Towards Schema-independent Querying on Document Data Stores

H. BEN HAMADOU[1], F. GHOZZI[2], A. PENINOU[1], O. TESTE[1]

[1]IRIT , Univesité de Toulouse - France UT3, UT2J

[2]MIRACL, Université de Sfax - Tunisie ISIMS

hamdi.ben-hamadou@irit.fr

26-03-2018, DOLAP'18

Institut de Recherche en Informatique de Toulouse

Documement-oriented Database

- **Data format:** Semi-structured documents, JSON, BSON . . .
- **Data model:** Schema-less
- **Advantage:** Big data support, Scalability, Availability
- **Example:** MongoDB, CouchDB
- **Applications:** Web, IoT, social media . . .
- **Interrogation:** JDBC, Drivers, API, Command line . . .


```
{
    "firstName": "John",
    "lastName": "Smith",
    "age": 25,
    "address": {
        "streetAddress": "21 2nd Street",
        "city": "New York",
        "state": "NY",
        "postalCode": "10021"
    },
    "phoneNumber": [
        { "type": "home", "number": "212 555-1234" },
        { "type": "fax", "number": "646 555-4567" }
    ]
}
```

Modeling Multi-structured Data

Collection

$$C = \{d_1, \ \ldots, \ d_c\}$$

Document

$$d_i = (k_i, v_i)$$

- k_i is the document' identifier.
- $v_i = \{a_{i,1} : v_{i,1}, \ldots, a_{i,n} : v_{i,n_i}\}$ is the document' value.

Document Schema

$$s_i = \{p_1, \ldots, p_m\}$$

where p_i is a path leading to leaf node in document d_i.

Collection Schema

$$S = \bigcup_{i=1}^{\|C\|} s_i$$

Structural Heterogeneity

Document 1

```
{
  "_id": 1,
  "title":"Fast and furious",
  "year":2017,
  "language":"English"
}
```

Document 2

```
{
  "_id": 2,
  "title": "Titanic",
   "details":
    {
        "year":1997,
        "language":"English"
    }
}
```

Document 3

```
{
  "_id": 3,
  "title": "Despicable Me 3",
  "year":2017
}
```

Document 4

```
{
  "_id": 4,
  "title": "The Hobbit",
   "versions":
    [{
        "year":2012,
        "language":"English"
     },
     {
        "year":2013,
        "language":"French"
     }]
}
```

Query Operators

Kernel of Unary Operators

$$k = \{\pi, \sigma\}$$

Projection Operator

$$\pi_{(A)}(C_{in}) = C_{out}$$

The project operator reduces the initial schemas of documents to a finite subset of attributes A.

Selection Operator

$$\sigma_{(P)}(C_{in}) = C_{out}$$

The select operator retrieves only documents that match the selection condition P expressed in normal form ($Norm_p$).

Querying Multi-structured Data Problem

$$\pi_{(\text{"title"}, \text{"year"})}(C)$$

Document 1

```
{
  "_id": 1,
  "title":"Fast and furious",
  "year":2017,
  "language":"English"
}
```

Document 2

```
 "_id": 2,
 "title": "Titanic",
 "details":
   {
       "year":1997,
       "language":"English"
   }
```

Document 3

```
{
  "_id": 3,
  "title": "Despicable Me 3",
  "year":2017
}
```

Document 4

```
{
  "_id": 4,
  "title": "The Hobbit",
  "versions":
    [{
        "year":2012,
        "language":"English"
     },
     {
        "year":2013,
        "language":"French"
     }]
}
```

Querying Multi-structured Data Problem

$\pi_{(\text{"title"}, \text{"year"})}(C)$

Document 1

```
{
  "_id": 1,
  "title": "Fast and furious",
  "year":2017
  "language":"English"
}
```

Document 2

```
{
  "_id": 2,
  "title": "Titanic",
  "details":
    {
        "year":1997
        "language":"English"
    }
}
```

Document 3

```
{
  "_id": 3,
  "title": "Despicable Me 3",
  "year":2017
}
```

Document 4

```
{
  "_id": 4,
  "title": "The Hobbit"",
  "versions":
    [{
        "year":2012
        "language":"English"
     },
     {
        "year":2013
        "language":"French"
     }]
}
```


