Work at PUC

A traditional Chilean drink made with pisco (a type of brandy), lemon juice, sugar, and egg white.

Freddy

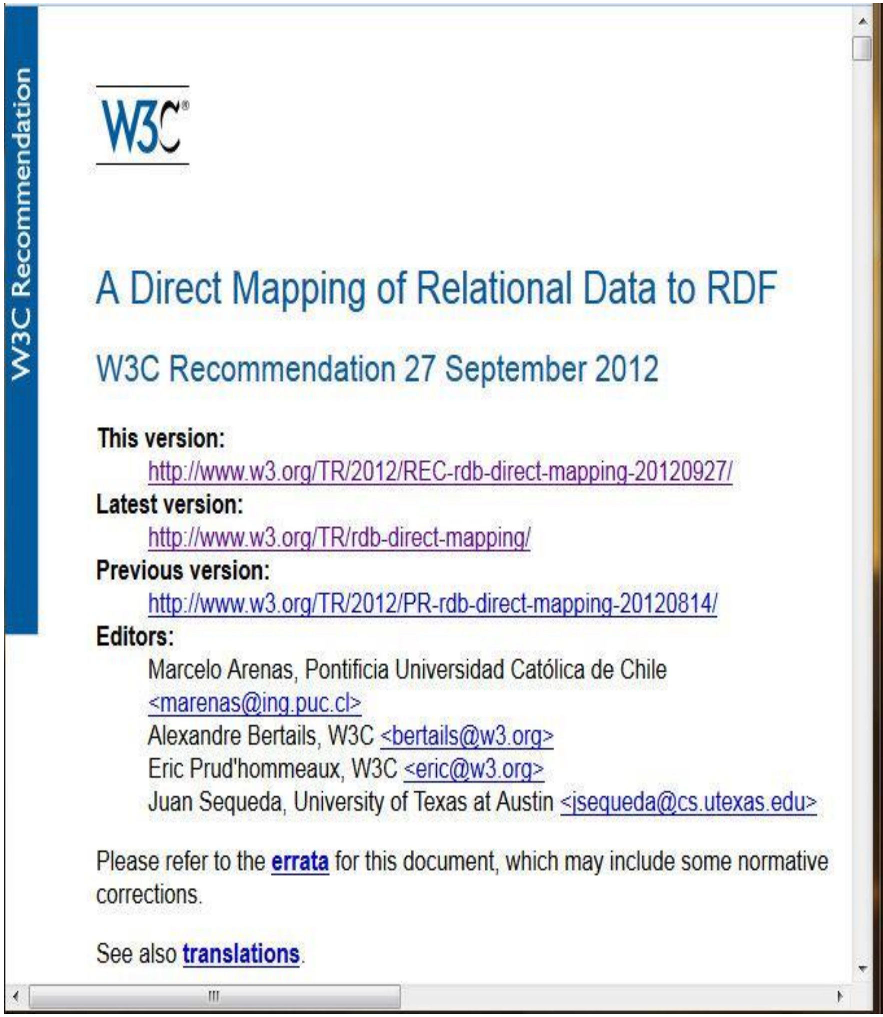
Oscr Corcho

https://www.google.com/maps/@-33.4482864,-70.6534626,3a,75y,257.04h,80.36t/data=!3m4!1e1!3m2!1sdq-IAOCo7g1DK5gpHJq_Iw!2e0?hl=es-419

The Title

- Option 1: On the Expressive Power of Direct Mapping and its Relationship with R2RML
- Option 2: Making morph-RDB out-of-business

A Tale of Two ~~Cities~~ Recommendations



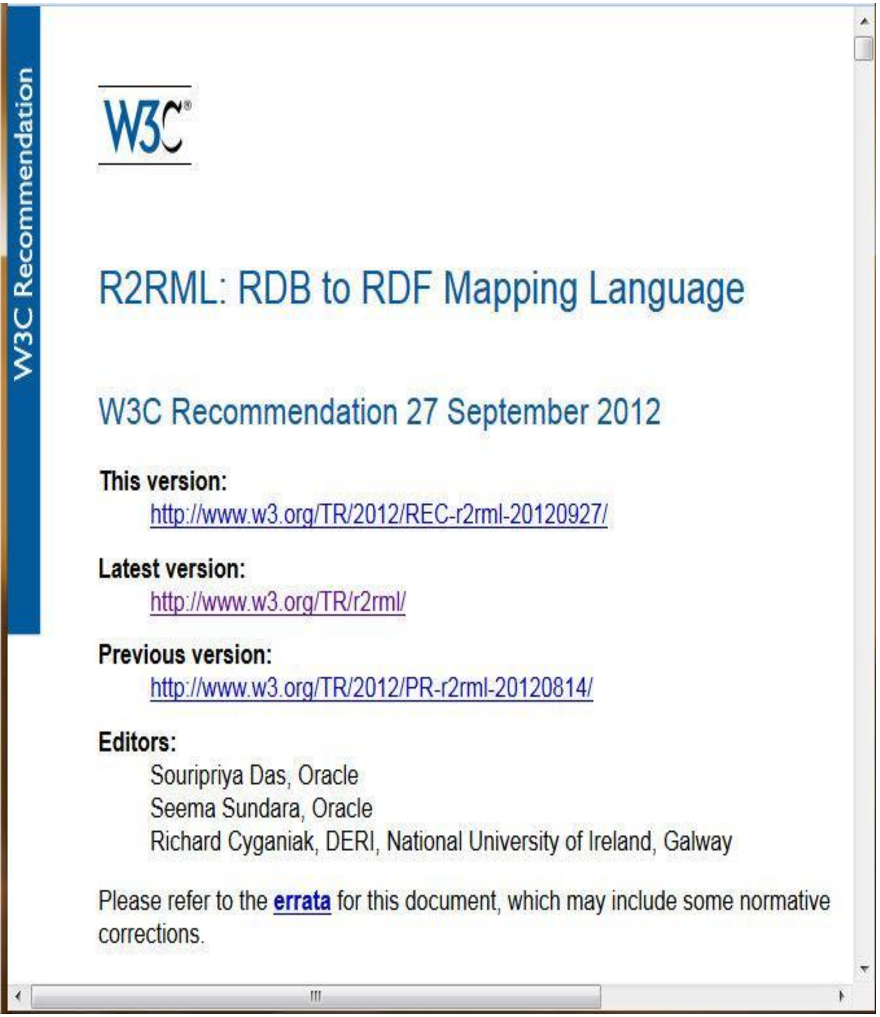
A Direct Mapping of Relational Data to RDF

W3C Recommendation 27 September 2012

This version:
http://www.w3.org/TR/2012/REC-rdb-direct-mapping-20120927/
Latest version:
http://www.w3.org/TR/rdb-direct-mapping/
Previous version:
http://www.w3.org/TR/2012/PR-rdb-direct-mapping-20120814/
Editors:
Marcelo Arenas, Pontificia Universidad Católica de Chile <marenas@ing.puc.cl>
Alexandre Bertails, W3C <bertails@w3.org>
Eric Prud'hommeaux, W3C <eric@w3.org>
Juan Sequeda, University of Texas at Austin <jsequeda@cs.utexas.edu>

Please refer to the **errata** for this document, which may include some normative corrections.

See also **translations**.



R2RML: RDB to RDF Mapping Language

W3C Recommendation 27 September 2012

This version:
http://www.w3.org/TR/2012/REC-r2rml-20120927/

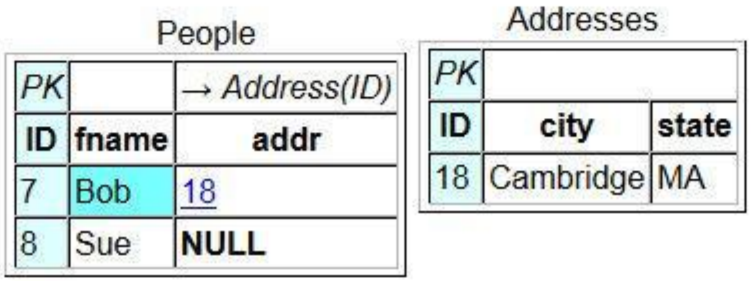
Latest version:
http://www.w3.org/TR/r2rml/

Previous version:
http://www.w3.org/TR/2012/PR-r2rml-20120814/

Editors:
Souripriya Das, Oracle
Seema Sundara, Oracle
Richard Cyganiak, DERI, National University of Ireland, Galway

Please refer to the **errata** for this document, which may include some normative corrections.

Direct Mapping

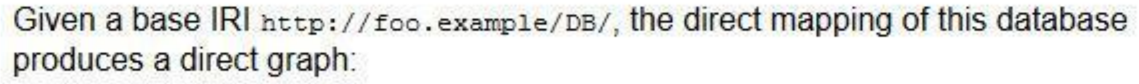


People

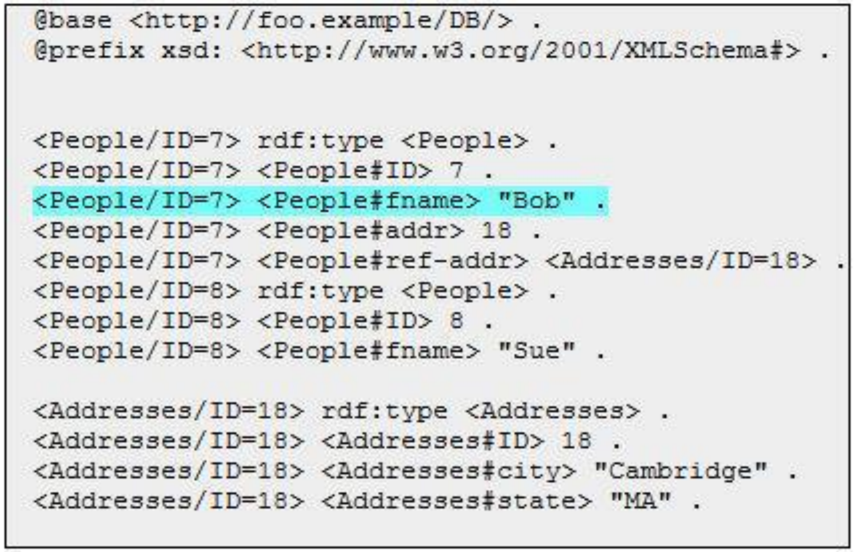
PK		→ Address(ID)
ID	**fname**	**addr**
7	Bob	18
8	Sue	**NULL**

Addresses

PK		
ID	**city**	**state**
18	Cambridge	MA



Given a base IRI `http://foo.example/DB/`, the direct mapping of this database produces a direct graph:

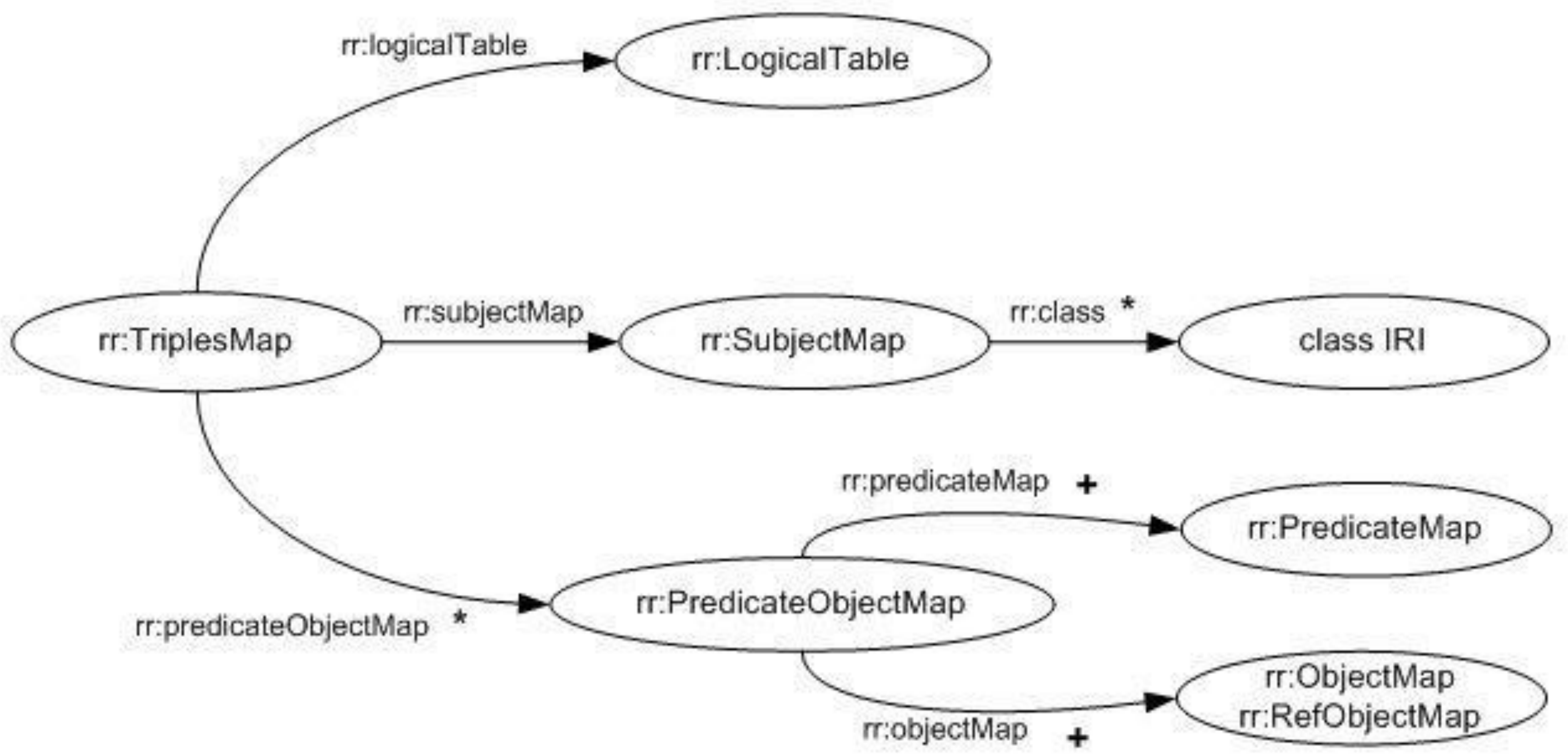


```
@base <http://foo.example/DB/> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .


<People/ID=7> rdf:type <People> .
<People/ID=7> <People#ID> 7 .
<People/ID=7> <People#fname> "Bob" .
<People/ID=7> <People#addr> 18 .
<People/ID=7> <People#ref-addr> <Addresses/ID=18> .
<People/ID=8> rdf:type <People> .
<People/ID=8> <People#ID> 8 .
<People/ID=8> <People#fname> "Sue" .

<Addresses/ID=18> rdf:type <Addresses> .
<Addresses/ID=18> <Addresses#ID> 18 .
<Addresses/ID=18> <Addresses#city> "Cambridge" .
<Addresses/ID=18> <Addresses#state> "MA" .
```

- One-click system
- Semantics in datalog (in the spec)
- Studied Properties (Sequeda et. al. WWW 2012)

http://www.w3.org/TR/rdb-direct-mapping/

R2RML



http://www.w3.org/TR/r2rml/

R2RML Example

DEPT

DEPTNO	DNAME	LOC
10	APPSERVER	NEW YORK

```
<TriplesMapDept>
  rr:logicalTable [ rr:tableName "DEPT" ];

  rr:subjectMap [
    rr:template “http://data.example.com/department/{DEPTNO}”;
    rr:class ex:Department; ];

  rr:predicateObjectMap [
    rr:predicateMap [ rr:constant ex:name;]
    rr:objectMap [ rr:column “DNAME”; ];

  rr:predicateObjectMap [
    rr:predicateMap [ rr:constant ex:location; ]
    rr:objectMap [ rr:column “LOC”; ]
]; .
```

```
<http://data.example.com/department/10> rdf:type ex:Department.
<http://data.example.com/department/10> ex:name "APPSERVER".
<http://data.example.com/department/10> ex:location "NEW YORK".
```

R2RML Example

EMP

EMPNO	ENAME	JOB	DEPTID
7369	SMITH	CLERK	10

```
<TriplesMapEmp>
  rr:logicalTable [ rr:tableName "EMP" ];

  rr:subjectMap [
    rr:template “http://data.example.com/employee/{EMPNO}”;
    rr:class ex:Employee; ];

  rr:predicateObjectMap [
    rr:predicateMap [ rr:constant ex:name; ]
    rr:objectMap [ rr:column “ENAME”; ];

  rr:predicateObjectMap [
    rr:predicateMap [ rr:constant ex:department; ]
    rr:objectMap [
      rr:parentTriplesMap <#TriplesMapDept>;
      rr:joinCondition [
        rr:child "DEPTID";  rr:parent "DEPTNO";
      ];
    ]
  ];.
```

```
<http://data.example.com/employee/7369> rdf:type ex:Employee.
<http://data.example.com/employee/7369> ex:name "SMITH".
<http://data.example.com/employee/7369> ex:department <http://data.example.com/department/10>.
```

What is missing?

