data Model behind the Semantic/Data Web.

RDF is a graph-based data model.

In RDF, Data is represented as triples: <Subject, Predicate, Object>, or in short: <S,P,O>.
RDF as a Data Model

RDF’s way of representing data as triples is more elementary than Databases or XML.

- This enables easy data integration and interoperability between systems (as will be demonstrated later).

• EXAMPLE: the fact that the movie *Sicko* (2007) is directed by Michael Moore from the USA can be represented by the following triples which form a directed labeled graph:

```xml
< M1, name, Sicko >
< M1, year, 2007 >
< M1, directedBy, D1 >
< D1, name, Michael Moore >
< D1, country, C1 >
< C1, name, USA >
```
RDF graphs as an SPO table

An RDF Graph can stored in one long and thin <S,P,O> table in RDBMS
RDF graphs as an SPO table

How do we query graph-shaped data stored in one relational table?

How do we traverse a graph stored in one relational table?

→ Using self joins!
How to Query RDF data stored in SPO table?

<table>
<thead>
<tr>
<th>S</th>
<th>P</th>
<th>O</th>
</tr>
</thead>
<tbody>
<tr>
<td>M1</td>
<td>year</td>
<td>2007</td>
</tr>
<tr>
<td>M1</td>
<td>Name</td>
<td>Sicko</td>
</tr>
<tr>
<td>M1</td>
<td>directedBy</td>
<td>D1</td>
</tr>
<tr>
<td>M2</td>
<td>directedBy</td>
<td>D1</td>
</tr>
<tr>
<td>M2</td>
<td>Year</td>
<td>2009</td>
</tr>
<tr>
<td>M2</td>
<td>Name</td>
<td>Capitalism</td>
</tr>
<tr>
<td>M3</td>
<td>Year</td>
<td>1995</td>
</tr>
<tr>
<td>M3</td>
<td>directedBy</td>
<td>D2</td>
</tr>
<tr>
<td>M3</td>
<td>Name</td>
<td>Brave Heart</td>
</tr>
<tr>
<td>M4</td>
<td>Year</td>
<td>2007</td>
</tr>
<tr>
<td>M4</td>
<td>directedBy</td>
<td>D3</td>
</tr>
<tr>
<td>M4</td>
<td>Name</td>
<td>Caramel</td>
</tr>
<tr>
<td>D1</td>
<td>Name</td>
<td>Michael Moore</td>
</tr>
<tr>
<td>D1</td>
<td>hasWonPrizeIn</td>
<td>P1</td>
</tr>
</tbody>
</table>

Exercises:

1. What is the name of director D3?
2. What is the name of the director of the movie M1?
3. List all the movies who have directors from the USA and their directors.
4. List all the names of the directors from Lebanon who have won prizes and the prizes they have won.
How to Query RDF data stored in SPO table?

Exercise 1: A simple query.
- What is the name of director D3?
- Consider the table MoviesTable(s,p,o).
- SQL:

Select o
From MoviesTable T
Where T.s = 'D3' AND T.p = 'Name';

Answer: Nadine Labaki
How to Query RDF data stored in SPO table?

Exercise 2: A path query with 1 join.

- What is the name of the director of the movie M1.
- Consider the table MoviesTable \((s,p,o)\).
- SQL:

```
Select T2.o
From MoviesTable T1, MoviesTable T2
Where {T1.s = 'M1' AND
T1.p = 'directedBy' AND
T1.o = T2.s AND
T2.p = 'name'};
```

Answer: Michael Moore
How to Query RDF data stored in SPO table?

Exercise 3: A path query with 2 joins.

- List all the movies who have directors from the USA and their directors.
- Consider the table `MoviesTable(s,p,o)`.
- SQL:

```sql
Select T1.s, T1.o
From MoviesTable T1, MoviesTable T2, MoviesTable T3
Where {
  T1.o = T2.s AND
  T2.o = T3.s AND
  T1.p = 'directedBy' AND
  T2.p = 'country' AND
  T3.p = 'name' AND
  T3.o = 'USA'};
```

**Answer:** M1 D1; M2 D1; M3 D2
Exercise 4: Star query.

- List all the names of the directors from Lebanon who have won prizes and the prizes they have won.
- Consider the table MoviesTable (S,P,O).