Introduction Querying Semi-structured Documents

Querying Multi-structured Data Problem

$\pi_{(\text{"title"}, \text{"year"}, \text{"details.year"}, \text{"versions.1.year"}, \text{"versions.2.year"})}(C)$

H. BEN HAMADOU et al. (IRIT) Schema-independent Querying 26-03-2018, DOLAP'18 6 / 28

Querying Multi-structured Data Problem

$$\pi_{(\text{“title”}, \text{“year”}, \text{“details.year”}, \text{“versions.1.year”}, \text{“versions.2.year”})}(C)$$

Document 1

```
{
  "_id": 1,
  "title":  "Fast and furious",
  "year":2017
  "language":"English"
}
```

Document 2

```
{
  "_id": 2,
  "title":  "Titanic",
  "details":
    {
        "year":1997
        "language":"English"
    }
}
```

Document 3

```
{
  "_id": 3,
  "title":  "Despicable Me 3",
  "year":2017
}
```

Document 4

```
{
  "_id": 4,
  "title":  "The Hobbit"",
  "versions":
    [{
        "year":2012
        "language":"English"
     },
     {
        "year":2013
        "language":"French"
     }]
}
```


Plan

Physical data transformation

- Flattening data.
- Using additional databases.
- Introducing new structures.

[$(Chasseuretal., 2013), (Taharaetal., 2014)(Taharaetal., 2014)$]
⇒ *Need to learn new schema.*
⇒ *Loss of initial document schemas/structures.*
⇒ *Need to re − build new schemas when structres are changed.*

Virtual data transformation

- Inferring existing schemas.
- Building an unified schema.
- Tracking different schemas versions.

[(Baazizi et al., 2017),(Ruiz et al., 2015),(Wang et al., 2015)]
⇒ *Need to learn new structures.*
⇒ *Querying is only limited to structural level.*
⇒ *Heterogeneity is manually managed to formulate application queries.*

EasyQ

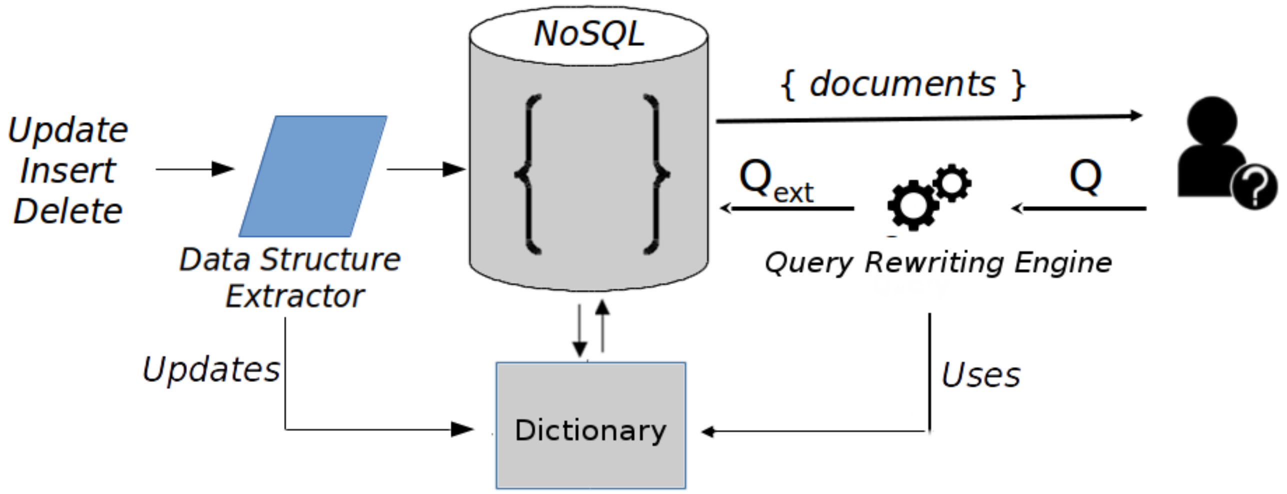


Figure: EasyQ Architecture

Dictionary

The dictionary $dict_C$ constructed from a collection C is defined by

$$dict_C = \{(p_k, \triangle_k)\} \; \forall p_k \in S_C$$

- $p_k \in S_C$ is a path leading to a leaf node which is present in at least one document;
- $\triangle_k = \{p_{p_k,1}, \ldots, p_{p_k,q}\} \subseteq S_C$, is a set of navigational paths leading to p_k;