- Formal study of the relationship between R2RML <-> DM:
  - Expressive power
    - DM -> R2RML (obvious, next slide)
    - R2RML -> DM (?)
  - Fundamental properties
    - Information Preservation
    - Query Result Preservation

Information Preservation: Example

PERSON

PERSONID	**LASTNAME**
7369	Smith

A Direct Mapping of Relational Data to RDF

EMP

PK	
EID	**LNAME**
7369	Smith

```
<TMEmp> a rr:TriplesMap;
  rr:logicalTable [ rr:tableName “EMP” ];

  rr:subjectMap [
    rr:template “:PERSON/PERSONID={EID}.LASTNAME={LNAME}”;
    rr:class PERSON; rr:termType rr:blankNode;
  ];

  rr:predicateObjectMap [
    rr:predicateMap [ rr:constant PERSON#PERSONID];
    rr:objectMap [ rr:column "EID"];
  ];

  rr:predicateObjectMap [
    rr:predicateMap [ rr:constant PERSON#LASTNAME];
    rr:objectMap [ rr:column "LNAME"];
  ]
].
```

R2RML: RDB to RDF Mapping Language

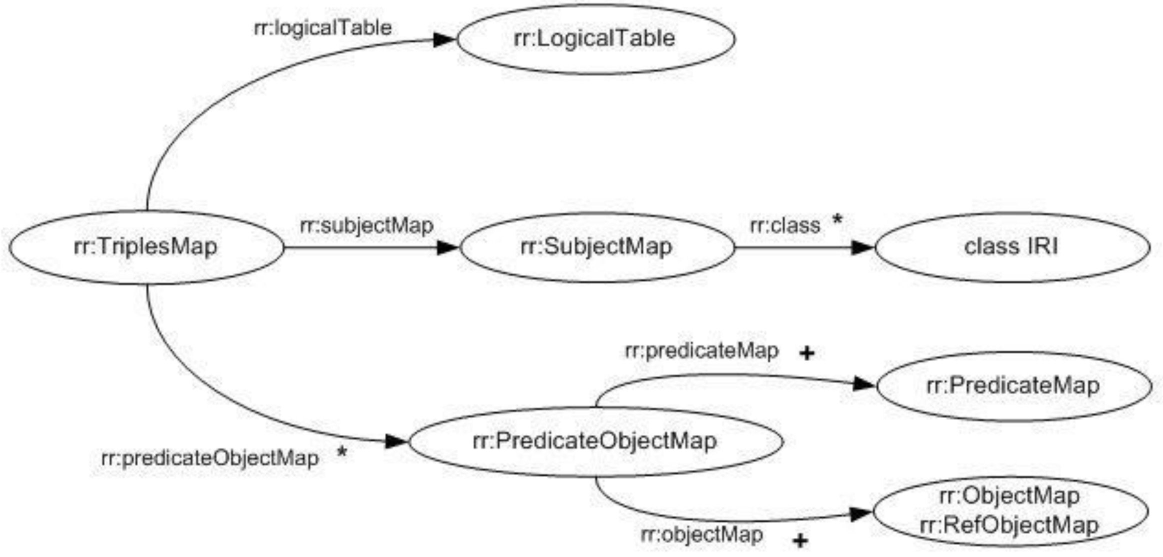
```
:PERSON/PERSONID=7369.LASTNAME=Smith rdf:type <PERSON> .
:PERSON/PERSONID=7369.LASTNAME=Smith <PERSON#PERSONID> "7369" .
:PERSON/PERSONID=7369.LASTNAME=Smith <PERSON#LASTNAME> "Smith" .
```

Preliminaries

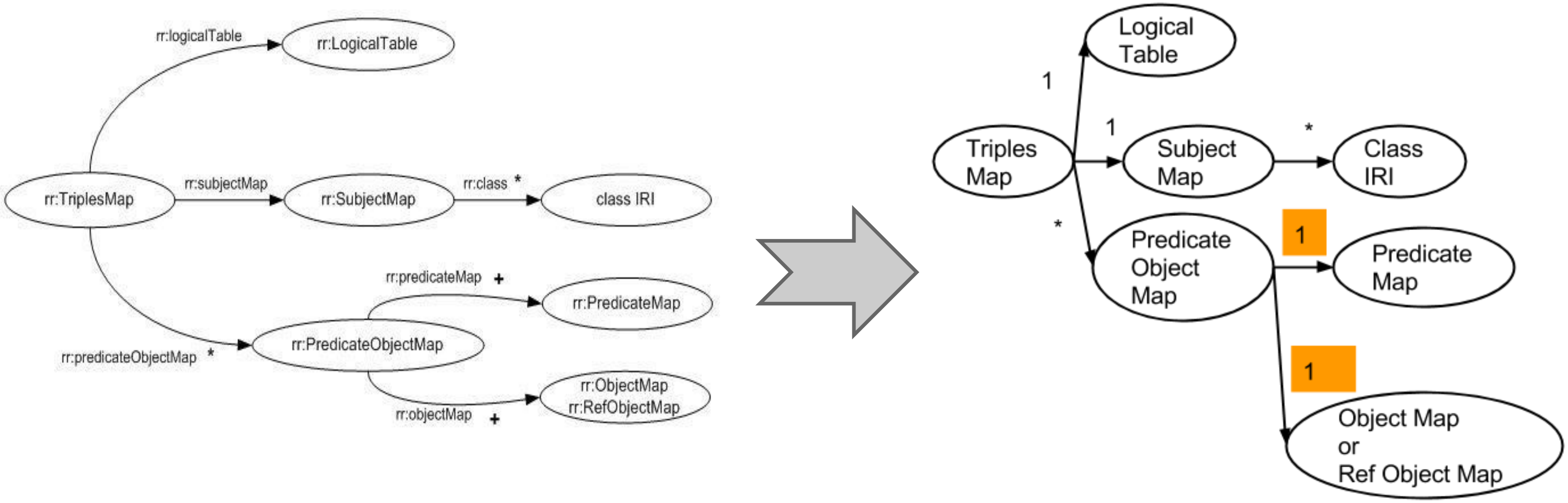
- Database schema: *R*
- Database instance: *I*
- SQL query: *Q*
- View Definition $V_Q^{\mathbf{R}}$ is a query *Q* over *R*
- View V_Q^I is the result of query evaluation *Q* over *I*
- Direct Mapping: *DM*
- R2RML Mapping Document: *M*
- *M* evaluated over *I*: $[[M]]_I$
- *DM* evaluated over *I*: $[[DM]]_I$
- Substitution: σ
- SPARQL Query: P, P_1, P_2

Preliminaries
R2RML$_{\text{Lite}}$

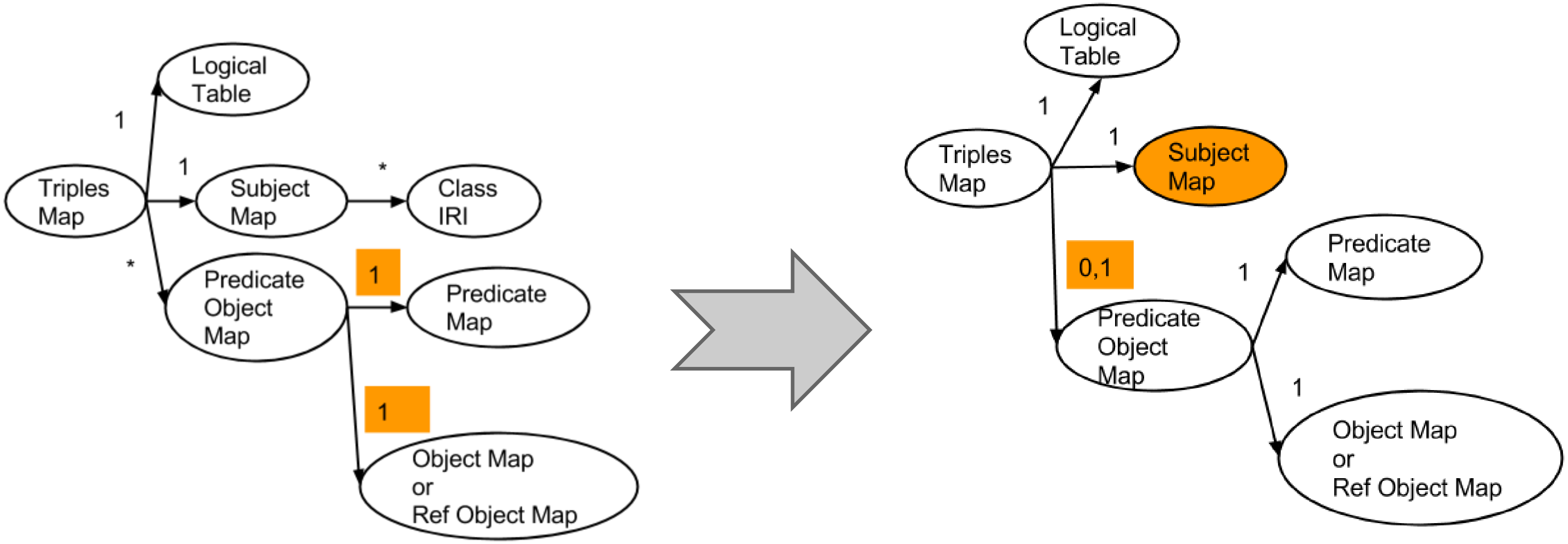
- no termtype (blanknode, literal,IRI)
- no datatype (xsd:dateTime, xsd:positiveInteger, etc)
- no language tag (english, spanish, etc)



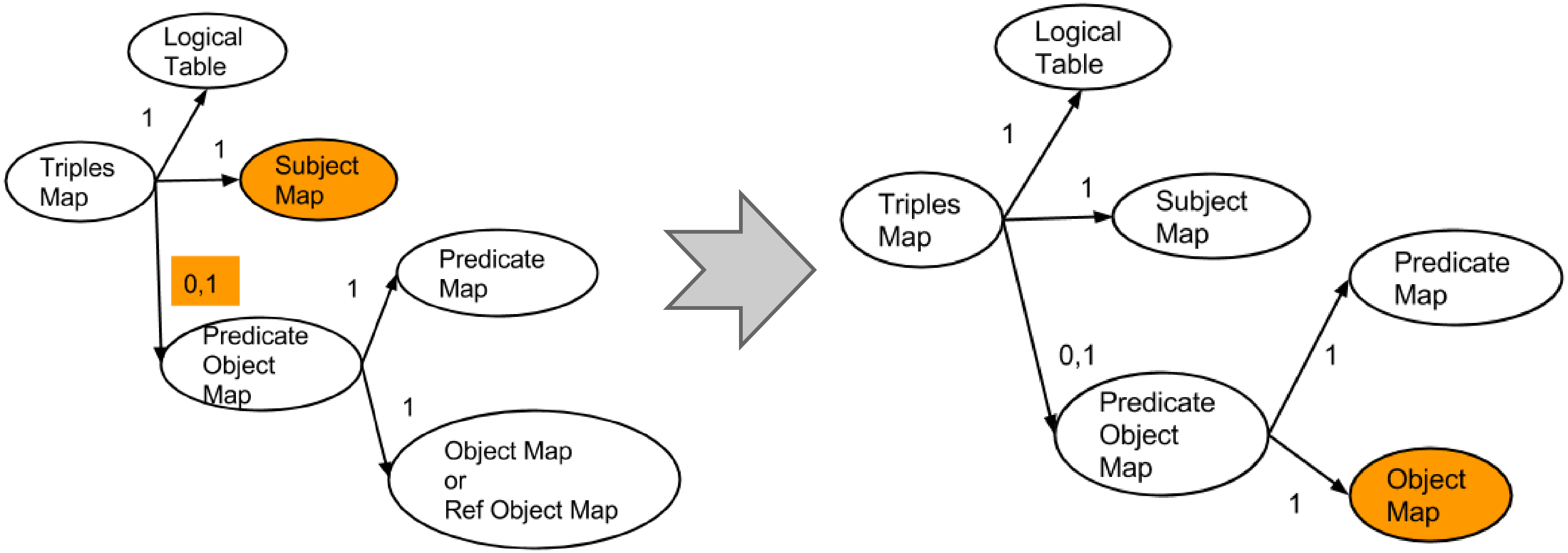
Preliminaries
R2RML Mappings 1st Normal Form



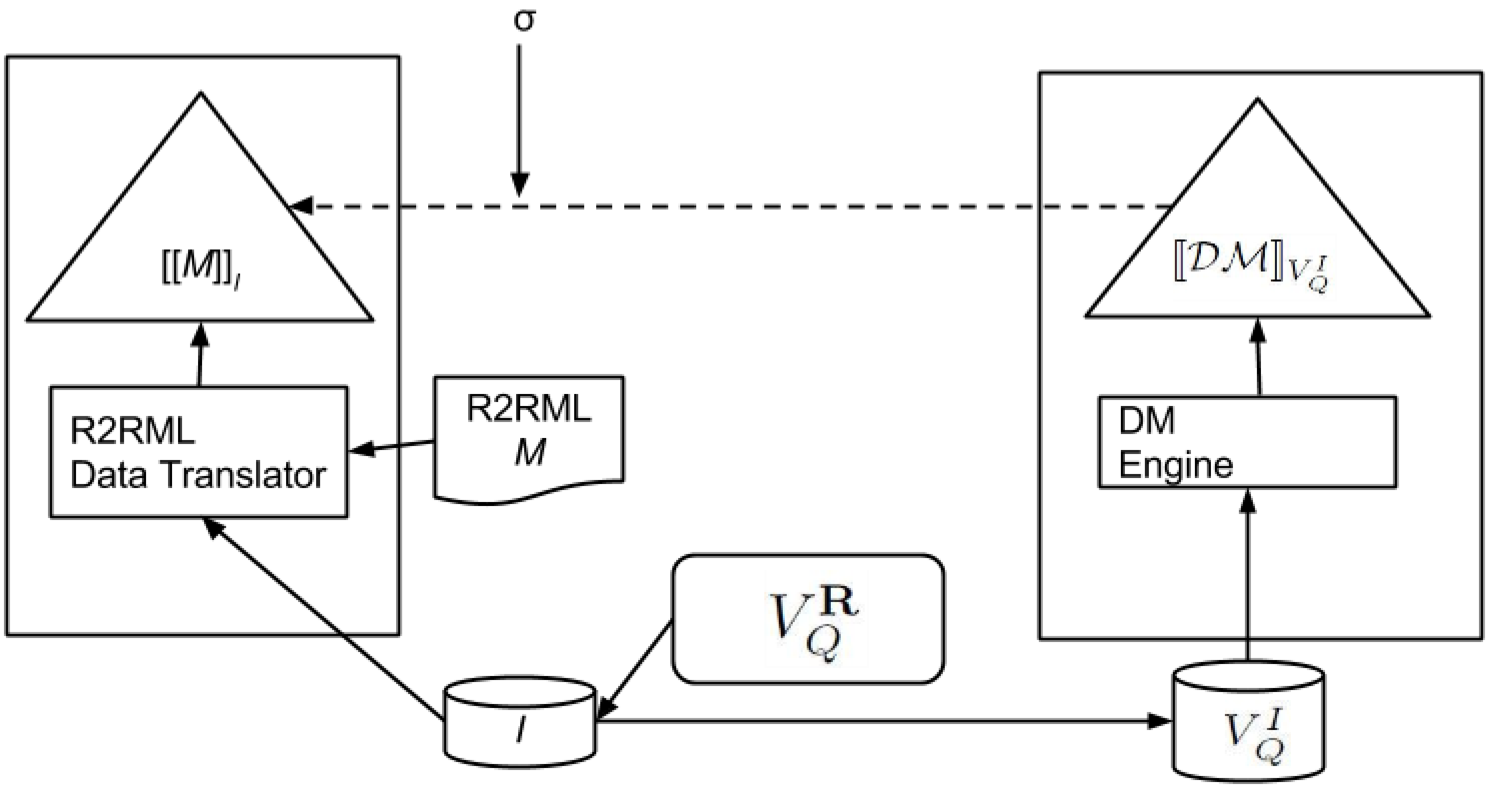
Preliminaries
R2RML Mappings 2nd Normal Form

