




A SQL-based Approach for Mapping Relational Data to RDF

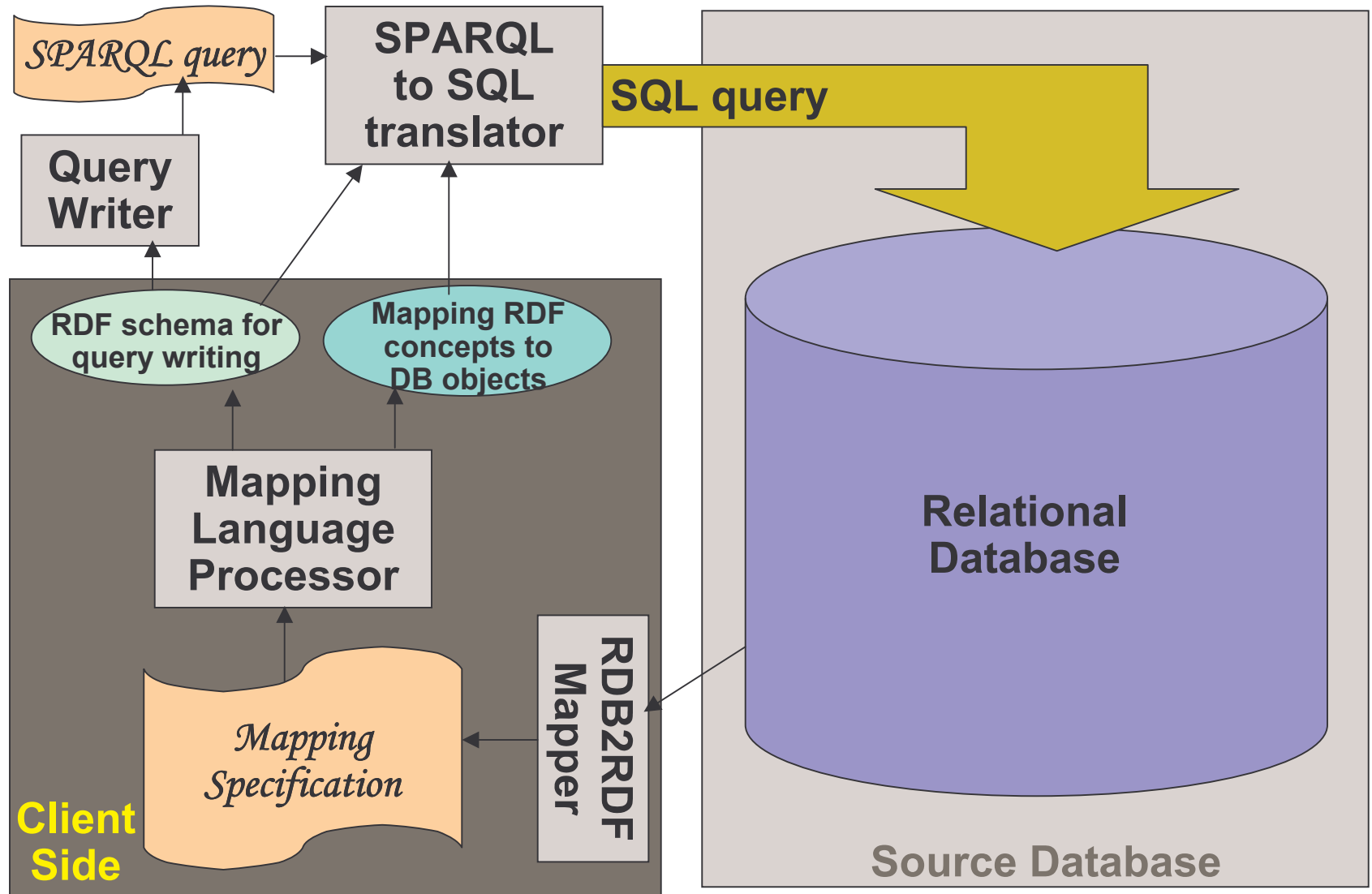
Souripriya Das and Seema Sundara

Database Semantic Technologies Group, Oracle

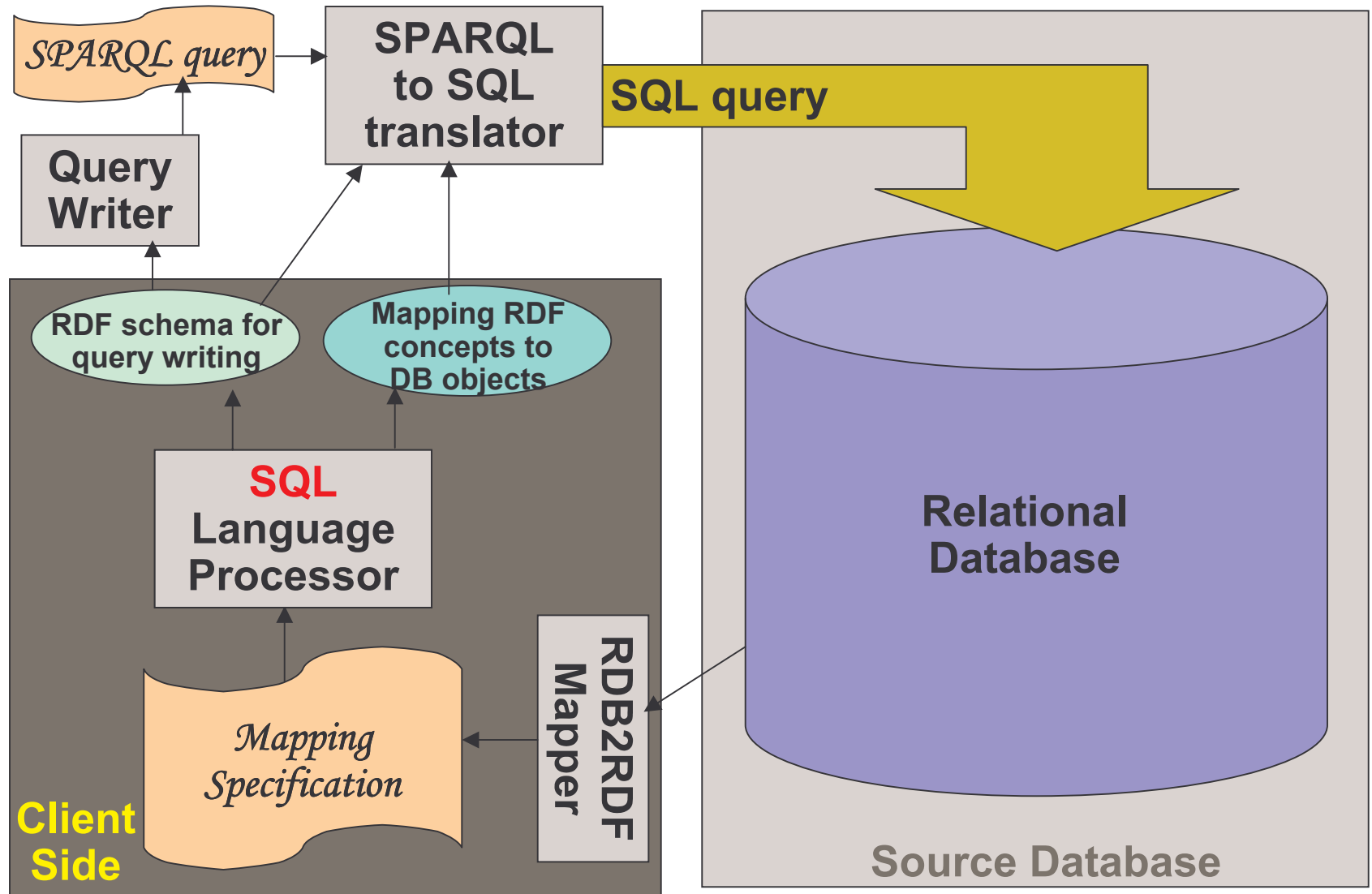


THE FOLLOWING IS INTENDED TO OUTLINE OUR GENERAL PRODUCT DIRECTION. IT IS INTENDED FOR INFORMATION PURPOSES ONLY, AND MAY NOT BE INCORPORATED INTO ANY CONTRACT. IT IS NOT A COMMITMENT TO DELIVER ANY MATERIAL, CODE, OR FUNCTIONALITY, AND SHOULD NOT BE RELIED UPON IN MAKING PURCHASING DECISION. THE DEVELOPMENT, RELEASE, AND TIMING OF ANY FEATURES OR FUNCTIONALITY DESCRIBED FOR ORACLE'S PRODUCTS REMAINS AT THE SOLE DISCRETION OF ORACLE.