Dictionary Construction Process

"year"

Document 1

```
{
 "_id": 1,
 "title": "Fast and furious",
  "year":2017,
  "language":"English"
}
```

Document 2

```
"_id": 2,
"title": "Titanic",
 "details":
   {
        "year":1997,
        "language":"English"
   }
```

Document 3

```
{
  "_id": 3,
  "title": "Despicable Me 3",
  "year":2017
}
```

Document 4

```
{
  "_id": 4,
  "title": "The Hobbit",
  "versions":
     [{
          "year":2012,
          "language":"English"
      },
      {
          "year":2013
          "language":"French"
      }]
}
```


Querying Heterogeneous Documents | Dictionary

Dictionary Construction Process

$$dict = \{\ (\text{"year"}, \{\text{"year"}, \text{"details.year"}, \text{"versions.1.year"}, \text{"versions.2.year"}\})\ \}$$

H. BEN HAMADOU et al. (IRIT) | Schema-independent Querying | 26-03-2018, DOLAP'18 | 12 / 28

Dictionary

```
dict = {
 ("title", {"title"} ),
 ( "year", {"year", "details.year", "versions.1.year", "versions.2.year"}),
 ( "language"", {"language", "details.language", "versions.1.language", "versions.2.language" }),
 ( "details", {"details"} ),
 ( "details.year", {"details.year" }),
 ( "details.language", {"details.language"}),
 ( "versions", {"versions" }),
 ( "versions.1", {"version.1" }) ,
 ( "versions.1.year", {"versions.1.year" }),
 ( "versions.1.language", {"versions.1.language" }),
 ( "versions.2", {"versions.2" }),
 ( "versions.2.year", {"versions.2.year" }),
 ( "versions.2.language", {"versions.2.language"} )
}
```


Algorithm for Automatic Query Extension

Algorithm 1: Automatic extension for the original user's query

Input: Q
Output: Q_{ext}
$Q_{ext} \leftarrow id$ // identity
foreach $q_i \in Q$ **do**
  switch q_i **do**
    case π_{A_i} // Projection operation
    do
      $A_{ext} \leftarrow \bigcup_{\forall a_k \in A_i} \triangle_k$; $Q_{ext} \leftarrow Q_{ext} \circ \pi_{A_{ext}}$
    end
    case σ_{Norm_p} // $Norm_p = \wedge_i(\vee_j \; a_{i,j} \; \varpi_{i,j} \; v_{i,j})$ selection operation
    do
      $P_{ext} \leftarrow \bigwedge_k \left(\bigvee_l \bigvee_{a_j \in \triangle_{k,l}} a_j \; \varpi_{k,l} \; v_{k,l} \right)$; $Q_{ext} \leftarrow Q_{ext} \circ \sigma_{P_{ext}}$
    end
  end
end

Extending project operator

Algorithm 1: Automatic extension for the original user's query

Input: Q
Output: Q_{ext}
$Q_{ext} \leftarrow id$ // identity
foreach $q_i \in Q$ **do**
  switch q_i **do**
    case π_{A_i} // Projection operation
    do
      $A_{ext} \leftarrow \bigcup_{\forall a_k \in A_i} \triangle_k$; $Q_{ext} \leftarrow Q_{ext} \circ \pi_{A_{ext}}$
    end
    case σ_{Norm_p} // $Norm_p = \wedge_i(\vee_j \; a_{i,j} \; \varpi_{i,j} \; v_{i,j})$ selection operation
    do
      $P_{ext} \leftarrow \bigwedge_k \left(\bigvee_l \bigvee_{a_j \in \triangle_{k,l}} a_j \; \varpi_{k,l} \; v_{k,l} \right)$; $Q_{ext} \leftarrow Q_{ext} \circ \sigma_{P_{ext}}$
    end
  end
end