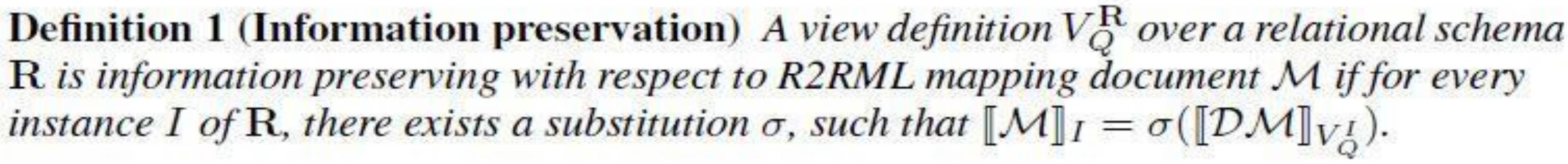


Preliminaries
R2RML Mappings 3rd Normal Form



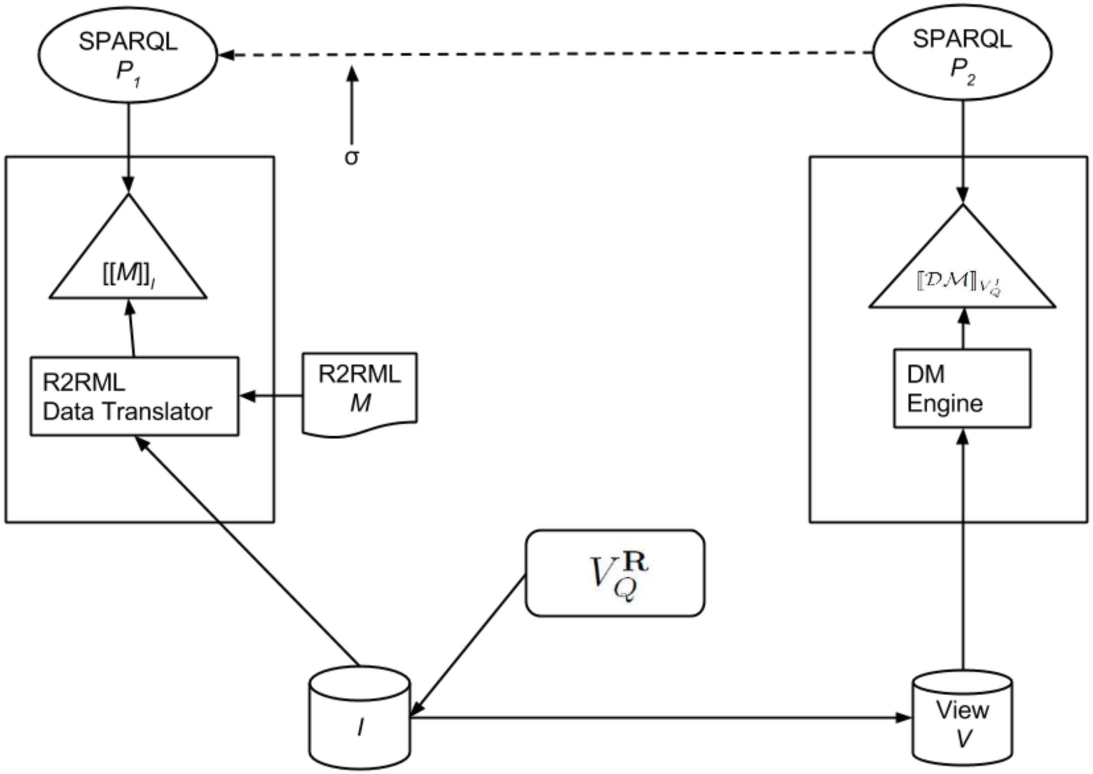
Information Preservation (IP) of View Definitions

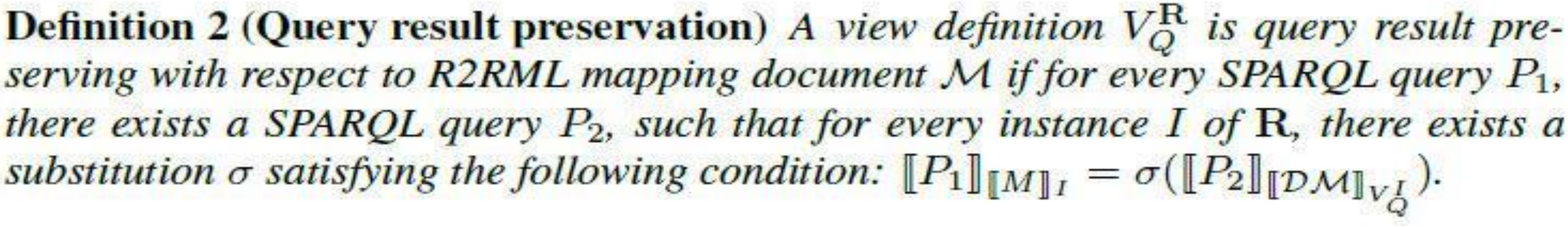




Definition 1 (Information preservation) *A view definition $V_Q^{\mathbf{R}}$ over a relational schema $\mathbf{R}$ is information preserving with respect to R2RML mapping document $\mathcal{M}$ if for every instance I of $\mathbf{R}$, there exists a substitution σ, such that $[\![\mathcal{M}]\!]_I = \sigma([\![\mathcal{DM}]\!]_{V_Q^I})$.*

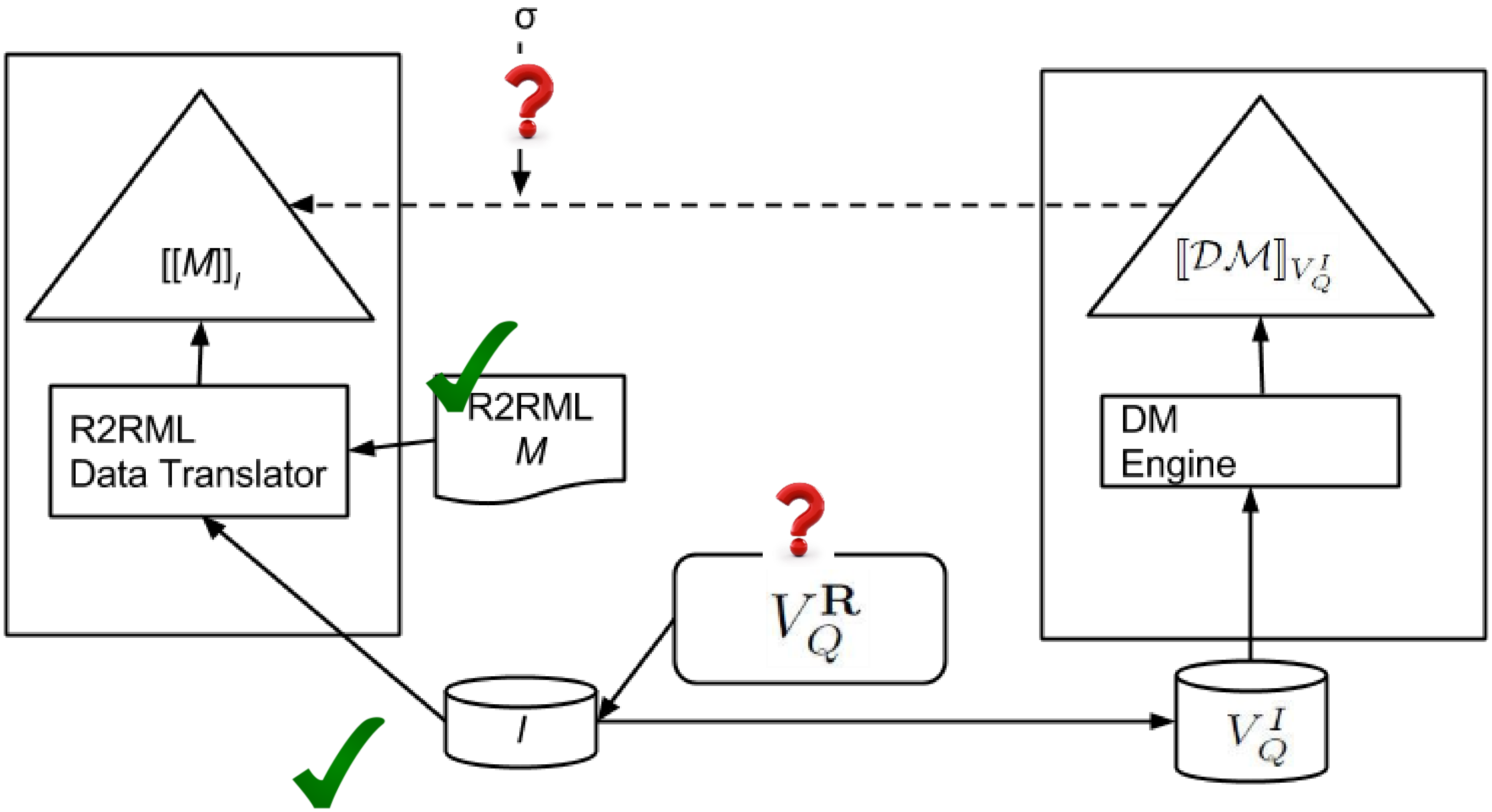
Query Result Preservation (QRP) of View Definitions





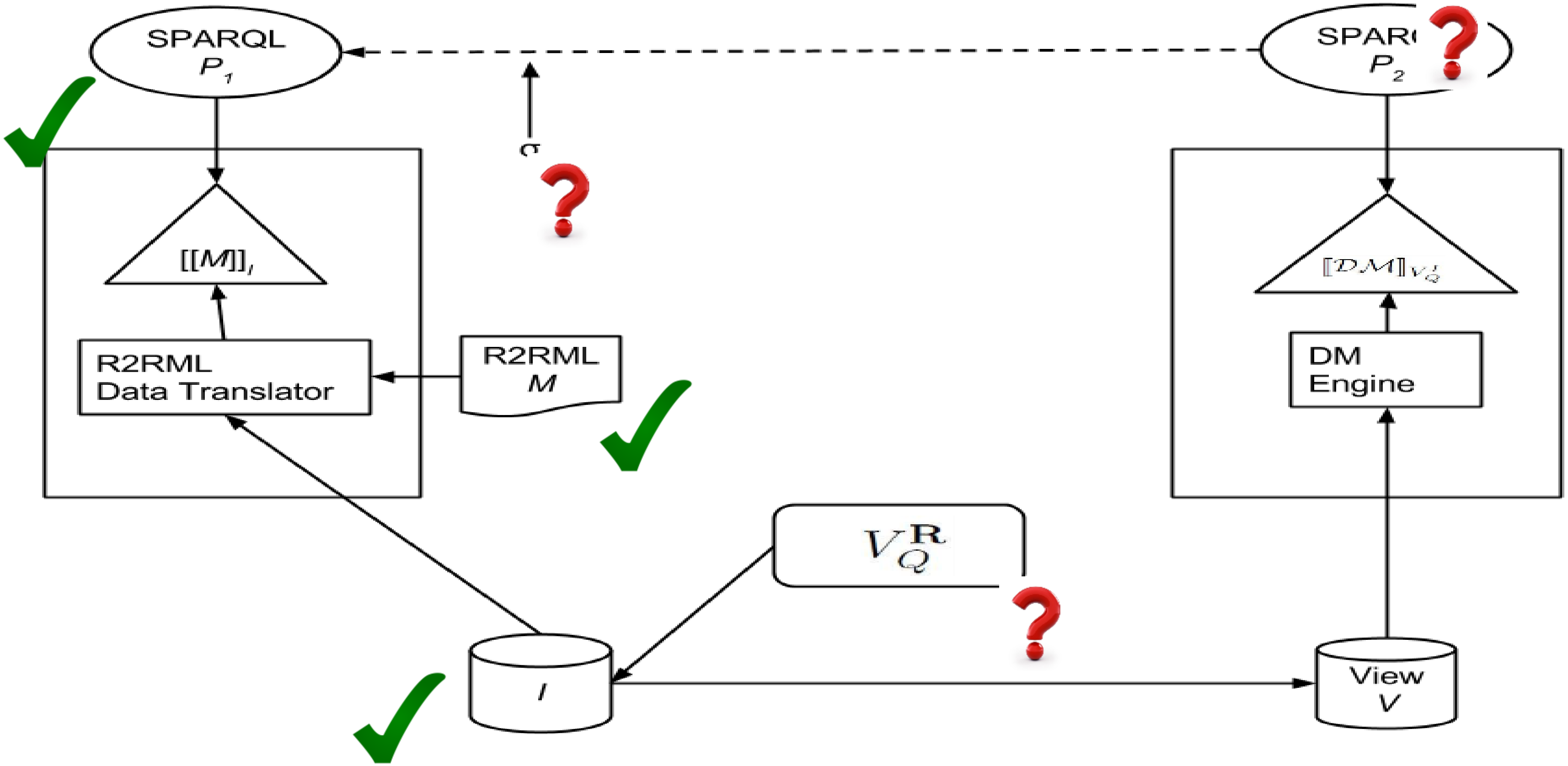
Definition 2 (Query result preservation) *A view definition $V_Q^{\mathbf{R}}$ is query result preserving with respect to R2RML mapping document $\mathcal{M}$ if for every SPARQL query P_1, there exists a SPARQL query P_2, such that for every instance I of $\mathbf{R}$, there exists a substitution σ satisfying the following condition: $[\![P_1]\!]_{[\![M]\!]_I} = \sigma([\![P_2]\!]_{[\![\mathcal{DM}]\!]_{V_Q^I}})$.*

IP Problem (IPP)



Given a relational schema *R* and R2RML mapping document *M*, find substitution σ and query $V_Q^{\mathbf{R}}$ that satisfy the Information Preservation Property

QRP Problem



Given a relational schema *R*, an R2RML mapping document *M*. and a query P_1, find substitution σ , view definition $V_Q^{\mathbf{R}}$, and query P_2 that satisfy the Query Result Preservation Property

Running Example

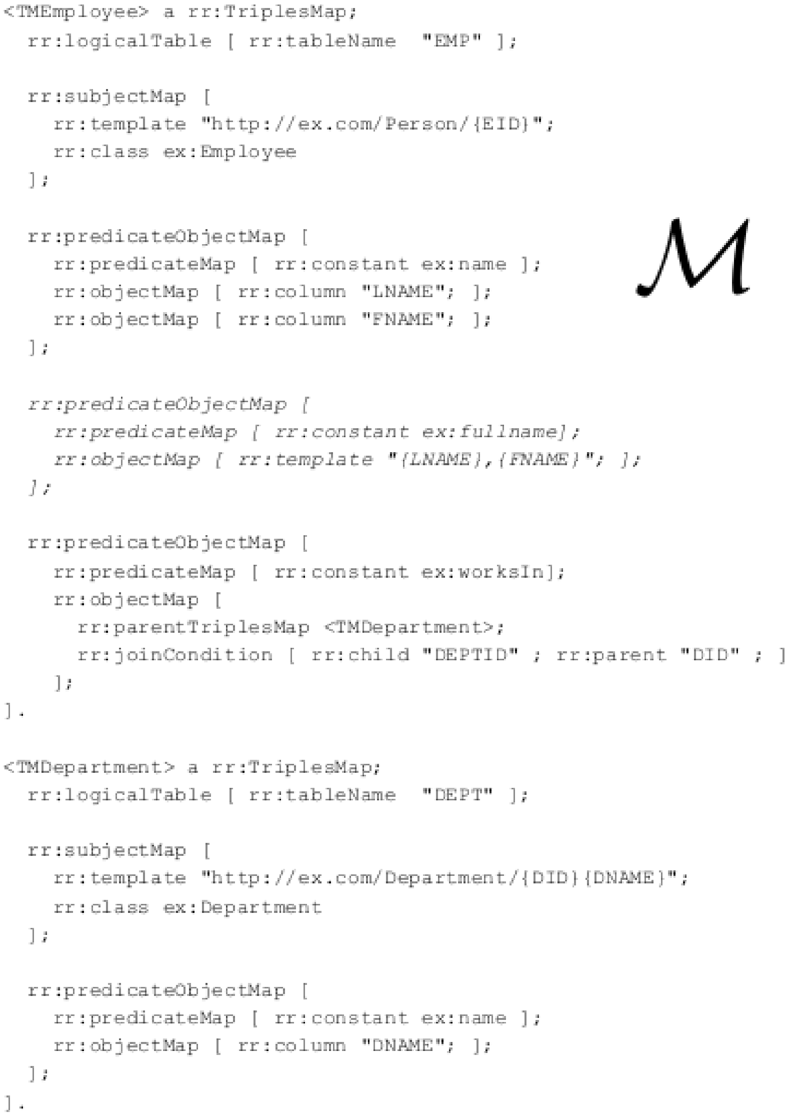
EMP

PK				-> DEPT (DID)
EID	**LNAME**	**FNAME**	**JOB**	**DEPTID**
7369	Smith	Adam	Clerk	10

