Meeting notes

- Shu, Ioana, Yanlei
- 12-04-2017

Main points

- Summary over the last week's remaining questions (OLAP operations over AnQ cube)
- Talk about query recommendations algorithms, how to go further

Next

- Reading the paper: **SEEDB**

> Next part is the support materials for this meeting

OLAP operations over RDF graph data

Idea

Relational data warehouse (traditional)

- construct analytical cube with a set of dimensions and measures
- typical OLAP operations: roll-up, drill-down, slice and dice (cube navigation, transforming a cube into another)

Heterogeneous RDF data in a DW setting

- analytical schema (AnS)[1], analytical schema instance[2], and analytical query (AnQ)[3]
- a cube ↔ an AnQ
- OLAP operations: traditional OLAP operations on cubes → AnQ rewritings (The definition of **Extended** AnQ is introduced[4])

Slice

Slice. Given an extended query $Q = \langle c_\Sigma(x, d_1, \ldots, d_n),\ m(x, v),\ \oplus\rangle$, a slice operation over a dimension d_i with value v_i returns the extended query $\langle c_{\Sigma'}(x, d_1, \ldots, d_n), m(x, v), \oplus\rangle$, where $\Sigma' = (\Sigma \setminus \{(d_i, \Sigma(d_i))\}) \cup \{(d_i, \{v_i\})\}$.

- Intuitively, slice operation binds an aggregation dimension to a concret value.

Example:

Q be an extended query corresponding to the query cube of example 8:

$$< C_{\sum}(x, a, c), m(x, y), count >$$

with $\sum = \{(a, \{a\}), (c, \{c\})\}$ (classifier and measure queries are same)
A slice operation on the age dimension a with a value 34 results in replacing extended classifier of Q with:

$$< c_{\sum'}(x, a, c) = \{c(x, 34, c)\} >$$

where:

$$\sum' = \sum \setminus \{(a, \{a\})\} \bigcup \{(a, \{34\})\}$$

Dice

Dice. Similarly, a dice operation on Q and over dimensions $\{d_{i_1}, \ldots, d_{i_k}\}$ and corresponding sets of values $\{S_{i_1}, \ldots, S_{i_k}\}$, returns the query $\langle c_{\Sigma'}(x, d_1, \ldots, d_n), m(x, v), \oplus \rangle$, where:

$$\Sigma' = \Sigma \setminus (\cup_1^k \{(d_j, \Sigma(d_j)\}) \cup (\cup_1^k \{(d_j, S_j)\})$$

- Intuitively, dice operation forces several aggregation dimensions to take values from specific sets.

Example:

Similarly as the example above, but applying a dice operation on both age and city dimensions with values $\{34\}$ for age (y_1) and {Paris, Berlin} for location (y_2) by replacing the extended classifier of Q with:

$$< c_{\sum'}(x, a, c) = \{c(x, 34, "\ Paris\ "), (x, 34, "\ Berlin\ ")\} >$$

where:

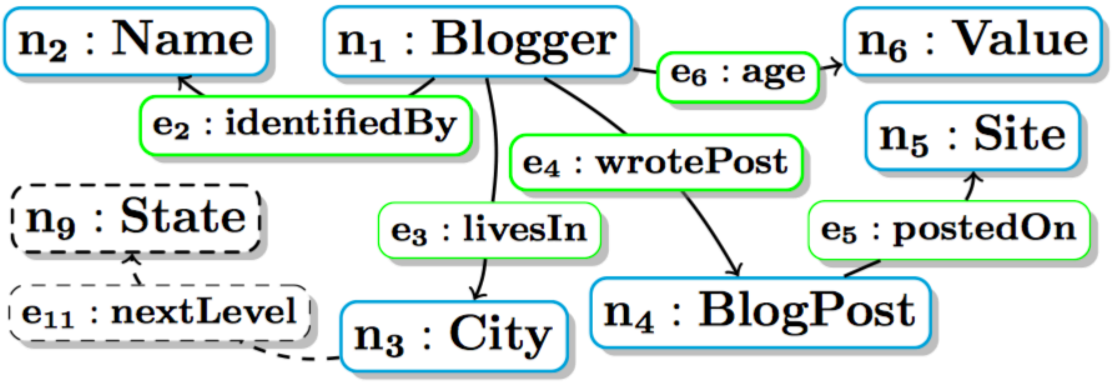
$$\sum' = \sum \setminus \{(y_1, \{y_1\}), (y_2, \{y_2\})\} \bigcup \{(y_1, \{34\}), (y_2, \{"\ Paris\ ", "\ Berlin\ "\})\}$$

Roll-up/Drill-down

To define roll-up and rill-down operations, a new property call **nextLevel** is introduced to model the parent-child diemnsions in a hierarchy structure.