Querying Heterogeneous Documents | Query Extension for Multi-structured Data

Extending project operator

Attributes extensions

$A_{ext} \leftarrow \bigcup_{\forall a_k \in A_i} \triangle_k$

Example

$$\pi_{(\text{“}title\text{”} \;,\; \text{“}year\text{”})}(C)$$

H. BEN HAMADOU et al. (IRIT) | Schema-independent Querying | 26-03-2018, DOLAP'18 | 15 / 28

Extending project operator

$$\pi_{(\text{"title"}, \text{"year"})}(C)$$

```
dict = {
 ("title", {"title"} ),
 ("year", {"year", "details.year", "versions.1.year", "versions.2.year"})
 ("language"}, {"language", "details.language", "versions.1.language", "versions.2.language"]
 ("details", {"details"} ),
 ( "details.year",{"details.year"}),
 ( "details.language", {"details.language"}),
 ("versions", {"versions"}),
 ( "versions.1", {"version.1"}) ,
 ( "versions.1.year",{"versions.1.year"} ),
 ( "versions.1.language", {"versions.1.language"} ),
 ( "versions.2", {"versions.2"} ),
 ( "versions.2.year", {"versions.2.year"}),
 ("versions.2.language", {"versions.2.language"} )
}
```


Querying Heterogeneous Documents | Query Extension for Multi-structured Data

Extending project operator

Attributes extensions

$$A_{ext} \leftarrow \bigcup_{\forall a_k \in A_i} \triangle_k$$

Example

$$\pi_{(\text{"title"}, \text{"year"})}(C)$$

- $A_{ext} \leftarrow \{\text{"title"}\} \bigcup \{\text{"year"}, \text{"details.year"}, \text{"versions.1.year"}, \text{"versions.2.year"}\}$

Projection query extended

$$\Rightarrow \pi_{(\text{"title"}, \text{"year"}, \text{"details.year"}, \text{"versions.1.year"}, \text{"versions.2.year"})}(C)$$

H. BEN HAMADOU et al. (IRIT) | Schema-independent Querying | 26-03-2018, DOLAP'18 | 15 / 28

Extending Select Operator

Algorithm 1: Automatic extension for the original user's query

Input: Q
Output: Q_{ext}
$Q_{ext} \leftarrow id$ // identity
foreach $q_i \in Q$ **do**
  switch q_i **do**
    case π_{A_i} // Projection operation
    do
      $A_{ext} \leftarrow \bigcup_{\forall a_k \in A_i} \triangle_k$; $Q_{ext} \leftarrow Q_{ext} \circ \pi_{A_{ext}}$
    end
    case σ_{Norm_p} // $Norm_p = \wedge_i(\vee_j \; a_{i,j} \; \varpi_{i,j} \; v_{i,j})$ selection operation
    do
      $P_{ext} \leftarrow \bigwedge_k \left(\bigvee_l \bigvee_{a_j \in \triangle_{k,l}} a_j \; \varpi_{k,l} \; v_{k,l} \right)$; $Q_{ext} \leftarrow Q_{ext} \circ \sigma_{P_{ext}}$
    end
  end
end

Querying Heterogeneous Documents | Query Extension for Multi-structured Data

Extending Select Operator

Attributes extensions

$$P_{ext} \leftarrow \bigwedge_k \left(\bigvee_l \bigvee_{a_j \in \triangle_{k,l}} a_j \; \varpi_{k,l} \; v_{k,l} \right)$$

Example

$$\sigma_{(\text{"title"} \neq Null \wedge \text{"language"} = \text{"English"})}(C)$$

H. BEN HAMADOU et al. (IRIT) | Schema-independent Querying | 26-03-2018, DOLAP'18 | 16 / 28

Extending Select Operator

Extending Selection's Predicates

$$P_{ext} \leftarrow \bigwedge_k \left(\bigvee_l \bigvee_{a_j \in \triangle_{k,l}} a_j \ \varpi_{k,l} \ v_{k,l} \right)$$

Example

$$\sigma_{(\text{"title"} \neq Null \wedge \text{"language"} = \text{"English"})}(C)$$