DEPT

PK		
DID	**DNAME**	**LOC**
10	AppServer	New York

I



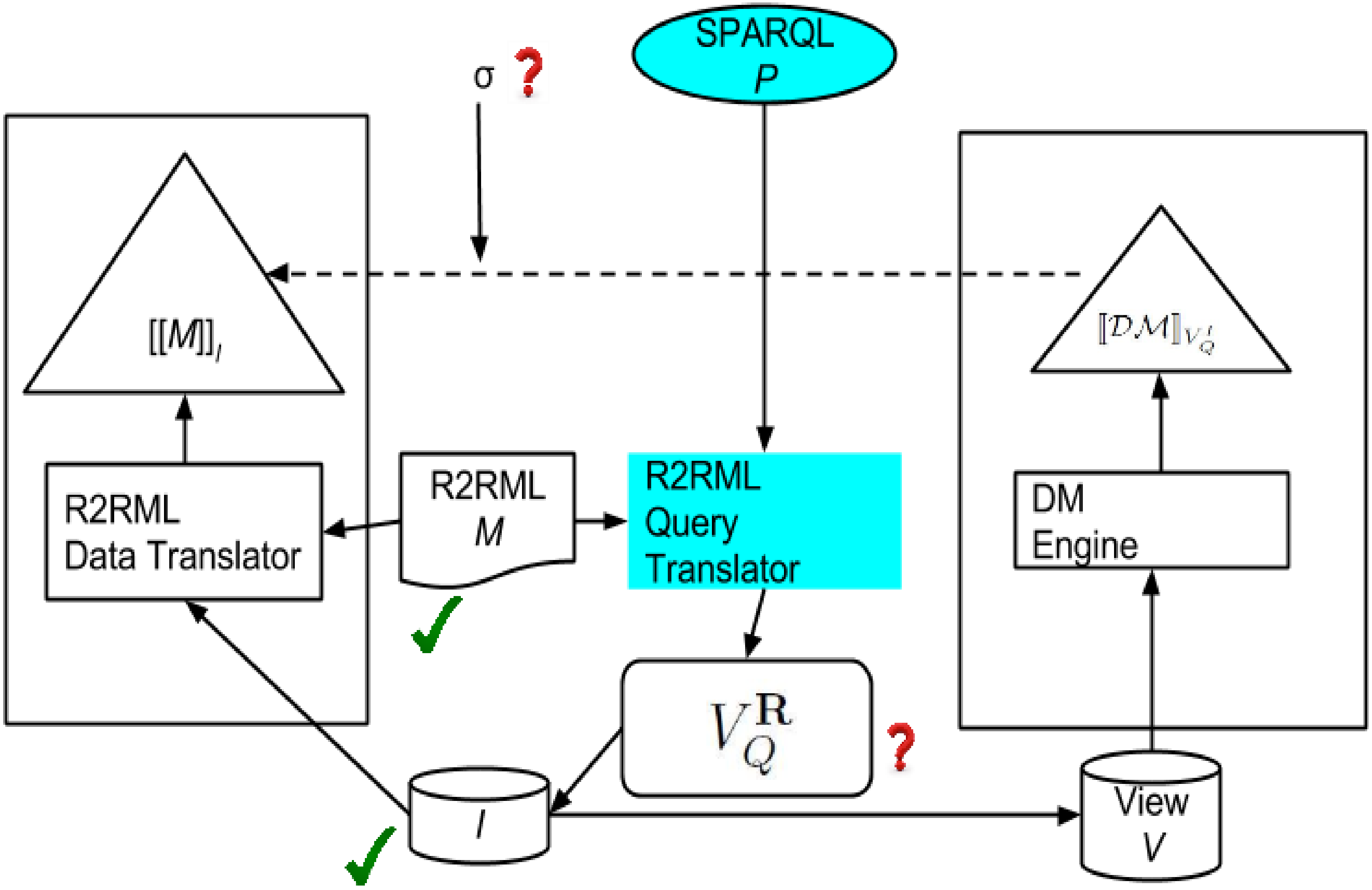
```
<TMEmployee> a rr:TriplesMap;
  rr:logicalTable [ rr:tableName  "EMP" ];

  rr:subjectMap [
    rr:template "http://ex.com/Person/{EID}";
    rr:class ex:Employee
  ];

  rr:predicateObjectMap [
    rr:predicateMap [ rr:constant ex:name ];
    rr:objectMap [ rr:column "LNAME"; ];
    rr:objectMap [ rr:column "FNAME"; ];
  ];

  rr:predicateObjectMap [
    rr:predicateMap [ rr:constant ex:fullname];
    rr:objectMap [ rr:template "{LNAME},{FNAME}"; ];
  ];

  rr:predicateObjectMap [
    rr:predicateMap [ rr:constant ex:worksIn];
    rr:objectMap [
      rr:parentTriplesMap <TMDepartment>;
      rr:joinCondition [ rr:child "DEPTID" ; rr:parent "DID" ; ]
    ];
].

<TMDepartment> a rr:TriplesMap;
  rr:logicalTable [ rr:tableName  "DEPT" ];

  rr:subjectMap [
    rr:template "http://ex.com/Department/{DID}{DNAME}";
    rr:class ex:Department
  ];

  rr:predicateObjectMap [
    rr:predicateMap [ rr:constant ex:name ];
    rr:objectMap [ rr:column "DNAME"; ];
  ];
].
```

$\mathcal{M}$

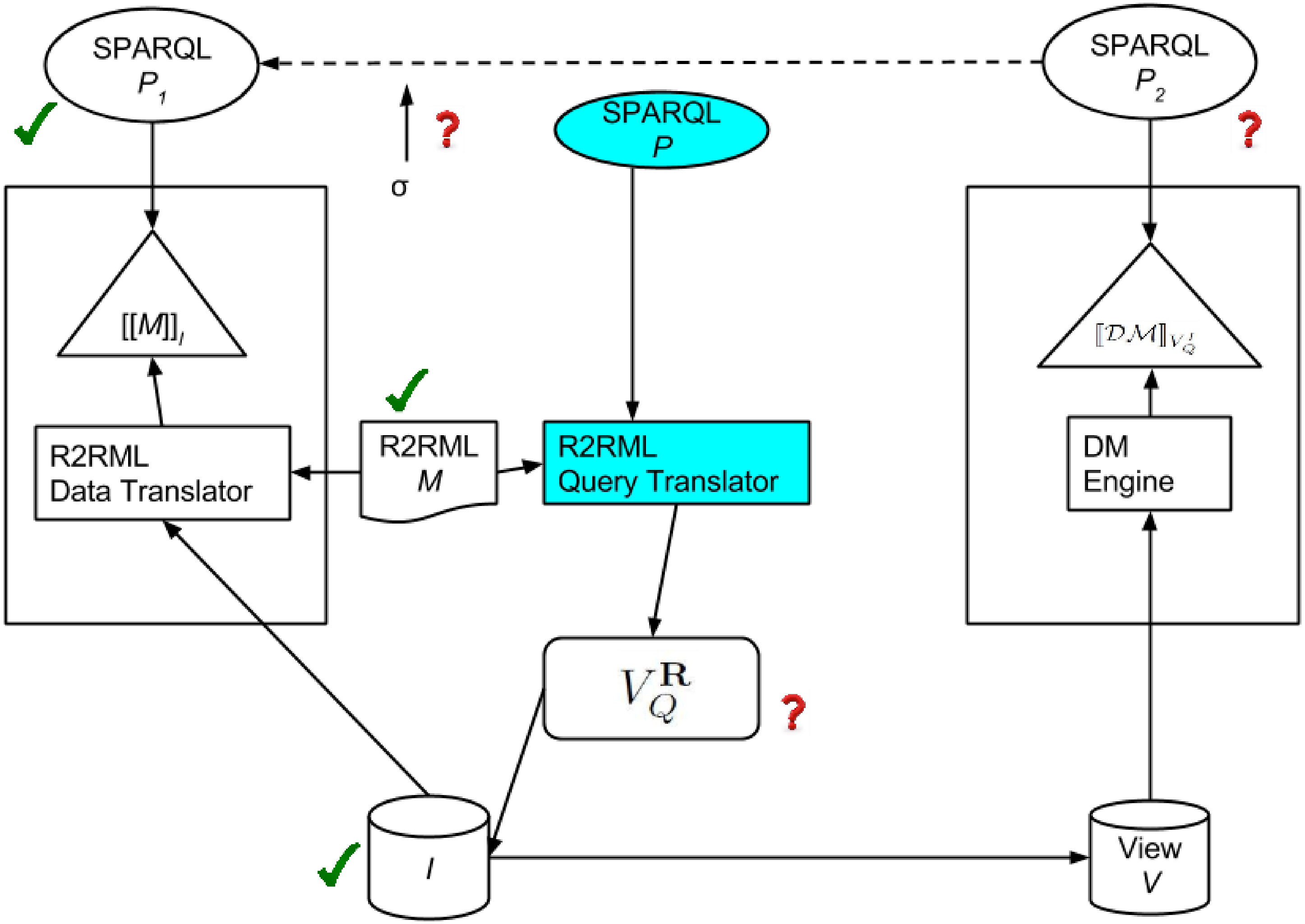
$$[\![\mathcal{M}]\!]_I$$

<http://ex.com/Person/7369> rdf:type ex:Employee.
<http://ex.com/Person/7369> ex:name "SMITH".
<http://ex.com/Person/7369> ex:name "ADAM".
<http://ex.com/Person/7369> ex:fullname "SMITH,ADAM".
<http://ex.com/Person/7369> ex:worksIn <http://ex.com/Department/10APPSERVER>.
<http://ex.com/Department/10APPSERVER> rdf:type ex:Department.
<http://ex.com/Department/10APPSERVER> ex:name "APPSERVER".

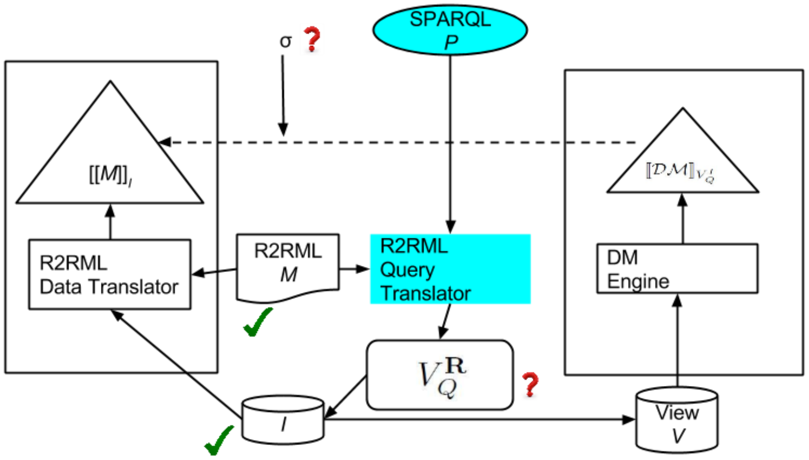
Our Approach for Solving the IP Problem



Our Approach for Solving QRP Problem

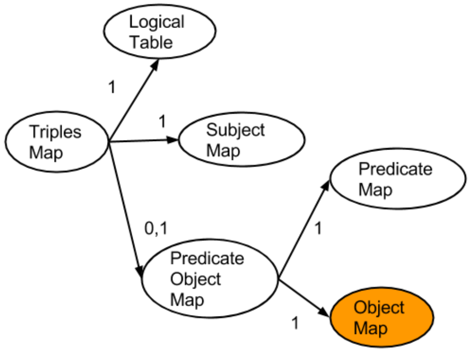


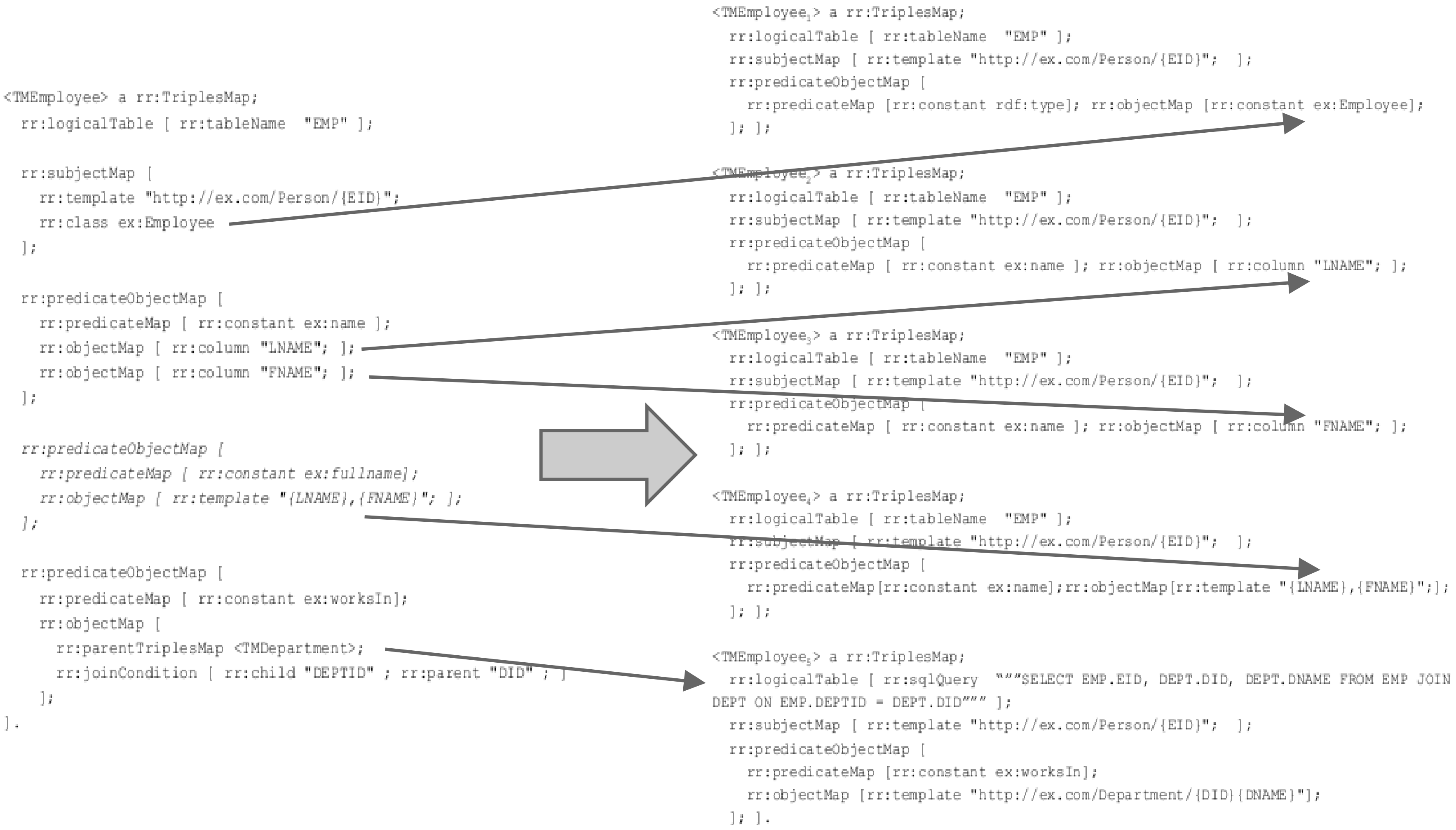
IP Problem
Vertical Approach



1. Normalize mappings in M_{Lite} form to M where M is in 3rd NF
2. Set Q = *trans(P, M)*
   - P = `SELECT ?s AS ?rdf:subject ?p AS ?rdf:predicate ?o AS ?rdf:object WHERE { ?s ?p ?o }.`
3. Set $V_Q^{\mathbf{R}}$ = `CREATE VIEW rdf:statement AS` `Q;`
4. Set σ =
   - Replace `rdf:Statement#rdf:subject` with `rdf:subject`
   - Replace `rdf:Statement#rdf:predicate` with `rdf:predicate`
   - Replace `rdf:Statement#rdf:object` with `rdf:object`