Select T1.o, T4.o
From MoviesTable T1, MoviesTable T2, MoviesTable T3, MoviesTable T4
Where {T1.p = 'name' AND
T2.s = T1.s AND
T2.p = 'country' AND
T2.o = T3.s AND
T3.p = 'name' AND
T3.o = 'Lebanon' AND
T4.s = T1.s AND
T4.p = 'hasWonPrizeIn'};

Answer: ‘Nadine Labaki’, P3
RDF Stores

Part 1: Querying RDF(S P O) tables using SQL

Part 2: Practical Session (RDF graphs)

Part 3: SQL-based RDF Stores

Part 4: MashQL (Graph Query Formulation)

Keywords: RDF, RDF Stores, querying SPO Table, Graph Databases, Querying Graph, self joins, RDF3X, Oracle Semantic Technology, C-Store, vertical patronizing, MashQL, Graph Signature, Semantic Web, Data Web,
Practical Session

Given the RDF graph in the next slide, do the following:

(1) Insert the data graph as <S,P,O> triples in a 3-column table in any DBMS.

   Schema of the table: Books (Subject, Predicate, Object)

(2) Write the following queries in SQL and execute them over the newly created table:

   • List all the authors born in a country which has the name Palestine.

   • List the names of all authors with the name of their affiliation who are born in a country whose capital’s population is 14M. Note that the author must have an affiliation.

   • List the names of all books whose authors are born in Lebanon along with the name of the author.
This data graph is about books. It talks about four books (BK1-BK4). Information recorded about a book includes data such as; its author, affiliation, country of birth including its capital and the population of its capital.
Practical Session

• Each student should work alone.

• The student is encouraged to first answer each query mentally from the graph before writing and executing the SQL query, and to compare the results of the query execution with his/her expected results.

• The student is strongly recommended to write two additional queries, execute them on the data graph, and hand them along with the required queries.

• The student must highlight problems and challenges of executing queries on graph-shaped data and propose solutions to the problem.

• Each student must expect to present his/her queries at class and, more importantly, he/she must be ready to discuss the problems of querying graph-shaped data and his/her proposed solutions.

• The final delivery should include (i) a snapshot of the built table, (ii) the SQL queries, (iii) The results of the queries, and (iv) a one-page discussion of the problems of querying graph-shaped data and the proposed possible solutions. These must be handed in a report form in PDF Format.
RDF Stores

Part 1: Querying RDF(S P O) tables using SQL

Part 2: Practical Session (RDF graphs)

Part 3: SQL-based RDF Stores

Part 4: MashQL (Graph Query Formulation)

Keywords: RDF, RDF Stores, querying SPO Table, Graph Databases, Querying Graph, self joins, RDF3X, Oracle Semantic Technology, C-Store, vertical patronizing, MashQL, Graph Signature, Semantic Web, Data Web,
Complexity of querying graph-shaped data

Where does the complexity of querying graph-shaped data stored in a relational table lie?

The complexity of querying a data graph lies in the many self-joins that are to be performed on the table.

Accessing multiple properties for a resource requires subject-subject joins (Star-shaped queries).

Path expressions require subject-object joins.

In general, a query with $n$ edges on an SPO table requires $n-1$ self-joins of that table.
Solution-1: Indexes and Dictionaries

This solution is called RDF3X, suggested by:


Build Indexes.
- On each column: S, P, O
- On Permutations of two columns: SP, SO, PS, PO, ...
- On Permutation of three columns: SPO, SOP, PSO, POS, OSP, OPS, ...

Dictionary Encoding String Data?
- Replacing all literals by IDs.
- Triple stores are compressed.
- Allows for faster comparison and matching operations.
Solution-2: Vertically Partitioning

This solution is called C-Store, suggested by:


Six Vertically Partitioned Tables:

- Triples table is rewritten into $n$ two column tables where $n$ is the number of unique properties.
- In each of these tables, the first column contains the subjects that define that property and the second column contains the object values for those subjects.
- An experimental implementation of this approach was performed using an open source column-oriented database system (C-Store).
Solution-3: Subject-Property Matrixes

This solution suggested by Oracle 11g, called **Oracle Semantic Technology**: Chong E, Das S, Eadon G, Srinivasan J: An efficient SQL-based RDF querying scheme. VLDB’05.