Selection query extended

$$P_{ext} \leftarrow (\bigvee_{a_j \in \triangle(\text{"title"})} \ a_j \neq Null) \wedge (\bigvee_{a_j \in \triangle(\text{"language"})} \ a_j = \text{"English"}) \Rightarrow \sigma_{(P_{ext})}(C)$$

Querying Heterogeneous Documents | Query Extension for Multi-structured Data

Extending Select Operator

$$\sigma_{(\bigvee_{a_j \in \triangle(\text{"title"})} \ a_j \neq Null) \wedge (\bigvee_{a_j \in \triangle(\text{"language"})} \ a_j = \text{"English"})}(C)$$

```
dict = {
 ("title", {"title"} ),
 ("year", {"year", "details.year", "versions.1.year", "versions.2.year"}),
 ("language", {"language", "details.language", "versions.1.language", "versions.2.language"}),
 ("details", {"details"} ),
 ( "details.year",{"details.year"}),
 ( "details.language", {"details.language"}),
 ("versions", {"versions"}),
 ( "versions.1", {"version.1"}) ,
 ( "versions.1.year",{"versions.1.year"} ),
 ( "versions.1.language", {"versions.1.language"} ),
 ( "versions.2", {"versions.2"} ),
 ( "versions.2.year", {"versions.2.year"}),
 ("versions.2.language", {"versions.2.language"} )
}
```

H. BEN HAMADOU et al. (IRIT) | Schema-independent Querying | 26-03-2018, DOLAP'18 | 16 / 28

Extending Select Operator

Selection query extended

$$\sigma_{(\bigvee_{a_j \in \triangle(\text{"title"})} a_j \neq Null) \wedge (\bigvee_{a_j \in \triangle(\text{"language"})} a_j = \text{"English"})}(C)$$

Rewritten Query

$$\sigma_{\Big((\text{"title"} \neq Null) \wedge (\text{"language"} = \text{"English"} \vee \text{"details.language"} = \text{"English"} \vee \text{"versions.1.language"} = \text{"English"} \vee \text{"versions.2.language"} = \text{"English"}\Big)}(C)$$

Plan

Synthetic dataset


```
{
    "_id" : "012cde54e24fa175a1ce7965d",
    "director_name" : "Bob Odenkirk",
    "gross" : 1954202,
    "movie_title" : "A Nightmare on Elm Street 4: The Dream Master ",
    "facenumber_in_poster" : 43,
    "plot_keywords" : "apartment|oven|stove|thanksgiving|thanksgiving dinner",
    "language" : "Mandarin",
    "title_year" : 1982,
    "aspect_ratio" : 4,
    "director_facebook_likes" : 177,
    "cast_total_facebook_likes" : 13716,
    "num_critic_for_reviews" : 478,
    "actor_2_name" : "George Sanders",
    "actor_1_facebook_likes" : 44000,
    "num_user_for_reviews" : 1416,
    "country" : "Philippines",
    "content_rating" : "TV-G",
    "actor_2_facebook_likes" : 162,
    "duration" : 201,
    "actor_3_name" : "Jason Bateman",
    "budget" : 26000000,
    "imdb_score" : 3.6,
    "actor_3_facebook_likes" : 794,
    "actor_1_name" : "Numan Acar",
    "movie_imdb_link" : "http://www.imdb.com/title/tt2622294/?ref_=fn_tt_tt_1"

}
```