IP Problem
Approach - Step 1
Normalize Mappings

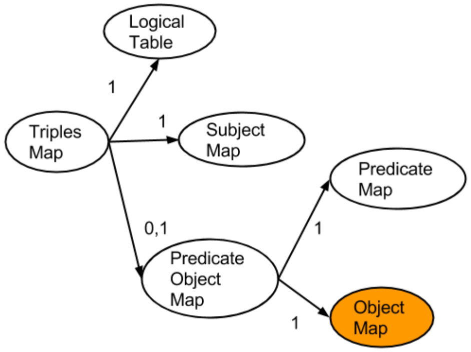


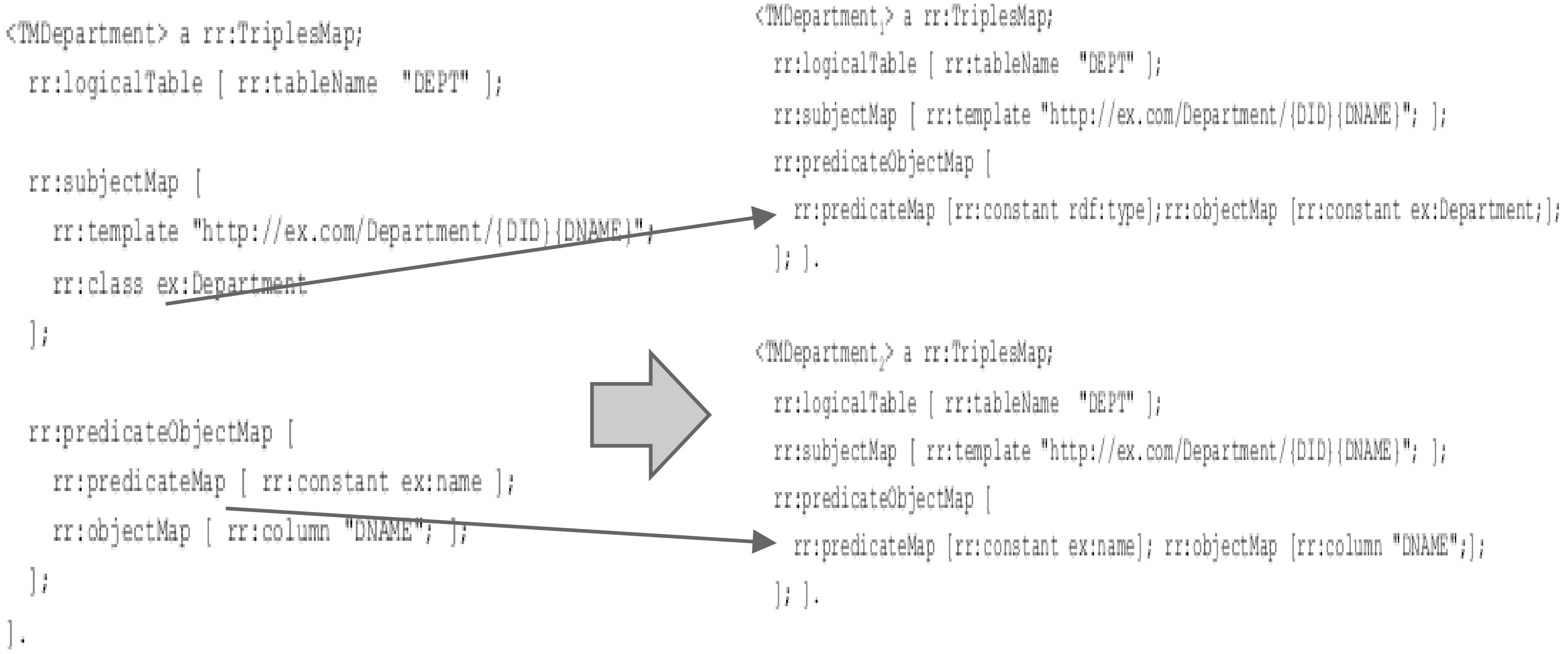


```
<TMEmployee> a rr:TriplesMap;
  rr:logicalTable [ rr:tableName  "EMP" ];

  rr:subjectMap [
    rr:template "http://ex.com/Person/{EID}";
    rr:class ex:Employee
  ];

  rr:predicateObjectMap [
    rr:predicateMap [ rr:constant ex:name ];
    rr:objectMap [ rr:column "LNAME"; ];
    rr:objectMap [ rr:column "FNAME"; ];
  ];

  rr:predicateObjectMap [
    rr:predicateMap [ rr:constant ex:fullname];
    rr:objectMap [ rr:template "{LNAME},{FNAME}"; ];
  ];

  rr:predicateObjectMap [
    rr:predicateMap [ rr:constant ex:worksIn];
    rr:objectMap [
      rr:parentTriplesMap <TMDepartment>;
      rr:joinCondition [ rr:child "DEPTID" ; rr:parent "DID" ; ]
    ];
].
```

```
<TMEmployee_1> a rr:TriplesMap;
  rr:logicalTable [ rr:tableName  "EMP" ];
  rr:subjectMap [ rr:template "http://ex.com/Person/{EID}";  ];
  rr:predicateObjectMap [
    rr:predicateMap [rr:constant rdf:type]; rr:objectMap [rr:constant ex:Employee];
  ]; ];

<TMEmployee_2> a rr:TriplesMap;
  rr:logicalTable [ rr:tableName  "EMP" ];
  rr:subjectMap [ rr:template "http://ex.com/Person/{EID}";  ];
  rr:predicateObjectMap [
    rr:predicateMap [ rr:constant ex:name ]; rr:objectMap [ rr:column "LNAME"; ];
  ]; ];

<TMEmployee_3> a rr:TriplesMap;
  rr:logicalTable [ rr:tableName  "EMP" ];
  rr:subjectMap [ rr:template "http://ex.com/Person/{EID}";  ];
  rr:predicateObjectMap [
    rr:predicateMap [ rr:constant ex:name ]; rr:objectMap [ rr:column "FNAME"; ];
  ]; ];

<TMEmployee_4> a rr:TriplesMap;
  rr:logicalTable [ rr:tableName  "EMP" ];
  rr:subjectMap [ rr:template "http://ex.com/Person/{EID}";  ];
  rr:predicateObjectMap [
    rr:predicateMap[rr:constant ex:name];rr:objectMap[rr:template "{LNAME},{FNAME}";];
  ]; ];

<TMEmployee_5> a rr:TriplesMap;
  rr:logicalTable [ rr:sqlQuery  """SELECT EMP.EID, DEPT.DID, DEPT.DNAME FROM EMP JOIN
DEPT ON EMP.DEPTID = DEPT.DID""" ];
  rr:subjectMap [ rr:template "http://ex.com/Person/{EID}";  ];
  rr:predicateObjectMap [
    rr:predicateMap [rr:constant ex:worksIn];
    rr:objectMap [rr:template "http://ex.com/Department/{DID}{DNAME}"];
  ]; ].
```

IP Problem Approach - Step 1 Normalize Mappings





```
<TMDepartment> a rr:TriplesMap;
  rr:logicalTable [ rr:tableName  "DEPT" ];

  rr:subjectMap [
    rr:template "http://ex.com/Department/{DID}{DNAME}";
    rr:class ex:Department
  ];

  rr:predicateObjectMap [
    rr:predicateMap [ rr:constant ex:name ];
    rr:objectMap [ rr:column "DNAME"; ];
  ];
].
```

```
<TMDepartment$_1$> a rr:TriplesMap;
  rr:logicalTable [ rr:tableName  "DEPT" ];
  rr:subjectMap [ rr:template "http://ex.com/Department/{DID}{DNAME}"; ];
  rr:predicateObjectMap [
    rr:predicateMap [rr:constant rdf:type];rr:objectMap [rr:constant ex:Department;];
  ]; ].
```

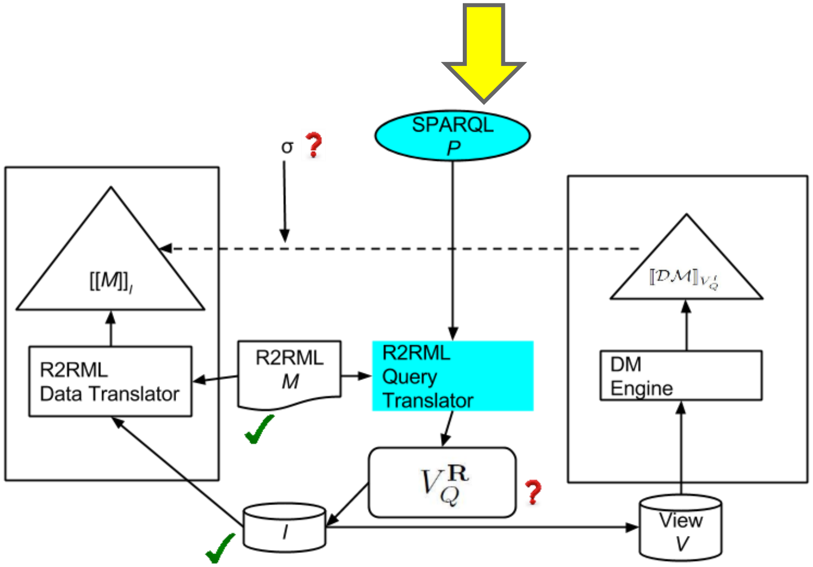
```
<TMDepartment$_2$> a rr:TriplesMap;
  rr:logicalTable [ rr:tableName  "DEPT" ];
  rr:subjectMap [ rr:template "http://ex.com/Department/{DID}{DNAME}"; ];
  rr:predicateObjectMap [
    rr:predicateMap [rr:constant ex:name]; rr:objectMap [rr:column "DNAME";];
  ]; ].
```

IP Problem Approach - Step 2 Generating Q

Set $Q = trans(P, M)$

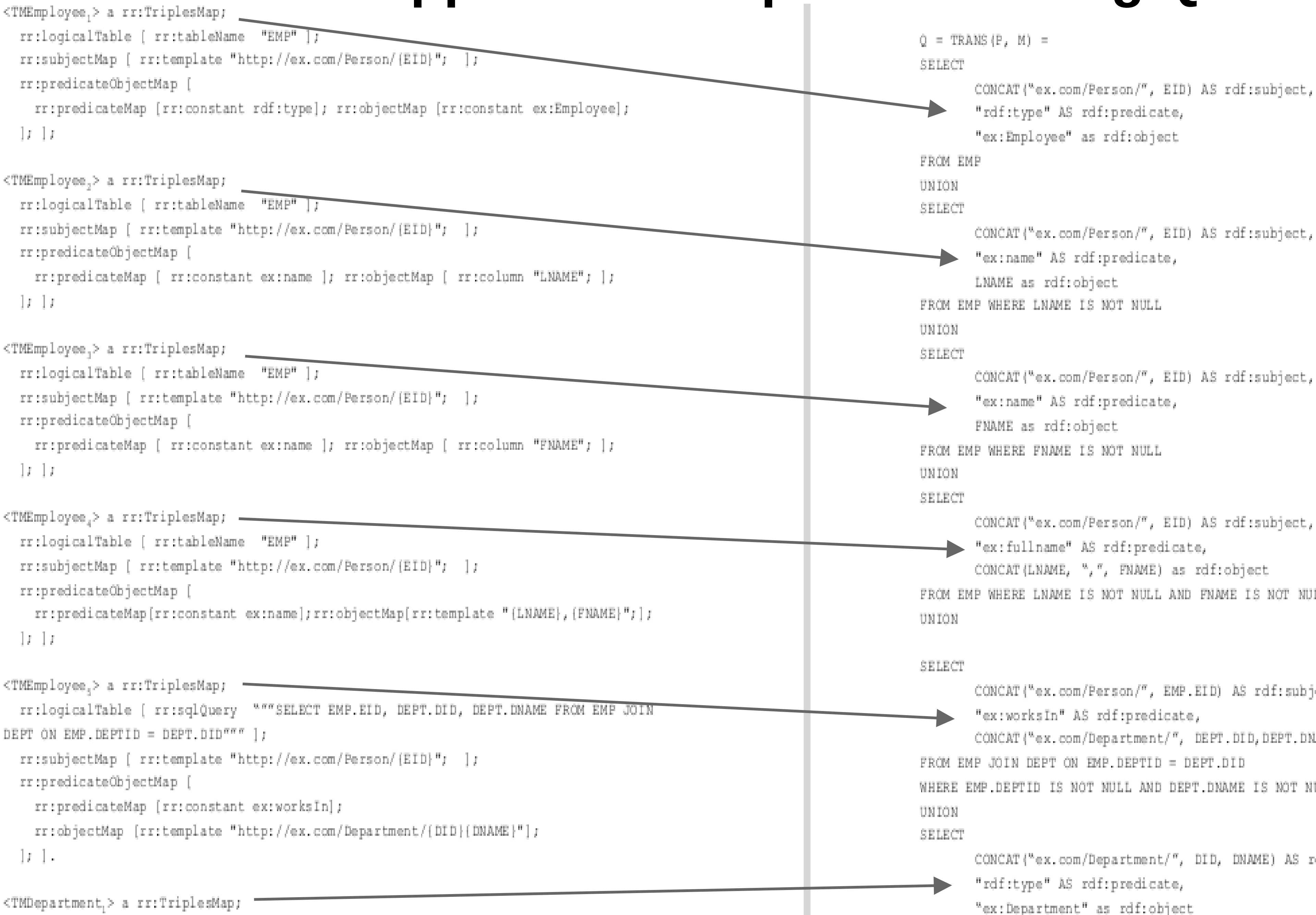
P = SELECT

```
    ?s AS ?rdf:subject
    ?p AS ?rdf:predicate
    ?o AS ?rdf:object
  WHERE { ?s ?p ?o }.
```



```
Q = TRANS(P, M) =
SELECT
        CONCAT("ex.com/Person/", EID) AS rdf:subject,
        "rdf:type" AS rdf:predicate,
        "ex:Employee" as rdf:object
FROM EMP
UNION
SELECT
        CONCAT("ex.com/Person/", EID) AS rdf:subject,
        "ex:name" AS rdf:predicate,
        LNAME as rdf:object
FROM EMP WHERE LNAME IS NOT NULL
UNION
SELECT
        CONCAT("ex.com/Person/", EID) AS rdf:subject,
        "ex:name" AS rdf:predicate,
        FNAME as rdf:object
FROM EMP WHERE FNAME IS NOT NULL
UNION
SELECT
        CONCAT("ex.com/Person/", EID) AS rdf:subject,
        "ex:fullname" AS rdf:predicate,
        CONCAT(LNAME, ",", FNAME) as rdf:object
FROM EMP WHERE LNAME IS NOT NULL AND FNAME IS NOT NULL
UNION

SELECT
        CONCAT("ex.com/Person/", EMP.EID) AS rdf:subject,
        "ex:worksIn" AS rdf:predicate,
        CONCAT("ex.com/Department/", DEPT.DID,DEPT.DNAME) as rdf:object
FROM EMP JOIN DEPT ON EMP.DEPTID = DEPT.DID
WHERE EMP.DEPTID IS NOT NULL AND DEPT.DNAME IS NOT NULL
UNION
SELECT
        CONCAT("ex.com/Department/", DID, DNAME) AS rdf:subject,
        "rdf:type" AS rdf:predicate,
        "ex:Department" as rdf:object
FROM DEPT WHERE DNAME IS NOT NULL
UNION
SELECT
        CONCAT("ex.com/Department/", DID, DNAME) AS rdf:subject,
        "ex:name" AS rdf:predicate,
        DNAME as rdf:object
FROM DEPT WHERE DNAME IS NOT NULL;
```

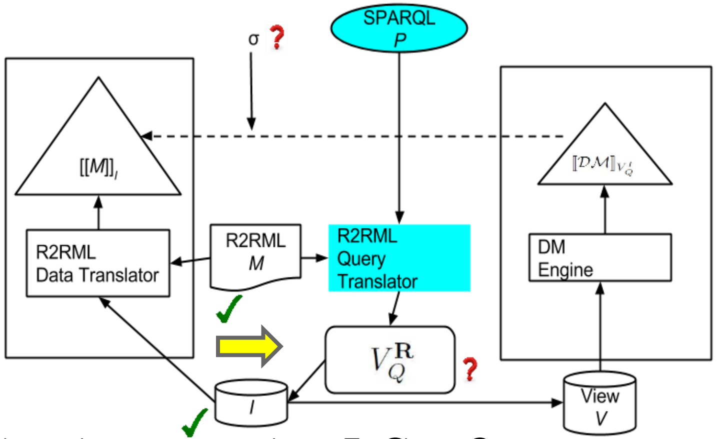
IP Problem Approach - Step 2 Generating Q