For instance:

- city → state → country → region
- isFriendWith → knows, isCoworkerOf → knows ...
- Example:



Example: Adding a new node and edge to AnS

Here, we added a new node (n_9) and a new edge (e_{11}) to illustrate the next level of **City** can be **State.**
Based on this, we can define roll-up/drill-down operations as **adding to/removing from the classifier, triple atoms** navigating such **nextLevel** edges.
- Example of roll-up, still using the previous example, from City to State, we got:

$$< c'_{\Sigma'}(x, y_1, y_3), m(x, z), count >$$

where

$$c'_{\Sigma'}(x, y_1, y_3) \ :- \ x\ age\ y_1, x\ livesIn\ y_2, y_2\ nextLevel\ y_3$$

- **Remarks** of the example above: the head and body of the query has changed!

Example:

Drill-in and Drill-out

- Drill-in and drill-out operations consist of adding and removing a dimension to the classifier.

Example (drill-in):

Consider the query: ask for the number of sites where each blogger posts, classified by the blogger's age and city:

$$< c(x, y_1, y_2), m(x, z), count >$$

where the classifier and measure queries are defined by:

$$c(x, y_1, y_2) :- x\ age\ y_1, x\ livesIn\ y_2$$

$$m(x, z) :- x\ wrotePost\ y, y\ postedOn\ z$$

A roll up operation on the age dimension consists of removing the age dimension of the original classifier query:

$$Q =< c'_{\Sigma'}(x, y_2), m(x, z), count >$$

with:

$$\Sigma' = \{(y_2, \{y_2\})\} and c'(x, y_2) = x\ livesIn\ y_2$$

References and notes

- Dario Colazzo, François Goasdoué, Ioana Manolescu, Alexandra Roatis. RDF Analytics: Lenses over Semantic Graphs. 23rd International World Wide Web Conference, Apr 2014, Seoul, South Korea. 2014, <10.1145/2566486.2567982>.
- Dario Colazzo, François Goasdoué, Ioana Manolescu, Alexandra Roatis. Warehousing RDF Graphs. Bases de Donn´ees Avanc´ees, Oct 2013, Nantes, France. 2013.

[1] Analytical Schema (AnS)

DEFINITION 4. (ANALYTICAL SCHEMA) *An* analytical schema *(AnS) is a labeled directed graph $\mathcal{S} = \langle \mathcal{N}, \mathcal{E}, \lambda, \delta \rangle$ in which:*

- *$\mathcal{N}$ is the set of nodes;*
- *$\mathcal{E} \subseteq \mathcal{N} \times \mathcal{N}$ is the set of directed edges;*

each note/edge ==> URI

- *$\lambda : \mathcal{N} \cup \mathcal{E} \to U$ is an injective labeling function, mapping nodes and edges to URIs;*

each note => unary query

each edge => binary query

- *$\delta : \mathcal{N} \cup \mathcal{E} \to \mathcal{Q}$ is a function assigning to each node $n \in \mathcal{N}$ a unary BGP query $\delta(n) = q(x)$, and to every edge $e \in \mathcal{E}$ a binary BGP query $\delta(e) = q(x, y)$.*

Definition: Analytical Schema (AnS)