Figure: Flat Document d1 Describing Movies from IMDB

Experiments | Experimental Protocol

Synthetic dataset


```
{ "_id" : "59cde11e25fa005a1ce7966d",
    "group_1B" : {
        "level0" : {
            "level1" : {"director_name" : "Bob Odenkirk",
                        "gross" : 1954202 }
        } },
     "group_2B" : {
        "level0" : {
            "level1" : {"director_facebook_likes" : 177,
                        "cast_total_facebook_likes" : 13716 }
        } },
     "group_3B" : {
        "level0" : {
            "level1" : {"movie_title" : "A Nightmare on Elm Street 4: The Dream Master ",
                        "facenumber_in_poster" : 43,
                        "plot_keywords" : "apartment|oven|stove|thanksgiving|thanksgiving dinner",
                        "language" : "Mandarin",
                        "title_year" : 1982,
                        "aspect_ratio" : 4 }
        } },
     "group_4B" : {
        "level0" : {
            "level1" : {"num_critic_for_reviews" : 478,
                        "actor_2_name" : "George Sanders",
                        "actor_1_facebook_likes" : 44000,
                        "num_user_for_reviews" : 1416,
                        "country" : "Philippines",
                        "content_rating" : "TV-G",
                        "actor_2_facebook_likes" : 162 }
        } },
    "group_5B" : {
        "level0" : {
            "level1" : { "duration" : 201,
                        "actor_3_name" : "Jason Bateman",
                        "budget" : 26000000,
                        "imdb_score" : 3.6 }
        } },
     "group_6B" : {
        "level0" : {
            "level1" : {"actor_3_facebook_likes" : 794,
                        "actor_1_name" : "Numan Acar",
                        "movie_imdb_link" : "http://www.imdb.com/title/tt2622294/?ref_=fn_tt_tt_1" }
        }}
}
```


Figure: Document D1 after structural heterogenetiy injection

Settings of the generated dataset

Setting	Value
# of schema	10
# of grouping objects per schema	{5,6,1,3,4,2,7,2,1,3}
Nesting levels per schema	{4,2,6,1,5,7,2,8,3,4}
Percentage of schema presence	10%
# of attributes per schema	Random
# of attributes per grouping objects	Random
Collection size	10 GB, 25 GB, 50 GB, 100 GB
Number of documents per collection	12 M, 30 M , 60 M, 120 M

Table: Settings of the generated dataset

Queries predicates

Predicate	Attribute	Type	Operator	Paths	Depths	selectivity
p1	DirectorName	String	Regex{^A}	8	{8,2,3,9,6,5,4,7}	0,06 %
p2	Gross	Int	$>$ 100 k	7	{7,8,2,3,9,6,4}	66 %
p3	Language	String	$=$ "English"	7	{7,8,3,9,6,5,4}	0,018%
p4	Imdb_score	Float	<4,7	8	{8,7,2,3,4,5,6,9}	29 %
p5	Duration	Int	$\leq$ 200	7	{7,8,2,3,6,5,4}	77%
p6	Country	String	$\neq$ *Null*	6	{7,2,3,9,5,4}	100 %
p7	year	Int	$<$ 1950	7	{7,8,2,3,6,5,4}	23 %
p8	FB_likes	Int	$\geq$ 500	7	{6,2,3,8,5,4,3}	83 %

Table: Query predicates

Queries

Q1/Q2

$\pi_{(*)}(\sigma_{(director\_name=``A\%"\ (\wedge/\vee) groos>100000)}(C))$

Q3/Q4

$\pi_{(*)}(\sigma_{(director\_name=``A\%"\ (\wedge/\vee) gross>100000 (\wedge/\vee) duration<200 (\wedge/\vee) title\_year<1950)}(C))$

Q5/Q6

$\pi_{(*)}(\sigma_{director\_name=``A\%"\ (\wedge/\vee) groos>100000 (\wedge/\vee) duration<200 (\wedge/\vee) title\_year<1950}$
$(\wedge/\vee) pays!=Null (\wedge/\vee) language=English (\wedge/\vee) imdb\_score<4 (\wedge/\vee)$
$cast\_total\_facebook\_likes>500 (C))$