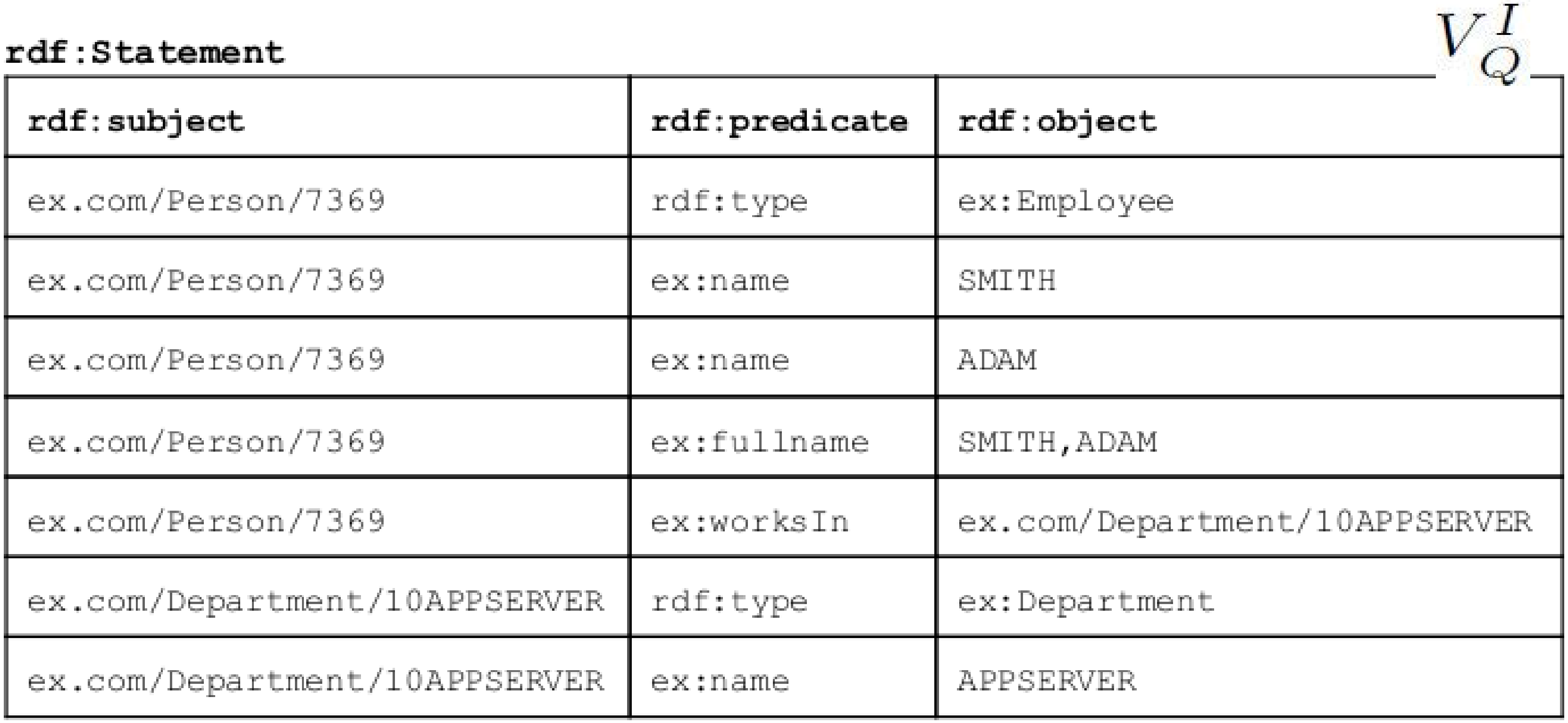
```
<TMEmployee1> a rr:TriplesMap;
  rr:logicalTable [ rr:tableName  "EMP" ];
  rr:subjectMap [ rr:template "http://ex.com/Person/{EID}";  ];
  rr:predicateObjectMap [
    rr:predicateMap [rr:constant rdf:type]; rr:objectMap [rr:constant ex:Employee];
  ]; ];

<TMEmployee2> a rr:TriplesMap;
  rr:logicalTable [ rr:tableName  "EMP" ];
  rr:subjectMap [ rr:template "http://ex.com/Person/{EID}";  ];
  rr:predicateObjectMap [
    rr:predicateMap [ rr:constant ex:name ]; rr:objectMap [ rr:column "LNAME"; ];
  ]; ];

<TMEmployee3> a rr:TriplesMap;
  rr:logicalTable [ rr:tableName  "EMP" ];
  rr:subjectMap [ rr:template "http://ex.com/Person/{EID}";  ];
  rr:predicateObjectMap [
    rr:predicateMap [ rr:constant ex:name ]; rr:objectMap [ rr:column "FNAME"; ];
  ]; ];

<TMEmployee4> a rr:TriplesMap;
  rr:logicalTable [ rr:tableName  "EMP" ];
  rr:subjectMap [ rr:template "http://ex.com/Person/{EID}";  ];
  rr:predicateObjectMap [
    rr:predicateMap[rr:constant ex:name];rr:objectMap[rr:template "{LNAME},{FNAME}";];
  ]; ];

<TMEmployee5> a rr:TriplesMap;
  rr:logicalTable [ rr:sqlQuery  """SELECT EMP.EID, DEPT.DID, DEPT.DNAME FROM EMP JOIN
DEPT ON EMP.DEPTID = DEPT.DID""" ];
  rr:subjectMap [ rr:template "http://ex.com/Person/{EID}";  ];
  rr:predicateObjectMap [
    rr:predicateMap [rr:constant ex:worksIn];
    rr:objectMap [rr:template "http://ex.com/Department/{DID}{DNAME}"];
  ]; ].

<TMDepartment1> a rr:TriplesMap;
```

```
Q = TRANS(P, M) =
SELECT
      CONCAT("ex.com/Person/", EID) AS rdf:subject,
      "rdf:type" AS rdf:predicate,
      "ex:Employee" as rdf:object
FROM EMP
UNION
SELECT
      CONCAT("ex.com/Person/", EID) AS rdf:subject,
      "ex:name" AS rdf:predicate,
      LNAME as rdf:object
FROM EMP WHERE LNAME IS NOT NULL
UNION
SELECT
      CONCAT("ex.com/Person/", EID) AS rdf:subject,
      "ex:name" AS rdf:predicate,
      FNAME as rdf:object
FROM EMP WHERE FNAME IS NOT NULL
UNION
SELECT
      CONCAT("ex.com/Person/", EID) AS rdf:subject,
      "ex:fullname" AS rdf:predicate,
      CONCAT(LNAME, ",", FNAME) as rdf:object
FROM EMP WHERE LNAME IS NOT NULL AND FNAME IS NOT NUL
UNION


SELECT
      CONCAT("ex.com/Person/", EMP.EID) AS rdf:subj
      "ex:worksIn" AS rdf:predicate,
      CONCAT("ex.com/Department/", DEPT.DID,DEPT.DN
FROM EMP JOIN DEPT ON EMP.DEPTID = DEPT.DID
WHERE EMP.DEPTID IS NOT NULL AND DEPT.DNAME IS NOT N
UNION
SELECT
      CONCAT("ex.com/Department/", DID, DNAME) AS r
      "rdf:type" AS rdf:predicate,
      "ex:Department" as rdf:object
```

IP Problem Approach - Step 3 Generating View



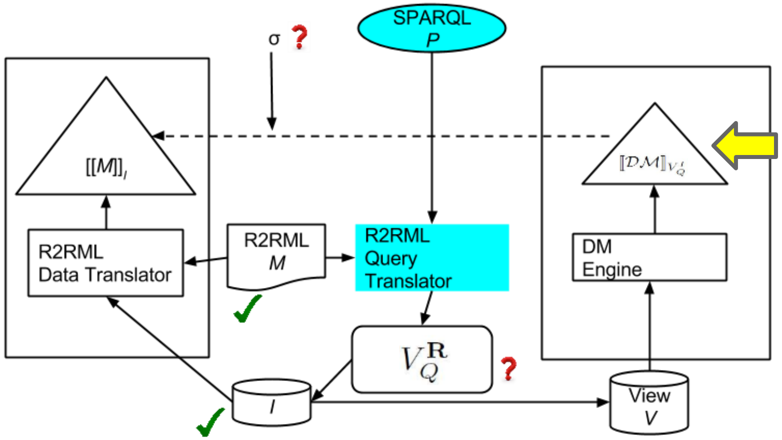
Set V_Q^R = `CREATE VIEW rdf:statement AS` Q;

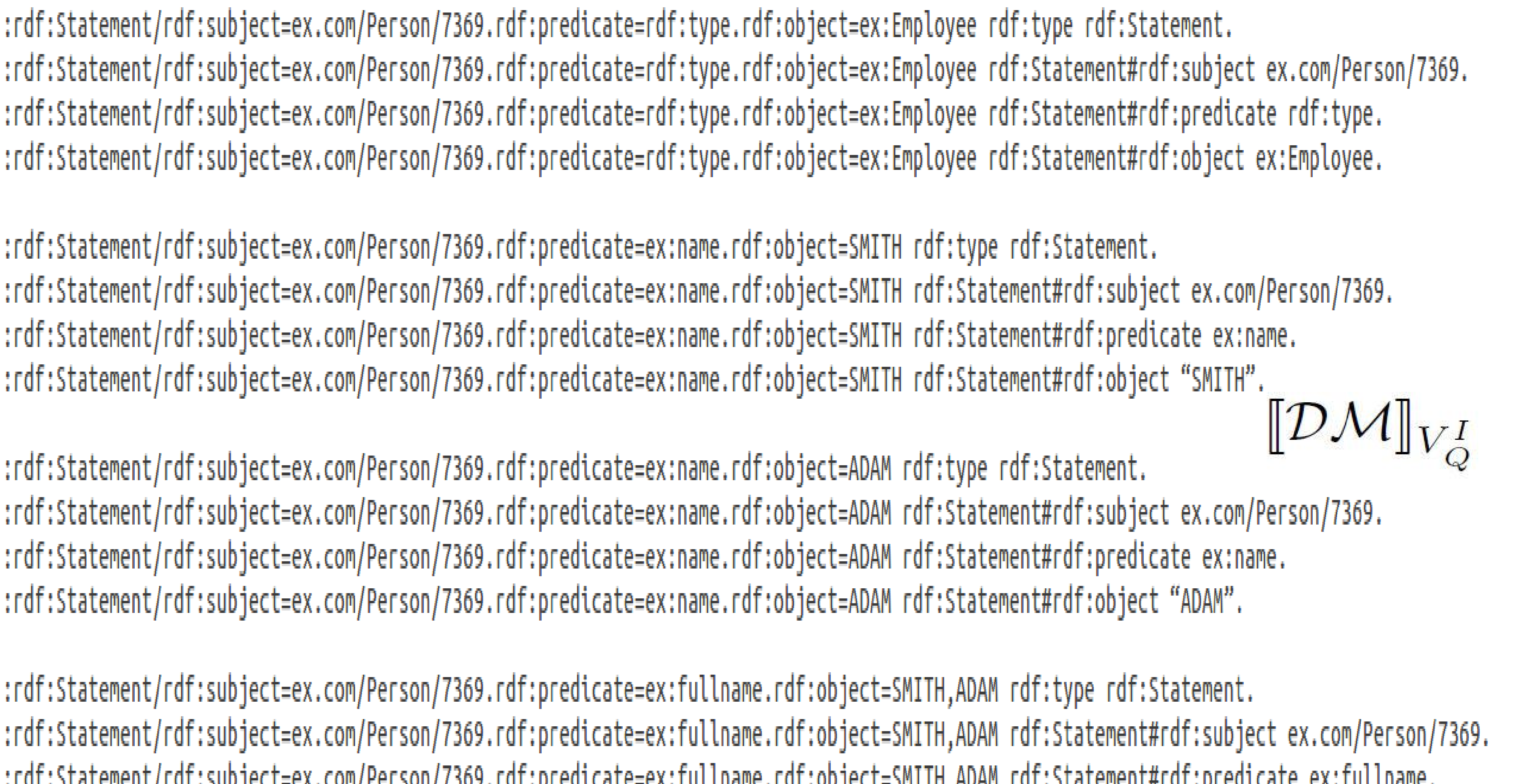


rdf:Statement V_Q^I

rdf:subject	rdf:predicate	rdf:object
ex.com/Person/7369	rdf:type	ex:Employee
ex.com/Person/7369	ex:name	SMITH
ex.com/Person/7369	ex:name	ADAM
ex.com/Person/7369	ex:fullname	SMITH,ADAM
ex.com/Person/7369	ex:worksIn	ex.com/Department/10APPSERVER
ex.com/Department/10APPSERVER	rdf:type	ex:Department
ex.com/Department/10APPSERVER	ex:name	APPSERVER

IP Problem Approach - Step 3
DM Evaluation

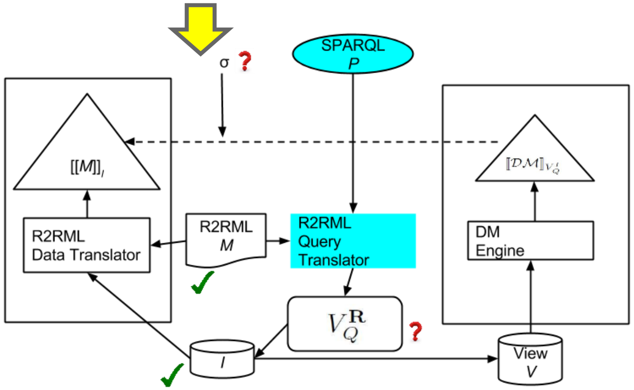




```
:rdf:Statement/rdf:subject=ex.com/Person/7369.rdf:predicate=rdf:type.rdf:object=ex:Employee rdf:type rdf:Statement.
:rdf:Statement/rdf:subject=ex.com/Person/7369.rdf:predicate=rdf:type.rdf:object=ex:Employee rdf:Statement#rdf:subject ex.com/Person/7369.
:rdf:Statement/rdf:subject=ex.com/Person/7369.rdf:predicate=rdf:type.rdf:object=ex:Employee rdf:Statement#rdf:predicate rdf:type.
:rdf:Statement/rdf:subject=ex.com/Person/7369.rdf:predicate=rdf:type.rdf:object=ex:Employee rdf:Statement#rdf:object ex:Employee.

:rdf:Statement/rdf:subject=ex.com/Person/7369.rdf:predicate=ex:name.rdf:object=SMITH rdf:type rdf:Statement.
:rdf:Statement/rdf:subject=ex.com/Person/7369.rdf:predicate=ex:name.rdf:object=SMITH rdf:Statement#rdf:subject ex.com/Person/7369.
:rdf:Statement/rdf:subject=ex.com/Person/7369.rdf:predicate=ex:name.rdf:object=SMITH rdf:Statement#rdf:predicate ex:name.
:rdf:Statement/rdf:subject=ex.com/Person/7369.rdf:predicate=ex:name.rdf:object=SMITH rdf:Statement#rdf:object “SMITH”.

:rdf:Statement/rdf:subject=ex.com/Person/7369.rdf:predicate=ex:name.rdf:object=ADAM rdf:type rdf:Statement.
:rdf:Statement/rdf:subject=ex.com/Person/7369.rdf:predicate=ex:name.rdf:object=ADAM rdf:Statement#rdf:subject ex.com/Person/7369.
:rdf:Statement/rdf:subject=ex.com/Person/7369.rdf:predicate=ex:name.rdf:object=ADAM rdf:Statement#rdf:predicate ex:name.
:rdf:Statement/rdf:subject=ex.com/Person/7369.rdf:predicate=ex:name.rdf:object=ADAM rdf:Statement#rdf:object “ADAM”.

:rdf:Statement/rdf:subject=ex.com/Person/7369.rdf:predicate=ex:fullname.rdf:object=SMITH,ADAM rdf:type rdf:Statement.
:rdf:Statement/rdf:subject=ex.com/Person/7369.rdf:predicate=ex:fullname.rdf:object=SMITH,ADAM rdf:Statement#rdf:subject ex.com/Person/7369.
:rdf:Statement/rdf:subject=ex.com/Person/7369.rdf:predicate=ex:fullname.rdf:object=SMITH,ADAM rdf:Statement#rdf:predicate ex:fullname.
```