- Example of AnS - Graph

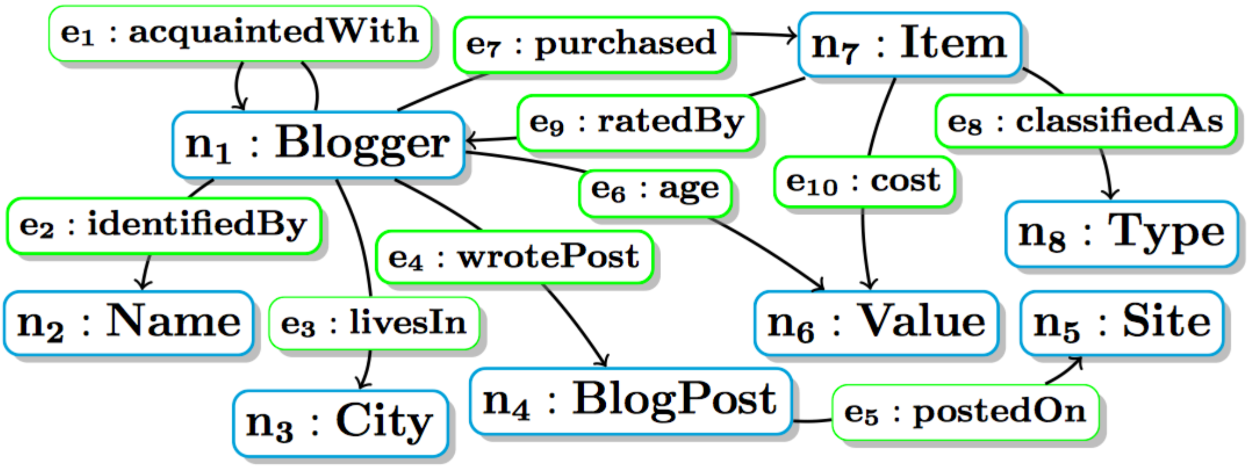


Figure: An example of AnS graph

- Example of AnS - **labels** and **queries** for the nodes and edges

node	$\lambda(n)$	$\delta(n)$
n_1	Blogger	$q(x)$:- x rdf:type Person, x wrote y, y inBlog z
n_2	Name	$q(x)$:- y hasName x
n_3	City	$q(x)$:- y inCity x
n_4	BlogPost	$q(x)$:- x rdf:type Message, x inBlog z, z rdf:type Blog
n_5	Site	$q(x)$:- y inBlog x, x rdf:type Blog
n_6	Value	$q(x)$:- z rdfs:range xsd:int, y z x
n_7	Item	$q(x)$:- x rdf:type y, y rdfs:subClassOf Product
n_8	Type	$q(x)$:- x rdfs:subClassOf Product

edge	$\lambda(e)$	$\delta(e)$
e_1	acquaintedWith	$q(x, y)$:- z rdfs:subPropertyOf knows, x z y
e_2	identifiedBy	$q(x, y)$:- x hasName y
e_3	livesIn	$q(x, y)$:- x hasCity y
e_4	wrotePost	$q(x, y)$:- x wrote y, y rdf:type Message
e_5	postedOn	$q(x, y)$:- x rdf:type Message, x inBlog y
e_6	age	$q(x, y)$:- x rdf:type Person, x hasAge y
e_7	purchased	$q(x, y)$:- x bought y
e_8	classifiedAs	$q(x, y)$:- x rdf:type Product, x rdf:type y
e_9	ratedBy	$q(x, y)$:- y gave z, z rdf:type Rating, z on x, x rdf:type Product
e_{10}	cost	$q(x, y)$:- x hasPrice y

Figure: labels and queries for the nodes and edges above

[2] Anlytical Schema Instance

DEFINITION 5. (INSTANCE OF AN AnS) *Let $\mathcal{S} = \langle \mathcal{N}, \mathcal{E}, \lambda, \delta \rangle$ be an analytical schema and* G *an RDF graph. The* instance of $\mathcal{S}$ w.r.t. G *is the RDF graph $\mathcal{I}(\mathcal{S}, \mathtt{G})$ defined as:*

$$\bigcup_{n \in \mathcal{N}} \{s \text{ rdf:type } \lambda(n) \mid s \in q(\mathtt{G}^{\infty}) \wedge q = \delta(n)\}$$
$$\cup$$
$$\bigcup_{e \in \mathcal{E}} \{s\ \lambda(e)\ o \mid s, o \in q(\mathtt{G}^{\infty}) \wedge q = \delta(e)\}.$$

Definition: Analytical Schema Instance

- Example of AnS instance

EXAMPLE 7. (ANALYTICAL SCHEMA INSTANCE) *Below we show part of the instance of the analytical schema introduced in Example* $\boxed{6}$. *We indicate at right of each triple the node (or edge) of the AnS which produced it.*

$$\mathcal{I}(\mathcal{S}, \mathtt{G}') = \begin{array}{ll}
\{\text{user}_1 \text{ rdf:type } \textbf{Blogger}, & n_1 \\
\ \text{user}_1 \ \textbf{acquaintedWith} \ \text{user}_2, & e_1 \\
\ \text{user}_1 \ \textbf{identifiedBy} \ \text{“Bill”}, & e_2 \\
\ \text{post}_1 \ \textbf{postedOn} \ \text{blog}_1, & e_5 \\
\ \text{user}_1 \ \textbf{age} \ \text{“28”}, & e_6 \\
\ \text{product}_1 \text{ rdf:type } \textbf{Item}, & n_7 \\
\ \text{SmartPhone rdf:type } \textbf{Type}, & n_8 \\
\ \text{product}_1 \ \textbf{cost} \ \text{“400”}, \ldots\} & e_{10}
\end{array}$$