Experiments | Experimental Protocol

Queries

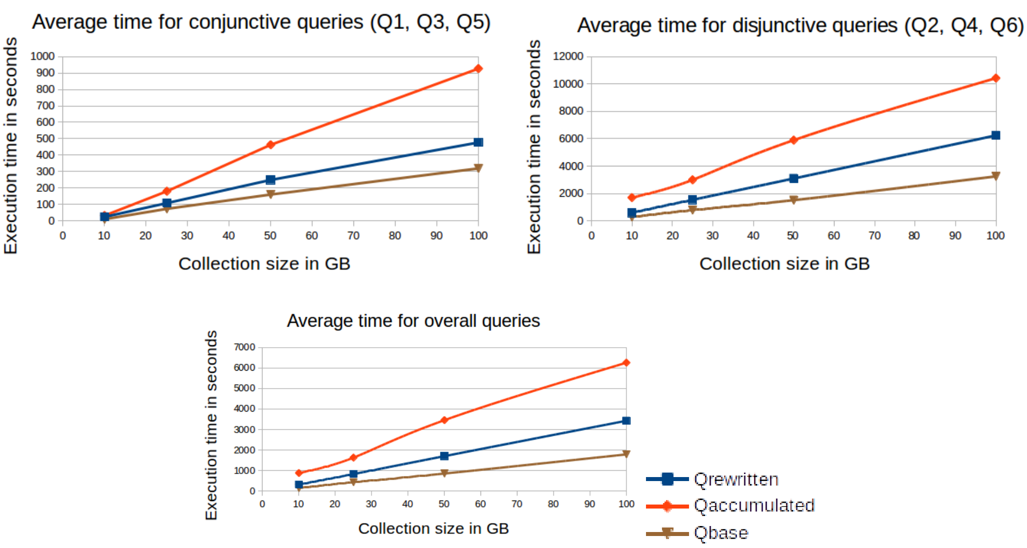
```
db.C.find(
    {'$and': [{'director_name': {'$regex': '^A'}} , {'gross': {'$gt': 100000}}]})
```

H. BEN HAMADOU et al. (IRIT) | Schema-independent Querying | 26-03-2018, DOLAP'18 | 21 / 28

Queries

```
db.C.find(
    {'$and': [
        {'$or': [
            {u'group_1C.level0.level1.level2.level3.level4.level5.director_name': {'$regex': '^A'}},
            {u'group_1F.level0.level1.level2.level3.level4.level5.level6.director_name': {'$regex': '^A'}},
            {u'group_1D.level0.director_name': {'$regex': '^A'}}, {u'group_1B.level0.level1.director_name': {'$regex': '^A'}},
            {u'group_1H.level0.level1.level2.level3.level4.level5.level6.level7.director_name': {'$regex': '^A'}},
            {u'group_1E.level0.level1.level2.level3.level4.director_name': {'$regex': '^A'}},
            {u'group_1J.level0.level1.level2.level3.director_name': {'$regex': '^A'}},
            {u'group_1I.level0.level1.level2.director_name': {'$regex': '^A'}}]},
        {'$or':
            [{u'group_1C.level0.level1.level2.level3.level4.level5.gross': {'$gt': 100000}},
            {u'group_1F.level0.level1.level2.level3.level4.level5.level6.gross': {'$gt': 100000}},
            {u'group_2D.level0.gross': {'$gt': 100000}},
            {u'group_1B.level0.level1.gross': {'$gt': 100000}},
            {u'group_2H.level0.level1.level2.level3.level4.level5.level6.level7.gross': {'$gt': 100000}},
            {u'group_3E.level0.level1.level2.level3.level4.gross': {'$gt': 100000}},
            {u'group_1I.level0.level1.level2.gross': {'$gt': 100000}}]}]})
```


Experimental Results



Data diversity effects on query rewriting time and dictionary size

# of schemas	Query rewriting in (s)	Dictionary size
10	0.0005	40 KB
100	0.0025	74 KB
1 K	0.139	2 MB
3 K	0.6	7.2 MB
5 K	1.52	12 MB

Dictionary online construction overhead

#of schemas	Load (s)	Load and dict. (s)	Overhead
2	201s	269s	33%
4	205s	277s	35%
6	207s	285s	37%
8	208s	300s	44%
10	210s	309s	47%

Table: Study of the overhead added during load time

Plan

Conclusion

EASYQ Advantages

- Overcoming the problem of querying documents with structural heterogeneity.
- Transparent rewriting mechanisms.
- Ensuring the coverage of latest structural changes. Therefore, the same query is rewritten at each execution

⇒ *The heterogeneity is automatically handled*.

Perspectives

- Employing real datasets
- Dealing with concurrent access
- Covering more operators


```
{''The_End'' :
      ''Thank you for your Kind Attention'' ,
 ''Next_?'' :
      ''It's Q&A Time '',
 ''Dataset'' :{
   ''Available_Online'':{
                              }
                     }
}
```