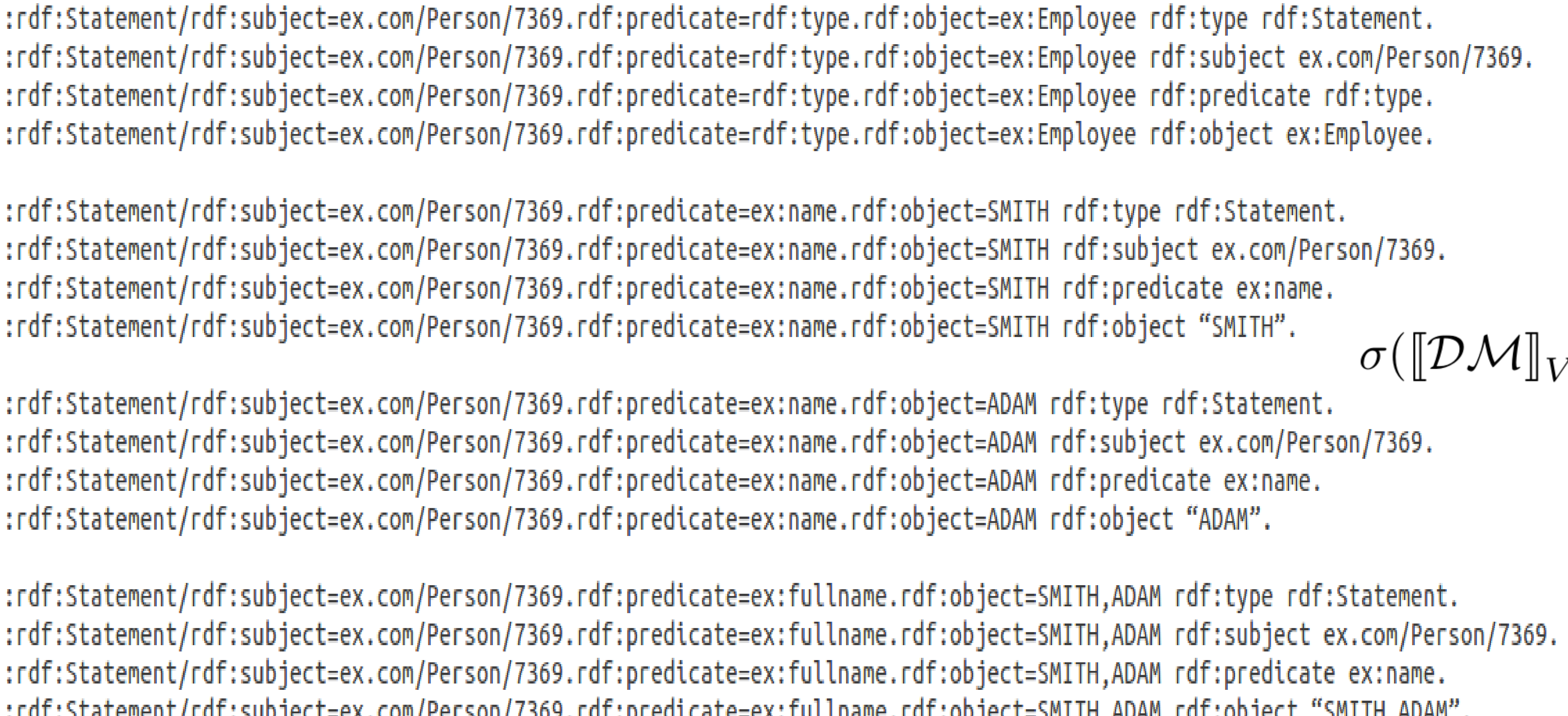
$$[\![\mathcal{DM}]\!]_{V_Q^I}$$

IP Problem Approach - Step 4 Substitution



Set σ =

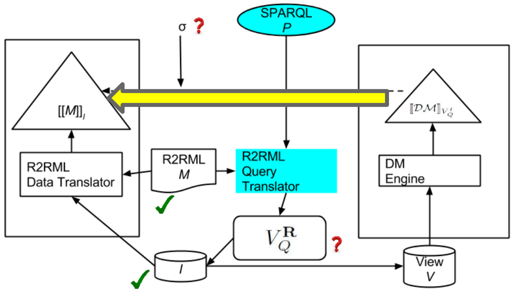
- Replace `rdf:Statement#rdf:subject` with `rdf:subject`
- Replace `rdf:Statement#rdf:predicate` with `rdf:predicate`
- Replace `rdf:Statement#rdf:object` with `rdf:object`



```
:rdf:Statement/rdf:subject=ex.com/Person/7369.rdf:predicate=rdf:type.rdf:object=ex:Employee rdf:type rdf:Statement.
:rdf:Statement/rdf:subject=ex.com/Person/7369.rdf:predicate=rdf:type.rdf:object=ex:Employee rdf:subject ex.com/Person/7369.
:rdf:Statement/rdf:subject=ex.com/Person/7369.rdf:predicate=rdf:type.rdf:object=ex:Employee rdf:predicate rdf:type.
:rdf:Statement/rdf:subject=ex.com/Person/7369.rdf:predicate=rdf:type.rdf:object=ex:Employee rdf:object ex:Employee.

:rdf:Statement/rdf:subject=ex.com/Person/7369.rdf:predicate=ex:name.rdf:object=SMITH rdf:type rdf:Statement.
:rdf:Statement/rdf:subject=ex.com/Person/7369.rdf:predicate=ex:name.rdf:object=SMITH rdf:subject ex.com/Person/7369.
:rdf:Statement/rdf:subject=ex.com/Person/7369.rdf:predicate=ex:name.rdf:object=SMITH rdf:predicate ex:name.
:rdf:Statement/rdf:subject=ex.com/Person/7369.rdf:predicate=ex:name.rdf:object=SMITH rdf:object “SMITH”.

:rdf:Statement/rdf:subject=ex.com/Person/7369.rdf:predicate=ex:name.rdf:object=ADAM rdf:type rdf:Statement.
:rdf:Statement/rdf:subject=ex.com/Person/7369.rdf:predicate=ex:name.rdf:object=ADAM rdf:subject ex.com/Person/7369.
:rdf:Statement/rdf:subject=ex.com/Person/7369.rdf:predicate=ex:name.rdf:object=ADAM rdf:predicate ex:name.
:rdf:Statement/rdf:subject=ex.com/Person/7369.rdf:predicate=ex:name.rdf:object=ADAM rdf:object “ADAM”.

:rdf:Statement/rdf:subject=ex.com/Person/7369.rdf:predicate=ex:fullname.rdf:object=SMITH,ADAM rdf:type rdf:Statement.
:rdf:Statement/rdf:subject=ex.com/Person/7369.rdf:predicate=ex:fullname.rdf:object=SMITH,ADAM rdf:subject ex.com/Person/7369.
:rdf:Statement/rdf:subject=ex.com/Person/7369.rdf:predicate=ex:fullname.rdf:object=SMITH,ADAM rdf:predicate ex:name.
:rdf:Statement/rdf:subject=ex.com/Person/7369.rdf:predicate=ex:fullname.rdf:object=SMITH,ADAM rdf:object “SMITH,ADAM”.
```

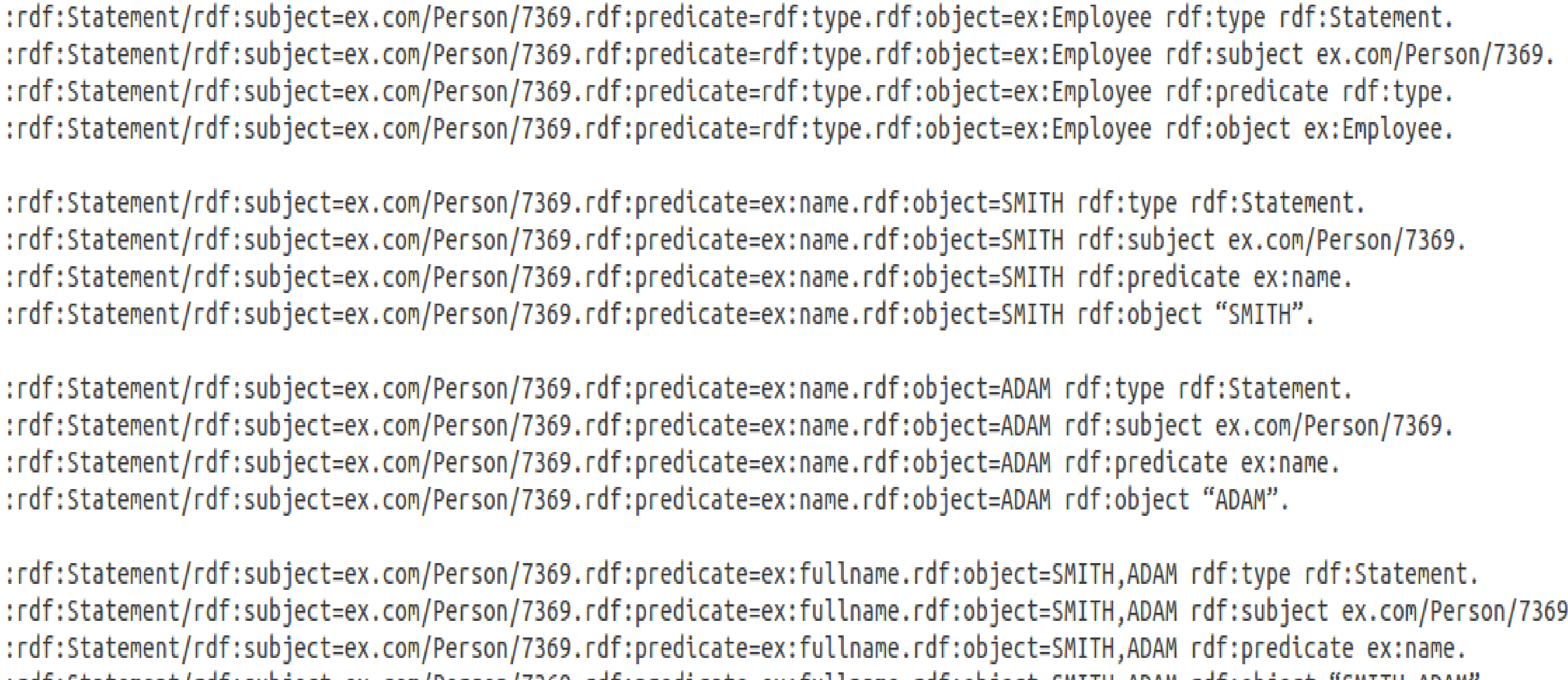
$$\sigma([\![\mathcal{DM}]\!]_{V_Q^I})$$

IP Problem Solved



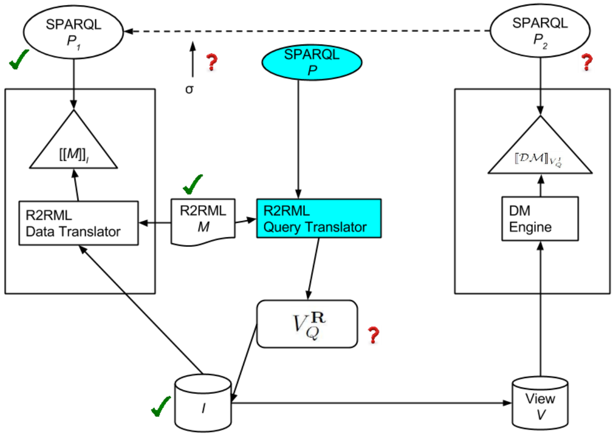
<http://ex.com/Person/7369> rdf:type ex:Employee.
<http://ex.com/Person/7369> ex:name "SMITH".
<http://ex.com/Person/7369> ex:name "ADAM".
<http://ex.com/Person/7369> ex:fullname "SMITH,ADAM".
<http://ex.com/Person/7369> ex:worksIn <http://ex.com/Department/10APPSERVER>.
<http://ex.com/Department/10APPSERVER> rdf:type ex:Department.
<http://ex.com/Department/10APPSERVER> ex:name "APPSERVER".

$$[\![\mathcal{M}]\!]_I = \sigma([\![\mathcal{DM}]\!]_{V_Q^I})$$



```
:rdf:Statement/rdf:subject=ex.com/Person/7369.rdf:predicate=rdf:type.rdf:object=ex:Employee rdf:type rdf:Statement.
:rdf:Statement/rdf:subject=ex.com/Person/7369.rdf:predicate=rdf:type.rdf:object=ex:Employee rdf:subject ex.com/Person/7369.
:rdf:Statement/rdf:subject=ex.com/Person/7369.rdf:predicate=rdf:type.rdf:object=ex:Employee rdf:predicate rdf:type.
:rdf:Statement/rdf:subject=ex.com/Person/7369.rdf:predicate=rdf:type.rdf:object=ex:Employee rdf:object ex:Employee.

:rdf:Statement/rdf:subject=ex.com/Person/7369.rdf:predicate=ex:name.rdf:object=SMITH rdf:type rdf:Statement.
:rdf:Statement/rdf:subject=ex.com/Person/7369.rdf:predicate=ex:name.rdf:object=SMITH rdf:subject ex.com/Person/7369.
:rdf:Statement/rdf:subject=ex.com/Person/7369.rdf:predicate=ex:name.rdf:object=SMITH rdf:predicate ex:name.
:rdf:Statement/rdf:subject=ex.com/Person/7369.rdf:predicate=ex:name.rdf:object=SMITH rdf:object “SMITH”.

:rdf:Statement/rdf:subject=ex.com/Person/7369.rdf:predicate=ex:name.rdf:object=ADAM rdf:type rdf:Statement.
:rdf:Statement/rdf:subject=ex.com/Person/7369.rdf:predicate=ex:name.rdf:object=ADAM rdf:subject ex.com/Person/7369.
:rdf:Statement/rdf:subject=ex.com/Person/7369.rdf:predicate=ex:name.rdf:object=ADAM rdf:predicate ex:name.
:rdf:Statement/rdf:subject=ex.com/Person/7369.rdf:predicate=ex:name.rdf:object=ADAM rdf:object “ADAM”.

:rdf:Statement/rdf:subject=ex.com/Person/7369.rdf:predicate=ex:fullname.rdf:object=SMITH,ADAM rdf:type rdf:Statement.
:rdf:Statement/rdf:subject=ex.com/Person/7369.rdf:predicate=ex:fullname.rdf:object=SMITH,ADAM rdf:subject ex.com/Person/7369
:rdf:Statement/rdf:subject=ex.com/Person/7369.rdf:predicate=ex:fullname.rdf:object=SMITH,ADAM rdf:predicate ex:name.
:rdf:Statement/rdf:subject=ex.com/Person/7369.rdf:predicate=ex:fullname.rdf:object=SMITH,ADAM rdf:object “SMITH,ADAM”
```

QRP Problem
Vertical Approach

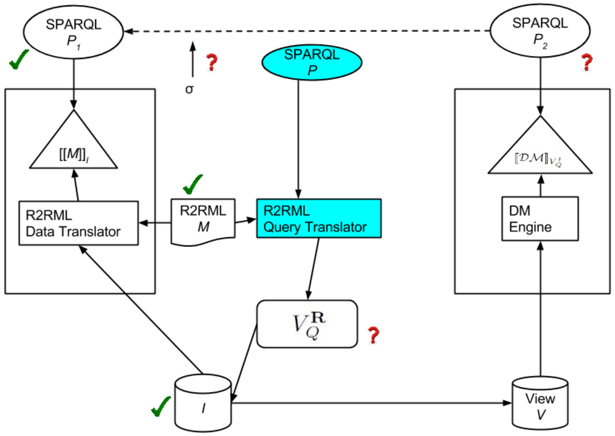


1. Normalize mappings in M_{Lite} form to M where M is in 3rd NF
2. Set Q = *trans(P, M)*
   - ○ P = `SELECT ?s AS ?rdf:subject ?p AS rdf:predicate ?o AS rdf:object WHERE { ?s ?p ?o }.`
3. Set $V_Q^{\mathbf{R}}$ = `CREATE VIEW rdf:statement AS` *`Q`*`;`
4. Set σ = {}
5. For P_1 = `SELECT ?s ?p ?o WHERE {?s ?p ?o}`

set P_2 = `SELECT ?s ?p ?o WHERE {`

```
?st a rdf:Statement .
?st rdf:Statement#rdf:subject ?s .
?st rdf:Statement#rdf:predicate ?p .
?st rdf:Statement#rdf:object ?o .}
```

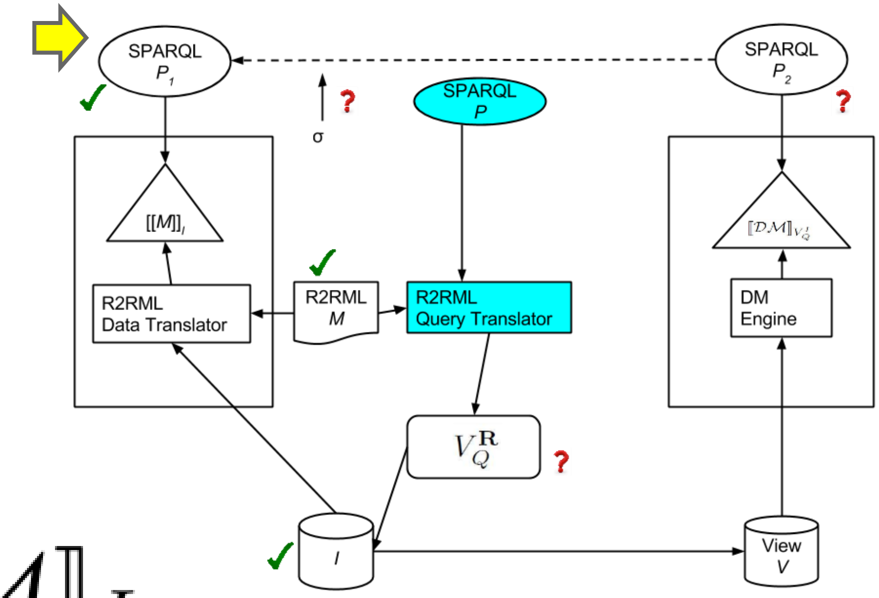
QRP Problem
Approach - Step 1-3



Same as IPP Approach 1 Step 1-3

1. Normalize M_{Core} to M where M is in 3rd NF
2. Set Q = *trans(P, M)*
   - *P* $V_Q^{\mathbf{R}}$`ELECT ?s AS ?rdf:subject ?p AS rdf:predicate ?o AS rdf:object WHERE { ?s ?p ?o }.`
3. Set = `CREATE VIEW rdf:statement AS` *Q*`;`

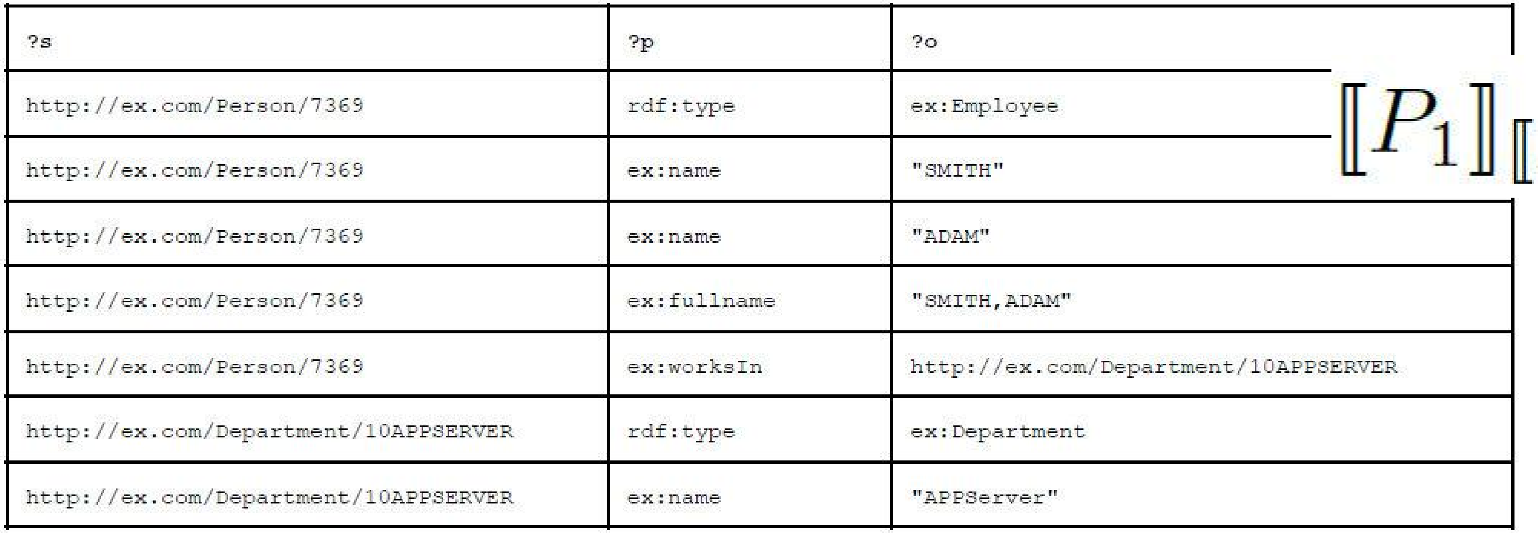
QRP Problem Approach - Step 4 (P_1)



P_1 = `SELECT ?s ?p ?o WHERE {?s ?p ?o}`

<http://ex.com/Person/7369> rdf:type ex:Employee.
<http://ex.com/Person/7369> ex:name "SMITH".
<http://ex.com/Person/7369> ex:name "ADAM".
<http://ex.com/Person/7369> ex:fullname "SMITH,ADAM".
<http://ex.com/Person/7369> ex:worksIn <http://ex.com/Department/10APPSERVER>.
<http://ex.com/Department/10APPSERVER> rdf:type ex:Department.
<http://ex.com/Department/10APPSERVER> ex:name "APPSERVER".