**Clustered Property Table:**

- A clustered property table, contains clusters of properties tend to be defined together.
- Multiple property tables with different clusters of properties may be created to optimize queries.
- A key requirement for this property table is that a particular property may only appear in at most one property table.
- Advantage: Some queries can be answered directly from the property table with no joins.
Solution-3: Subject-Property Matrixes

This solution suggested by Oracle 11g, called **Oracle Semantic Technology**:  

Oracle allows you write graph queries inside SQL using the SEM_Match table function ➔

Example

```
SELECT x, y
FROM TABLE(SEM_MATCH(
  '{?x :grandParentOf ?y. ?x rdf:type :Male}',
  SEM_Models('family'),
  SEM_Rulebases('RDFS','family_rb'),
  SEM_ALIASES(SEM_ALIAS('','http://www.example.org/family/')), null));
```

SEM_MATCH(
    query VARCHAR2,
    models SEM_MODELS,
    rulebases SEM_RULEBASES,
    aliases SEM_ALIASES,
    filter VARCHAR2,
    index_status VARCHAR2,
    options VARCHAR2,
    graphs SEM_GRAPHS,
    named_graphs SEM_GRAPHS
  )  RETURN ANYDATASET;
Later: Oracle Spatial and Graph

As part of Oracle Spatial and Graph, Oracle delivers advanced RDF Knowledge Graph data management and analysis.

With native support for World Wide Web Consortium (W3C) standards - RDF and OWL are standards for representing and defining semantic data and SPARQL is a query language designed specifically for graph analysis - application developers benefit from the industry’s leading open, scalable graph data platform on Oracle Database with triple-level security, and high performance and availability. RDF Graphs can create a unified metadata layer for disparate applications that facilitates identification, integration, and discovery. RDF Graphs are becoming central to linked data and social network applications common in healthcare and life sciences, finance, media and intelligence communities.
RDF Stores

Part 1: Querying RDF(S P O) tables using SQL

Part 2: Practical Session (RDF graphs)

Part 3: SQL-based RDF Stores

Part 4: MashQL (Graph Query Formulation)

Keywords: RDF, RDF Stores, querying SPO Table, Graph Databases, Querying Graph, self joins, RDF3X, Oracle Semantic Technology, C-Store, vertical patronizing, MashQL, Graph Signature, Semantic Web, Data Web,
MashQL (Query Formulation & indexing Graphs)

(Research started at the University of Cyprus, then at Birzeit University)

What about summarizing the data graph, and instead of querying the original data, we query the summary and get the same results?

What about summarizing 15M SPO triples into less then 0.5M?

This is called **Graph Signature** Indexing. It is currently in research at Birzeit University.

Initial comparison between GS performance and Oracle’s Semantic Technologies
MashQL (Query Formulation & indexing Graphs)

A graphical query formulation language for the Data Web

A general structured-data retrieval language (not merely an interface)

Addresses all of the following challenges:

- The user does not know the schema.
- There is no offline or inline schema\ontology.
- The query may involve multiple sources.
- The query language is expressive (not a single-purpose interface)
Interactive Query Formulation.
- MashQL queries are translated into and executed as SPARQL.
MashQL Example

Celine Dion’s Albums after 2003?

RDF Input

www.site1.com/rdf
:A1 :Title “Taking Chances”
:A1 :Artist “Dion C.”
:A1 :ProdYear 2007
:A1 :Producer “Peer Astrom”
:A2 :Title “Monodose”
:A2 :Artist “Ziad Rahbani”

www.site2.com/rdf
:B1 rdf:Type bibo:Album
:B1 :Title “The prayer”
:B1 :Artist “Celine Dion”
:B1 :Artist “Josh Groban”
:B1 :Year 2008
:B2 rdf:Type bibo:Album
:B2 :Title “Miracle”
:B2 :Author “Celine Dion”
:B2 :Year 2004
MashQL Example

Celine Dion’s Albums after 2003?

RDF Input

From:
- http://www.site1.com/rdf
- http://www.site2.com/rdf

Query

www.site1.com/rdf
:A1 :Title “Taking Chances”
:A1 :Artist “Dion C.”
:A1 :ProdYear 2007
:A1 :Producer “Peer Astrom”
:A2 :Title “Monodose”
:A2 :Artist “Ziad Rahbani”

www.site2.com/rdf
:B1 rdf:Type bibo:Album
:B1 :Title “The prayer”
:B1 :Artist “Celine Dion”
:B1 :Artist “Josh Groban”
:B1 :Year 2008
:B2 rdf:Type bibo:Album
:B2 :Title “Miracle”
:B2 :Author “Celine Dion”
:B2 :Year 2004

Background queries

SELECT X WHERE {?X ?P ?O}
Union
SELECT X WHERE {?S ?P ?X}

SELECT X WHERE {?S rdf:Type ?X}
MashQL Example

Celine Dion’s Albums after 2003?