Figure: an example of AnS instance

[3] Analytical query (AnQ)

DEFINITION 7. (ANALYTICAL QUERY) *Given an analytical schema* $\mathcal{S} = \langle \mathcal{N}, \mathcal{E}, \lambda, \delta \rangle$, *an* analytical query *(AnQ) rooted in the node* $r \in \mathcal{N}$ *is a triple:*

$$Q = \langle c(x, d_1, \ldots, d_n), m(x, v), \oplus \rangle$$

where:

- $c(x, d_1, \ldots, d_n)$ *is a query rooted in the node* r_c *of its graph* G_c, *with* $\lambda(r_c) = x$. *This query is called the* classifier *of* x *w.r.t. the* n *dimensions* $d_1, \ldots, d_n$.
- $m(x, v)$ *is a query rooted in the node* r_m *of its graph* G_m, *with* $\lambda(r_m) = x$. *This query is called the* measure *of* x.
- $\oplus$ *is a* function *computing a value (a literal) from an input set of values. This function is called the* aggregator *for the measure of* x *w.r.t. its classifier.*
- *For every homomorphism* h_c *from the classifier to* $\mathcal{S}$ *and every homomorphism* h_m *from the measure to* $\mathcal{S}$, $h_c(r_c) = h_m(r_m) = r$ *holds.*

Definition: Analytical Query

- Example of AnQ

EXAMPLE 8. (ANALYTICAL QUERY) *The query below asks for the number of sites where each blogger posts, classified by the blogger's age and city:*

$$\langle c(x, y_1, y_2), m(x, z), count \rangle$$

where the classifier and measure queries are defined by:

$$c(x, y_1, y_2)\text{:- } x \text{ age } y_1, x \text{ livesIn } y_2$$
$$m(x, z)\text{:- } x \text{ wrotePost } y, y \text{ postedOn } z$$

Figure: an example of AnQ

- AnQ answer

EXAMPLE 9. (ANALYTICAL QUERY ANSWER) *Consider the query in Example 8, over the AnS in Figure 4. Some triples from the instance of this analytical schema were shown in Example 7. The classifier query' answer set is:*

$$\{\langle \text{user}_1, 28, \text{“}Madrid\text{”}\rangle, \langle \text{user}_3, 35, \text{“}NY\text{”}\rangle\}$$

while that of the measure query is:

$$\{\langle \text{user}_1, \text{blog}_1\rangle, \langle \text{user}_1, \text{blog}_2\rangle, \langle \text{user}_2, \text{blog}_2\rangle, \langle \text{user}_3, \text{blog}_2\rangle\}$$

Aggregating the blogs among the classification dimensions leads to the AnQ answer:

$$\{\langle 28, \text{“}Madrid\text{”}, 2\rangle, \langle 35, \text{“}NY\text{”}, 1\rangle\}$$

Figure: an example of AnQ answering

[4] Extended AnQ

DEFINITION 10. (EXTENDED AnQ) *As in Definition 7, let $\mathcal{S}$ be an AnS, and $d_1, \ldots, d_n$ be a set of dimensions, each ranging over a non-empty finite set $V_{i, 1 \leq i \leq n}$. Let Σ be a total function over $\{d_1, \ldots, d_n\}$ associating to each d_i, either $\{d_i\}$ or a non-empty subset of V_i. An* extended analytical query *Q is defined by a triple:*

$$Q\text{:- } \langle c_\Sigma(x, d_1, \ldots, d_n), m(x, v), \oplus\rangle$$

where (as in Definition 7) c is a classifier and m a measure query over $\mathcal{S}$, $\oplus$ is an aggregation operator, and moreover:

$$c_\Sigma(x, d_1, \ldots, d_n) = \bigcup_{(\chi_1, \ldots, \chi_n) \in \Sigma(d_1) \times \ldots \times \Sigma(d_n)} c(x, \chi_1, \ldots, \chi_n)$$

Definition: extended AnQ

- **Remarks:**
    - $\sum$ is a total function that maps each d_i over $\{d_1, \ldots, d_n\}$ to $\{d_i\}$ or a non-empty subset of V_i
    - $C_\sum(x, d_1, \ldots, d_n)$ is the set of all possible classifiers by substituting each dimension variable d_i with a value in $\sum(d_i)$

- The total function $\sum$ is like a **filter-clause**, which restricts the classifier result
- Semantics of an extended AnQ: instead of picking tuples from c(I), pick tuples from $c_{\sum}(I)$
- An ordinary AnQ: an extended analytical query where $\sum$ only contains mapping pairs of the form $(d_i, \{d_i\})$