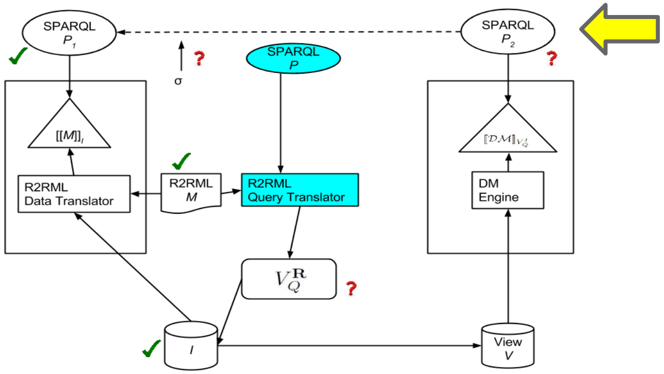
$$[\![\mathcal{M}]\!]_I$$

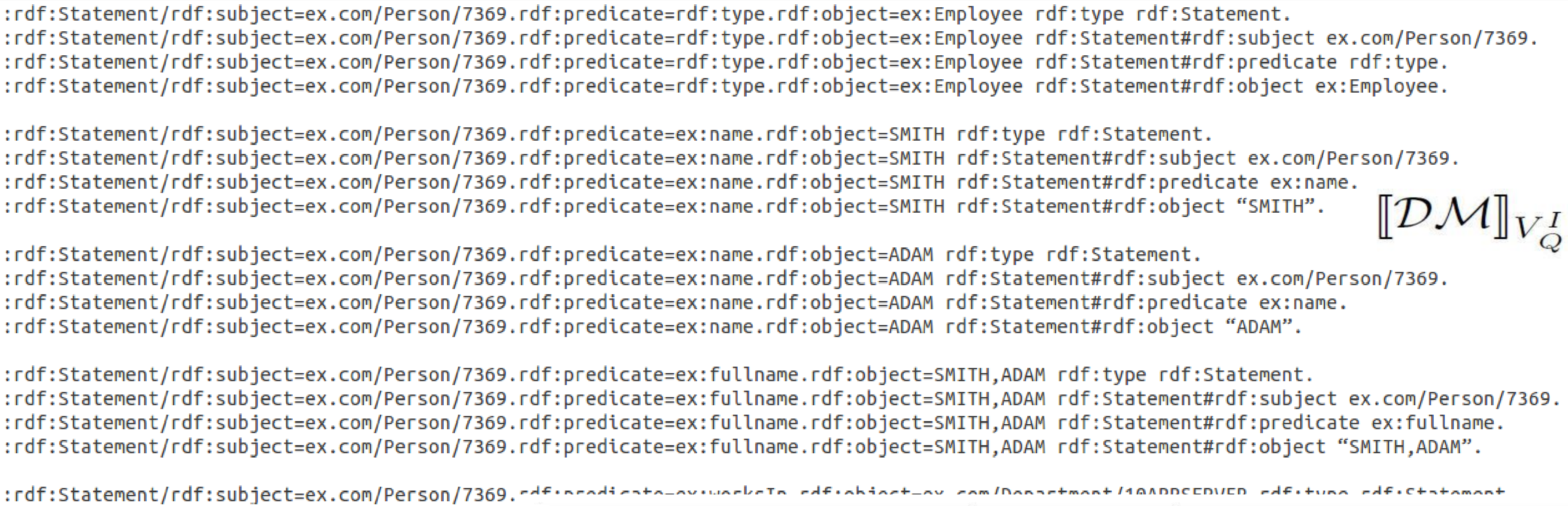


?s	?p	?o
http://ex.com/Person/7369	rdf:type	ex:Employee
http://ex.com/Person/7369	ex:name	"SMITH"
http://ex.com/Person/7369	ex:name	"ADAM"
http://ex.com/Person/7369	ex:fullname	"SMITH,ADAM"
http://ex.com/Person/7369	ex:worksIn	http://ex.com/Department/10APPSERVER
http://ex.com/Department/10APPSERVER	rdf:type	ex:Department
http://ex.com/Department/10APPSERVER	ex:name	"APPServer"

$$[\![P_1]\!]_{[\![M]\!]_I}$$

QRP Problem Approach Step 4 (P_2)

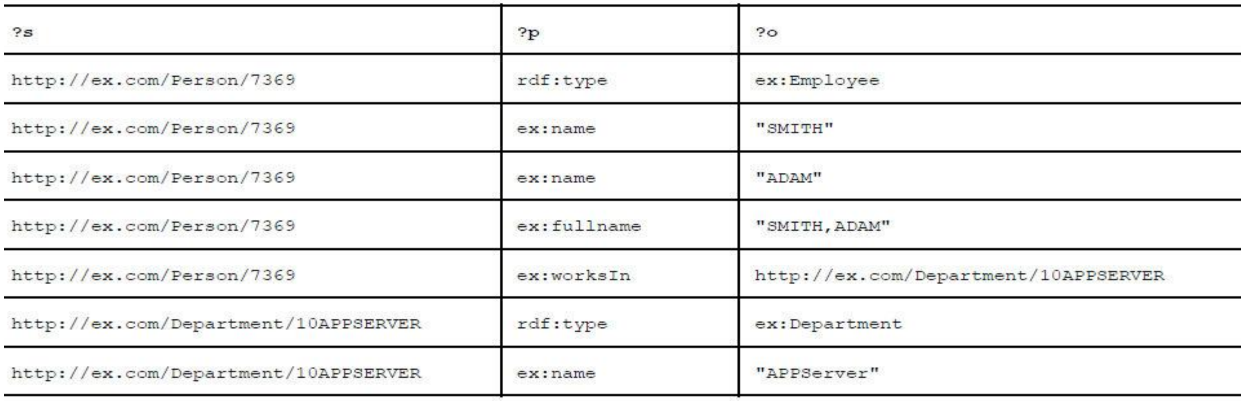




```
:rdf:Statement/rdf:subject=ex.com/Person/7369.rdf:predicate=rdf:type.rdf:object=ex:Employee rdf:type rdf:Statement.
:rdf:Statement/rdf:subject=ex.com/Person/7369.rdf:predicate=rdf:type.rdf:object=ex:Employee rdf:Statement#rdf:subject ex.com/Person/7369.
:rdf:Statement/rdf:subject=ex.com/Person/7369.rdf:predicate=rdf:type.rdf:object=ex:Employee rdf:Statement#rdf:predicate rdf:type.
:rdf:Statement/rdf:subject=ex.com/Person/7369.rdf:predicate=rdf:type.rdf:object=ex:Employee rdf:Statement#rdf:object ex:Employee.

:rdf:Statement/rdf:subject=ex.com/Person/7369.rdf:predicate=ex:name.rdf:object=SMITH rdf:type rdf:Statement.
:rdf:Statement/rdf:subject=ex.com/Person/7369.rdf:predicate=ex:name.rdf:object=SMITH rdf:Statement#rdf:subject ex.com/Person/7369.
:rdf:Statement/rdf:subject=ex.com/Person/7369.rdf:predicate=ex:name.rdf:object=SMITH rdf:Statement#rdf:predicate ex:name.
:rdf:Statement/rdf:subject=ex.com/Person/7369.rdf:predicate=ex:name.rdf:object=SMITH rdf:Statement#rdf:object “SMITH”.

:rdf:Statement/rdf:subject=ex.com/Person/7369.rdf:predicate=ex:name.rdf:object=ADAM rdf:type rdf:Statement.
:rdf:Statement/rdf:subject=ex.com/Person/7369.rdf:predicate=ex:name.rdf:object=ADAM rdf:Statement#rdf:subject ex.com/Person/7369.
:rdf:Statement/rdf:subject=ex.com/Person/7369.rdf:predicate=ex:name.rdf:object=ADAM rdf:Statement#rdf:predicate ex:name.
:rdf:Statement/rdf:subject=ex.com/Person/7369.rdf:predicate=ex:name.rdf:object=ADAM rdf:Statement#rdf:object “ADAM”.

:rdf:Statement/rdf:subject=ex.com/Person/7369.rdf:predicate=ex:fullname.rdf:object=SMITH,ADAM rdf:type rdf:Statement.
:rdf:Statement/rdf:subject=ex.com/Person/7369.rdf:predicate=ex:fullname.rdf:object=SMITH,ADAM rdf:Statement#rdf:subject ex.com/Person/7369.
:rdf:Statement/rdf:subject=ex.com/Person/7369.rdf:predicate=ex:fullname.rdf:object=SMITH,ADAM rdf:Statement#rdf:predicate ex:fullname.
:rdf:Statement/rdf:subject=ex.com/Person/7369.rdf:predicate=ex:fullname.rdf:object=SMITH,ADAM rdf:Statement#rdf:object “SMITH,ADAM”.

:rdf:Statement/rdf:subject=ex.com/Person/7369.rdf:predicate=ex:worksIn.rdf:object=ex.com/Department/10APPSERVER rdf:type rdf:Statement
```

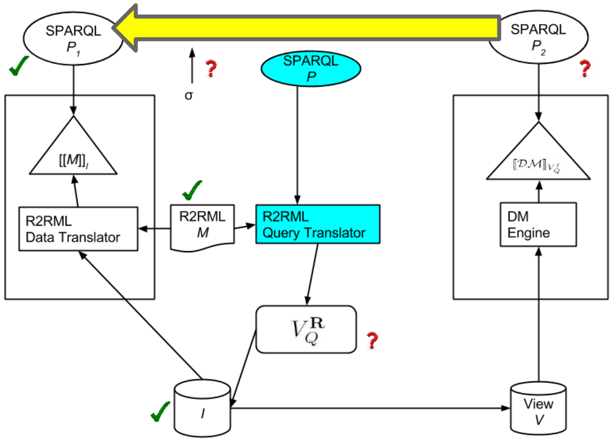
$$[\![\mathcal{DM}]\!]_{V_Q^I}$$

P_2 =
```
SELECT ?s ?p ?o WHERE {
  ?st a rdf:Statement .
  ?st rdf:Statement#rdf:subject ?s .
  ?st rdf:Statement#rdf:predicate ?p .
  ?st rdf:Statement#rdf:object ?o .}
```



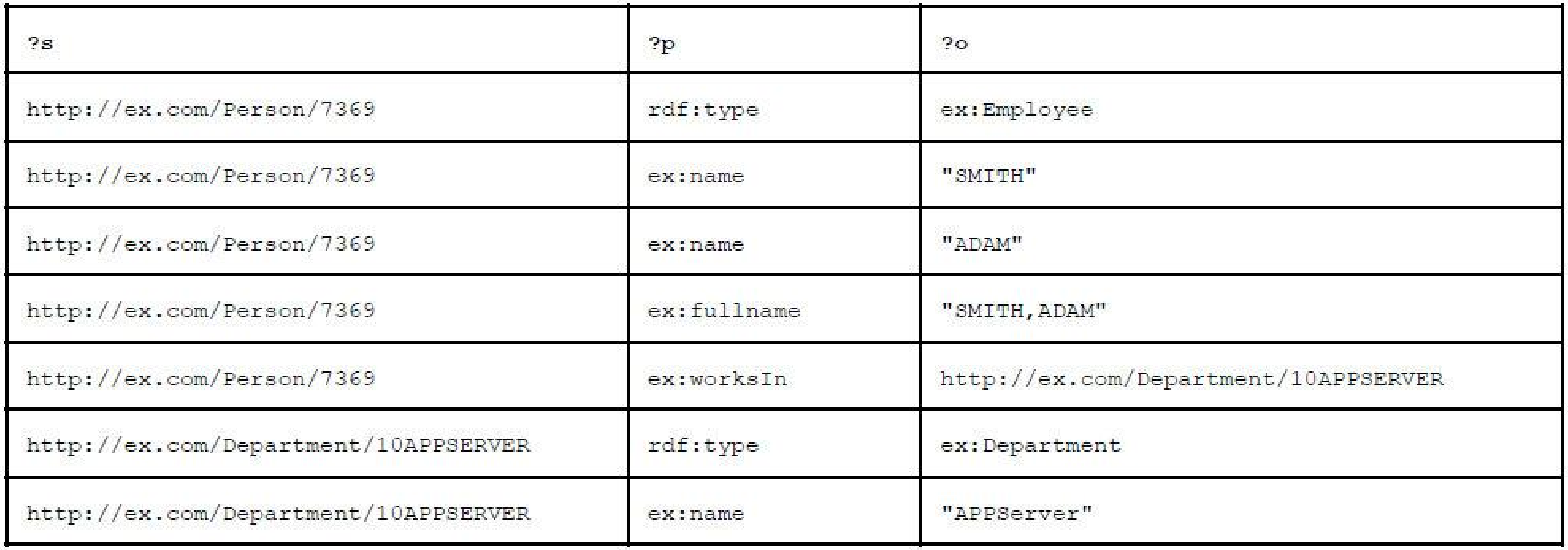
?s	?p	?o
http://ex.com/Person/7369	rdf:type	ex:Employee
http://ex.com/Person/7369	ex:name	"SMITH"
http://ex.com/Person/7369	ex:name	"ADAM"
http://ex.com/Person/7369	ex:fullname	"SMITH,ADAM"
http://ex.com/Person/7369	ex:worksIn	http://ex.com/Department/10APPSERVER
http://ex.com/Department/10APPSERVER	rdf:type	ex:Department
http://ex.com/Department/10APPSERVER	ex:name	"APPServer"

QRP Problem Solved



$\sigma = \{\}$

$$[\![P_1]\!]_{[\![M]\!]_I} = \sigma([\![P_2]\!]_{[\![\mathcal{DM}]\!]_{V_Q^I}})$$



?s	?p	?o
http://ex.com/Person/7369	rdf:type	ex:Employee
http://ex.com/Person/7369	ex:name	"SMITH"
http://ex.com/Person/7369	ex:name	"ADAM"
http://ex.com/Person/7369	ex:fullname	"SMITH,ADAM"
http://ex.com/Person/7369	ex:worksIn	http://ex.com/Department/10APPSERVER
http://ex.com/Department/10APPSERVER	rdf:type	ex:Department
http://ex.com/Department/10APPSERVER	ex:name	"APPServer"

Summary

- R2RML <-> DM
  - DM -> R2RML (obvious)
  - R2RML -> DM (?)
    - $\text{R2RML}_{\text{Lite}}$
- We say they are “equivalent”, if
  - Information preservation property holds
  - Query result preservation property holds
- The consequence
  - R2RML engine (morph-RDB, D2RQ, ontop, etc) is not the only option for evaluating $\text{R2RML}_{\text{Lite}}$ mappings

Future Work (FW) and Conclusions (C)

- FW1: Supporting more expressive R2RML fragment
- FW2: Use horizontal approach (see next slide)
- C1: DM + View + Substitution is as expressive as $\text{R2RML}_{\text{Lite}}$

- C2: Be very careful when choosing your internship topic!

(FW2) Sneak Peek of Horizontal Approach

- Use mappings NF1
- Use self-join elimination
- Without reification
- Generate multiple tables, not a single triples table in the view
  - $P_{Department}$ = SELECT ?s ?ex_name ?ex_department WHERE {

    ```
    ?s rdf:type ex:Department .
    ?s ex:name  ?ex_name . }
    ```

  - $P_{Employee}$ = SELECT ?s ?ex_name ?ex_fullname ?ex_department WHERE {

    ```
    ?s rdf:type ex:Employee .
    ?s ex:name  ?ex_name .
    ?s ex:fullname ?ex_fullname .
    ?s ex:worksIn ?ex_worksIn. }
    ```