Towards OLAP over Federated RDF Sources

DILSHOD IBRAGIMOV, KATJA HOSE, TORBEN BACH PEDERSEN, ESTEBAN ZIMÁNYI.
Outline

- Intro and Objectives
- Brief Intro to Technologies
- Proposed System
- Test Case
- Source Discovery
- Future Works
More and more data are published on the Web

Business Intelligence tools need to analyze these data

OLAP-style analysis of Linked Data may help in better decision making

Expected challenges are data sources discovery and selection, data collection and integration, and data aggregation and presentation
Objectives

Design, develop, and evaluate an approach for performing OLAP over federated RDF sources

The solution should support the discovery and integration of previously unknown data sources

- Develop a framework for discovering previously unknown data sources necessary for the given data warehouse
- Design and implement a data cube that uses Global Conceptual Schema with newly discovered data sources
- Develop a framework for extracting multidimensional data, aggregating the results, and building an OLAP cube from the RDF sources
Outline

- Intro and Objectives
- Brief Intro to Technologies
- Proposed System
- Test Case
- Source Discovery
- Future Works
What is Data Warehouse

- **Consolidates information** from different data sources, enabling OLAP (OnLine Analytical Processing) to help decision making
- **Maintained separately** from operational database (which uses OLTP – OnLine Transaction Processing)
Languages and Vocabularies

MDX (Multi Dimensional eXpressions) is a language for querying and manipulating multidimensional data stored in OLAP cubes.

RDF (Resource Description Framework) is a standard model for data interchange on the Web.

SPARQL is a query language for RDF. Queries defined in terms of graph patterns that are matched against the directed graph representing the RDF data.

QB4OLAP is a special RDF vocabulary for OLAP Cubes on the Semantic Web.

VoID is an RDF Schema vocabulary for expressing metadata about RDF datasets.
MDX

MDX adopted by majority of OLAP vendors and has become a de-facto standard for OLAP systems

Ex: Show Sales Tax for Fiscal year 2013 for Southwestern states

SELECT
    { [Measures].[Sales Amount], [Measures].[Tax Amount] } ON COLUMNS,
    { [Date].[Fiscal].[Fiscal Year 12], [Date].[Fiscal].[Fiscal Year 13] } ON ROWS
FROM [DW]
WHERE ( [Sales Territory].[Southwest] )
RDF

Statements about resources in the form of subject-predicate-object expressions

<P1> <is called> < Jimmy Wales > 

TurtleExample : <P1> rdf:type foaf:Person .

<P1> foaf:name "Jimmy Wales" .
<P1> foaf:mbox <mailto:jwales@bomis.com> .

RDF extends the linking structure of the Web to use URLs to name the relationship between things
SPARQL Query Language

Developed by W3C Data Access Working Group

Queries defined in terms of graph patterns that are matched against the directed graph representing the RDF data

Ex: *Show name and email of a person*

PREFIX foaf: <http://xmlns.com/foaf/0.1/>

SELECT ?name ?email

WHERE {
  ?person a foaf:Person.
  ?person foaf:name ?name.
  ?person foaf:mbox ?email. }

<table>
<thead>
<tr>
<th>Name</th>
<th>Email</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jimmy Wales</td>
<td><a href="mailto:jwales@bomis.com">mailto:jwales@bomis.com</a></td>
</tr>
</tbody>
</table>
SPARQL Query Language

4 types of queries to retrieve (read) data:
- SELECT
- CONSTRUCT
- ASK
- DESCRIBE

CONSTRUCT defines a pattern template to create a new graph

Ex: Create a triple defining a first name as a whole name of a person

PREFIX foaf: <http://xmlns.com/foaf/0.1/>

CONSTRUCT { ?r foaf:name ?name }
WHERE { ?r foaf:firstName ?name }
QB4OLAP

QB4OLAP is a special RDF vocabulary for OLAP Cubes on the Semantic Web

-- Data structure definition and dimensions
exqb:NorthwindDW a qb:DataStructureDefinition;
qb:component [qb:dimension exqb:Employee];

-- Definition of measures
qb:component [qb:measure exqb:Quantity].

-- Attributes
exqb4o:CompanyName a qb:AttributeProperty;
rdfs:comment "Company Name"@en.