**RDF Input**

From:
- http://www.site1.com/rdf
- http://www.site2.com/rdf

**Query**

```
SELECT P WHERE {?Everything ?P ?O}
```

**Background query**

```
SELECT * WHERE {?Everything ?P ?O}
```

www.site1.com/rdf

:A1 :Title “Taking Chances”
:A1 :Artist “Dion C.”
:A1 :ProdYear 2007
:A1 :Producer “Peer Astrom”
:A2 :Title “Monodose”
:A2 :Artist “Ziad Rahbani”

www.site2.com/rdf

:B1 rdf:Type bibo:Album
:B1 :Title “The prayer”
:B1 :Artist “Celine Dion”
:B1 :Artist “Josh Groban”
:B1 :Year 2008
:B2 rdf:Type bibo:Album
:B2 :Title “Miracle”
:B2 :Author “Celine Dion”
:B2 :Year 2004

Jarrar © 2018
Celine Dion’s Albums after 2003?

Background query

```
SELECT P WHERE {?Everything ?P ?O}
```
MashQL Example

Celine Dion’s Albums after 2003?

RDF Input
From:
- http://www.site1.com/rdf
- http://www.site2.com/rdf

Query
Everything
- Artist “Dion
- Year|ProdYear > 2003

Background query
SELECT P WHERE {?Everything ?P ?O}
MashQL Example

Celine Dion’s Albums after 2003?

RDF Input

FROM:
- http://www.site1.com/rdf
- http://www.site2.com/rdf

Query

Everything
- Artist "^Dion"
- Year\ProdYear > 2003
- Title ✓AlbumTitle

PREFIX S1: <http://site1.com/rdf>
PREFIX S2: <http://site1.com/rdf>
SELECT ?AlbumTitle
FROM <http://site1.com/rdf>
FROM <http://site2.com/rdf>
WHERE {
  {?X S1:Artist ?X1} UNION {?X S2:Artist ?X1}
  {?X S1:ProdYear ?X2} UNION {?X S2:Year ?X2}
  {{?X S1:Title ?AlbumTitle} UNION {?X S2:Title ?AlbumTitle}}
  FILTER regex(?X1, "^Dion")
  FILTER (?X2 > 2003)}
MashQL Query Optimization

• The major challenge in MashQL is **interactivity**!
• User interaction must be within **100ms**.
• However, querying RDF is typically slow.
• How about dealing with huge datasets!?
• Implemented Solutions (i.e., Oracle) proved to perform weakly specially in MashQL background queries.

→ A query optimization solution was developed., called **Graph Signature Index**.
‘Graph Signature’ optimizes RDF queries by:

Summarizing the RDF dataset
(Algorithms were developed)

Then, executing the queries on the summary instead of the original dataset
(A new execution plan was developed)
Experiments

- Experiments performed on 15M SPO triples (rows) of the Yago Dataset.
- Graph Signature Index of less than 500K was built on the data.
- Two sets of queries were performed, and the results were as follows:

Experiments  
- Table 1: Query Runtime (s) for Graph Signature Index (GS) and Oracle

<table>
<thead>
<tr>
<th>Query</th>
<th>A1</th>
<th>A2</th>
<th>A3</th>
<th>A4</th>
<th>A5</th>
</tr>
</thead>
<tbody>
<tr>
<td>GS</td>
<td>0.344</td>
<td>1.953</td>
<td>5.250</td>
<td>10.234</td>
<td>24.672</td>
</tr>
<tr>
<td>Oracle</td>
<td>110.390</td>
<td>302.672</td>
<td>525.844</td>
<td>702.969</td>
<td>&gt;20min</td>
</tr>
</tbody>
</table>

Queries that cover all MashQL Background queries some queries that might occur as the final queries generated in MashQL:

<table>
<thead>
<tr>
<th>Query</th>
<th>Q1</th>
<th>Q2</th>
<th>Q3</th>
<th>Q4</th>
<th>Q5</th>
<th>Q6</th>
<th>Q7</th>
<th>Q8</th>
<th>Q9</th>
</tr>
</thead>
<tbody>
<tr>
<td>GS</td>
<td>0.441</td>
<td>0.578</td>
<td>0.587</td>
<td>0.556</td>
<td>0.290</td>
<td>0.291</td>
<td>0.587</td>
<td>0.785</td>
<td>2.469</td>
</tr>
</tbody>
</table>
References