-- Dimension
exqb4o:CustomerDim a rdf:Property, qb:DimensionProperty.
Outline

- Intro and Objectives
- Brief Intro to Technologies
- Proposed System
- Test Case
- Source Discovery
- Future Works
Proposed System

Semantic Query Processor

Global Conceptual Schema

Federated Query Processor

Source Discovery/Schema Builder

MDX

SPARQL

MD RDF

SPARQL

RDF

SPARQL

RDF

MD RDF

SPARQL

RDF

SPARQL

RDF
Proposed System

Source Discovery/Schema Builder is responsible for the discovery of data sources and construction of the Global Conceptual Schema
Global Conceptual Schema defines the high-level view of the system - expressed in QB4OLAP, VoID
Semantic Query Processor (using the Global Conceptual Schema) converts the MDX query to the new format and passes it to the Federated Query Processor.
Proposed System

Federated Query Processor retrieves data from several federated data sources
Outline

- Intro and Objectives
- Brief Intro to Technologies
- Proposed System
- Test Case
- Source Discovery
- Future Works
Test Case – Linked Movie Database

Semantic Web database dedicated to movie-related information

Contains millions of RDF triples with hundreds of thousands of RDF interlinks to other existing web data sources and documents (GeoNames, DBpedia, Yago, etc.)

Publishes linked open data using the D2R Server

Can be accessed by SPARQL clients – SPARQL endpoint: http://data.linkedmdb.org/sparql

Our Data Cube captures only part of the information from LinkedMDB
LinkedMDB Logical schema (partial)

- **editor**
  - PK: editorid
  - editor_name

- **genre**
  - PK: genreid
  - genre_name

- **producer**
  - PK: producerid
  - Producer_name

- **film**
  - PK: filmid
    - title
    - runtime
    - language
    - initial_release_date
    - country
    - writer_writerid
    - director_directorid
    - actor_actorid
    - producerid
    - film_formatid
    - genreid
    - editorid
    - country_id

- **film_format**
  - PK: film_formatid
    - film_format_name

- **actor**
  - PK: actor_actorid
    - actor_name

- **director**
  - PK: director_directorid
    - director_name

- **country**
  - PK: country_id
    - country_capital
    - country_currency
    - country_continent

- **writer**
  - PK: writer_writerid
    - writer_name
Test Case – LinkedMDB Interlinks to GeoNames

Semantic Web database of geographic data, such as countries, administrative divisions, cities, and related information.

Contains over 10 million geographical names with over 9 million populated places.

Can be accessed via mirrored SPARQL endpoint: http://lod2.openlinksw.com/sparql

We retrieve only part of the information from GeoNames.
GeoNames Logical schema (partial)

@prefix gn: <http://www.geonames.org/ontology#>
@prefix wgs84_pos: <http://www.w3.org/2003/01/geo/wgs84_pos#>

<table>
<thead>
<tr>
<th>GeoNames</th>
</tr>
</thead>
<tbody>
<tr>
<td>rdfs:isDefinedBy</td>
</tr>
<tr>
<td>gn:alternateName</td>
</tr>
<tr>
<td>gn:shortName</td>
</tr>
<tr>
<td>gn:officialName</td>
</tr>
<tr>
<td>gn:name</td>
</tr>
<tr>
<td>gn:wikipediaArticle</td>
</tr>
<tr>
<td>gn:population</td>
</tr>
<tr>
<td>wgs84_pos:lat</td>
</tr>
<tr>
<td>wgs84_pos:long</td>
</tr>
<tr>
<td>rdfs:seeAlso</td>
</tr>
</tbody>
</table>
MDX Query and Answer Examples

- Data cube answers queries expressed in MDX
- Ex: Show average runtime for films by director and country

WITH
MEMBER [Measures].[Average Runtime] AS
  Avg([Film].[Director].CurrentMember, [Measures].[Runtime])
SELECT {[Film].[Country].Members} ON COLUMNS,
  {[Film].[Director].Members} ON ROWS
FROM [MoviesDataWarehouse]
WHERE ([Measures].[Average Runtime]);

<table>
<thead>
<tr>
<th>Director</th>
<th>Great Britain</th>
<th>Venezuela</th>
<th>United States</th>
<th>Netherlands</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sally Potter (Director)</td>
<td>86</td>
<td></td>
<td></td>
<td>96</td>
</tr>
<tr>
<td>Robert Aldrich (Director)</td>
<td></td>
<td></td>
<td>88</td>
<td></td>
</tr>
<tr>
<td>Roman Chalbaud (Director)</td>
<td></td>
<td>87</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Roland Joe (Director)</td>
<td>78</td>
<td></td>
<td>83</td>
<td></td>
</tr>
</tbody>
</table>
Proposed System
Data Cube Conceptual Schema

@prefix movie: <http://data.linkedmdb.org/resource/movie/>
@prefix geo: <http://www.geonames.org/ontology#>
Global Conceptual Schema (ex: linkedmdb.org)

The Global Conceptual Schema of the system is defined during the Source Discovery stage.

Contains possible dimensions [Location (City, Country), Date (Year, Quarter, Month), Director, Actor] and possible measures [Runtime, etc.]

The Global Conceptual Schema module needs to have necessary information to query the endpoints

It is proposed to use QB4OLAP and VoID vocabularies since it allows defining dimensions, measures, aggregations, and metadata about datasets
## Data structure definition and dimensions

exqb:MoviesDataCube a qb:DataStructureDefinition ;
void:sparqlEndpoint <http://data.linkedmdb.org/sparql> ;
rdfs:label "Films"@en ;
rdf:type movie:film ;

## Dimensions

qb:component [qb:dimension exqb:Actor] ;
qb:component [qb:dimension exqb:ReleaseDate] ;
qb:component [qb:dimension exqb:Director] ;

## Definition of measures

qb:component [qb:measure exqb:Runtime] ;
qb:component [qb:attribute exqb:FilmName] .
exqb:ReleaseDateDim a qb:DimensionProperty .
exqb:year a qb4o:LevelProperty ;
    skos:closeMatch db:Year ;
    rdfs:comment "Year of the film release date"@en ;
    qb4o:inDimension exqb:ReleaseDateDim .
exqb:ActorDim a qb:DimensionProperty ;
    skos:closeMatch foaf:Person ;
    rdfs:seeAlso owl:sameAs ;
    rdfs:isDefinedBy <http://data.linkedmdb.org/directory/actor> ;
    qb4o:hasAttribute exqb:ActorID ;
    qb4o:hasAttribute exqb:PersonName .
Query Processing

Semantic Query Processor

Federated Query Processor

Global Conceptual Schema

Source Discovery/Schema Builder

RDF

SPARQL

MDX

MDX Answer

SPARQL

RDF

SPARQL

RDF

SPARQL

RDF
Querying Data Sources

Based on the global schema the Federated Query Processor will send SPARQL queries to the data sources.

It is suggested to use `CONSTRUCT` query form to automatically create triples.

The RDF data returned by several sources will be merged into a `QB4OLAP` file.
Retrieving Attributes, Dimensions, and Measures

CONSTRUCT {
  ?movieUrl exqb:FilmName ?movieName .  
  ?country owl:sameAs ?owlCountry .  
  ?movieUrl exqb:Director ?directorID .  
  ?directorID exqb:PersonName ?directorName .
}
WHERE {
  ?country owl:sameAs ?owlCountry .  
  ?directorID rdfs:label ?directorName .
} #(create triples for runtime, film, director, country)
CONSTRUCT Query Results

#actors
<rdf:Description rdf:about="http://data.linkedmdb.org/resource/actor/49727">
  <movie:actor_name>Nayanthara (Actor)</movie:actor_name>
</rdf:Description>
<rdf:Description rdf:about="http://data.linkedmdb.org/resource/actor/33937">
  <movie:actor_name>Mandira Bedi (Actor)</movie:actor_name>
</rdf:Description>

#directors
<rdf:Description rdf:about="http://data.linkedmdb.org/resource/director/177">
  <movie:dir_name>S. Shankar (Director)</movie:dir_name>
</rdf:Description>
CONSTRUCT Query Results

#countries
<rdf:Description rdf:about="http://data.linkedmdb.org/resource/country/IN">
    <owl:sameAs rdf:resource="http://sws.geonames.org/1269750/"/>
</rdf:Description>

#films
<rdf:Description rdf:about="http://data.linkedmdb.org/resource/film/1309">
    <movie:runtime>175</movie:runtime>
    <movie:actor rdf:resource="http://data.linkedmdb.org/resource/actor/36456"/>
    <movie:director rdf:resource="http://data.linkedmdb.org/resource/director/177"/>
    <movie:name>Jeans</movie:name>
    <movie:country rdf:resource="http://data.linkedmdb.org/resource/country/IN"/>
</rdf:Description>
Example of Results in QB4OLAP Format

<http://data.linkedmdb.org/resource/film/810> a qb:Observation;
  qb:dataSet exqb:MoviesDataWarehouse;
  exqb:Director <http://data.linkedmdb.org/resource/director/8629>;
  exqb:Runtime 188;

http://data.linkedmdb.org/resource/film/930> a qb:Observation;
  qb:dataSet exqb:MoviesDataWarehouse;
  exqb:Director <http://data.linkedmdb.org/resource/director/448>;
  exqb:Runtime 158;

<http://data.linkedmdb.org/resource/country/IN>
  exqb:CountryName "India";
  exqb:CountryCode "IN";
  exqb:Population "1173108018" .

<http://data.linkedmdb.org/resource/director/8629>
  exqb:PersonName "Richard Attenborough (Director)" .

<http://data.linkedmdb.org/resource/director/448>
  exqb:PersonName "K. S. Ravikumar (Director)" .
Results Presentation

The MDX query will be answered based on the information stored in QB4OLAP file (qb:Observation)

The measures will be aggregated based on qb4o:AggregateFunction

<table>
<thead>
<tr>
<th></th>
<th>Great Britain</th>
<th>Venezuela</th>
<th>United States</th>
<th>Netherlands</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sally Potter (Director)</td>
<td>86</td>
<td></td>
<td></td>
<td>96</td>
</tr>
<tr>
<td>Robert Aldrich (Director)</td>
<td></td>
<td></td>
<td>88</td>
<td></td>
</tr>
<tr>
<td>Roman Chalbaud (Director)</td>
<td></td>
<td>87</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Roland Joe (Director)</td>
<td>78</td>
<td></td>
<td>83</td>
<td></td>
</tr>
</tbody>
</table>
Outline

- Intro and Objectives
- Brief Intro to Technologies
- Proposed System
- Test Case
- Source Discovery
- Future Works
Source Discovery

Querying **large knowledge bases** (DBpedia, Yago, Freebase)

Querying **Data Management Platforms** (Datahub.io)

Querying **Semantic Web Search Engines** (Sindice.com)
Querying Large Knowledge Bases

- **Show id, label, and count of items where label contains “Film”, sorted by number of items**

```sql
SELECT ?s ?l COUNT(?s) as ?count 
WHERE {
  FILTER(CONTAINS(?l, "Film")) .
  FILTER (lang(?l) = 'en').
  FILTER (!isLiteral(?someobj)).
} ORDER BY DESC(?count) LIMIT 20
```

<table>
<thead>
<tr>
<th>S</th>
<th>L</th>
<th>COUNT</th>
</tr>
</thead>
<tbody>
<tr>
<td><a href="http://rdf.freebase.com/ns/m.02nsjl9">http://rdf.freebase.com/ns/m.02nsjl9</a></td>
<td>Film character</td>
<td>2001832</td>
</tr>
<tr>
<td><a href="http://rdf.freebase.com/ns/film.film_character">http://rdf.freebase.com/ns/film.film_character</a></td>
<td>Film character</td>
<td>1384754</td>
</tr>
<tr>
<td><a href="http://rdf.freebase.com/ns/film.actor">http://rdf.freebase.com/ns/film.actor</a></td>
<td>Film actor</td>
<td>874840</td>
</tr>
<tr>
<td><a href="http://rdf.basekb.com/ns/m.0jsg30">http://rdf.basekb.com/ns/m.0jsg30</a></td>
<td>Film performance</td>
<td>673398</td>
</tr>
<tr>
<td><a href="http://rdf.freebase.com/ns/film.film">http://rdf.freebase.com/ns/film.film</a></td>
<td>Film</td>
<td>557505</td>
</tr>
</tbody>
</table>
Querying Large Knowledge Bases

- After identifying the type of the object the system queries for the instances of this type and then identifies possible properties of the type
- Ex: *Show instances of the type ns:film.film*

```sparql
PREFIX ns: <http://rdf.freebase.com/ns/>
SELECT ?s ?p ?o
WHERE {
    FILTER (lang(?o) = 'en').
}
```

<table>
<thead>
<tr>
<th>S</th>
<th>P</th>
<th>O</th>
</tr>
</thead>
<tbody>
<tr>
<td><a href="http://rdf.freebase.com/ns/m.0pj5t">http://rdf.freebase.com/ns/m.0pj5t</a></td>
<td>rdfs:label</td>
<td>Falling Down</td>
</tr>
<tr>
<td><a href="http://rdf.freebase.com/ns/m.0swhj">http://rdf.freebase.com/ns/m.0swhj</a></td>
<td>rdfs:label</td>
<td>A Charlie Brown Christmas</td>
</tr>
<tr>
<td><a href="http://rdf.freebase.com/ns/m.0m2kd">http://rdf.freebase.com/ns/m.0m2kd</a></td>
<td>rdfs:label</td>
<td>Stand by Me</td>
</tr>
<tr>
<td><a href="http://rdf.freebase.com/ns/m.07cz2">http://rdf.freebase.com/ns/m.07cz2</a></td>
<td>rdfs:label</td>
<td>The Matrix</td>
</tr>
<tr>
<td><a href="http://rdf.freebase.com/ns/m.0prk8">http://rdf.freebase.com/ns/m.0prk8</a></td>
<td>rdfs:label</td>
<td>Hamlet</td>
</tr>
</tbody>
</table>
Querying Large Knowledge Bases

- Ex: *Show properties of the type ns:film.film*

```sql
PREFIX ns: <http://rdf.freebase.com/ns/>
SELECT ?s ?p ?o
WHERE {
  FILTER (?s=<http://rdf.freebase.com/ns/m.0c296>).
}
ns:m.0c296 ns:film.film.country ns:m.0345h ;
ns:film.film.directed_by ns:m.0k181 ;
ns:film.film.edited_by ns:m.07nw1y6 ;
ns:film.film.genre ns:m.05p553 ;
ns:film.film.initial_release_date "2001-04-25"^^xsd:datetime ;
ns:film.film.runtime..film.film_cut.runtime ns:122.0 ;
ns:film.film.starring..film.performance.actor ns:m.01y9t4 ;
ns:film.film.starring..film.performance.actor ns:m.0jtcpc .
```
Querying Data Management Platforms

[uploaded to the Internet Archive](http://archive.org/details/kasabi)
Linked Movie DataBase - [http://data.linkedmdb.org/sparql](http://data.linkedmdb.org/sparql)
Data exposed: Linked Data about Movies
Size of data set: 6,148,121 triples.

### Openness: Open

Mixture of material from Wikipedia, Freebase and Geonames and states on [http://wiki.linkedmdb.org/Main/Licensing](http://wiki.linkedmdb.org/Main/Licensing):

> Content created by (or contributed to) LinkedMDB (interlinking data, in particular) is licensed under the Creative Commons Attributions License (CC-BY). You are free to use CC-BY content as long as you provide proper attribution back to the source (LinkedMDB). Attribution should be given with a link or a reference to LinkedMDB.

DBpedia-Live - [http://live.dbpedia.org/sparql](http://live.dbpedia.org/sparql)
DBpedia.org is a community effort to extract structured information from Wikipedia and to make this information available on the Web. DBpedia allows you to ask sophisticated queries against Wikipedia and to link other
Querying Semantic Web Search Engines

http://api.sindice.com/v3/search?q=Film&format=json&fq=format%3ARDF&page=6&facet.field=domain

```json
{
    "facet_counts": {
        "facet_queries": [],
        "facet_ranges": {},
        "facet_dates": {},
        "facet_fields": {
            "domain_facet": {
                "elbo.ws", 161,
                "waptrick.com", 161,
                "www4.wiwiss.fu-berlin.de", 156,
                "date.linkedmdb.org", 148,
                "era.rkbexplorer.com", 147,
                "b creek.de", 133,
                "kisti.rkbexplorer.com", 131,
                "bulk.resource.org", 129,
                "dbpedia.openlinksw.com", 122,
                "semanticweb.org", 121
            }
        }
    }
}
```
Outline

○ Intro and Objectives
○ Brief Intro to Technologies
○ Proposed System
○ Test Case
○ Source Discovery
○ Future Works
We are now working on...

Implementation of Federated Query Processor

Conversion of MDX to SPARQL, calculation of aggregated values from QB4OLAP cube
Questions?