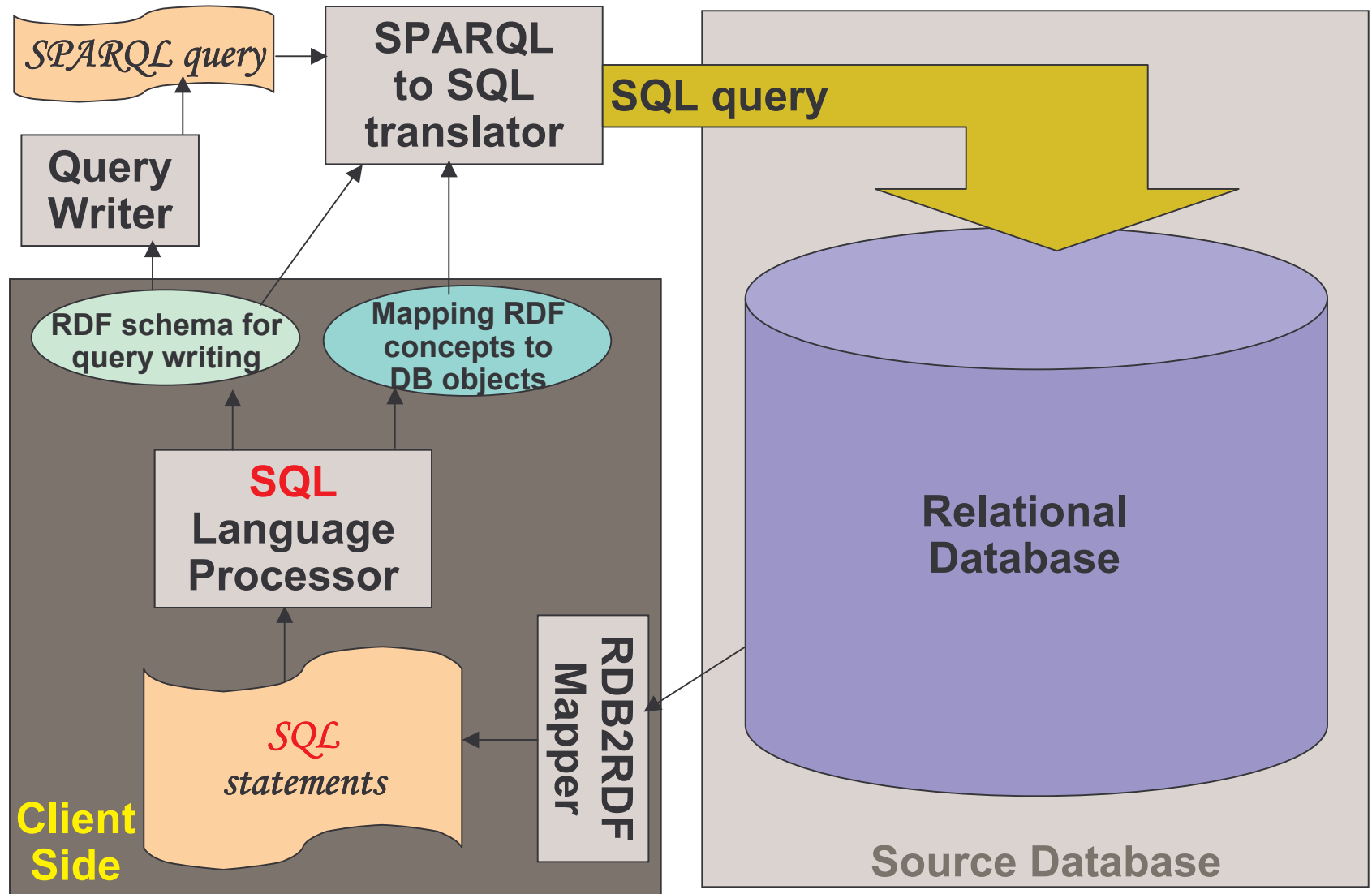
Architecture Diagram for RDB2RDF processing



Architecture Diagram for RDB2RDF processing

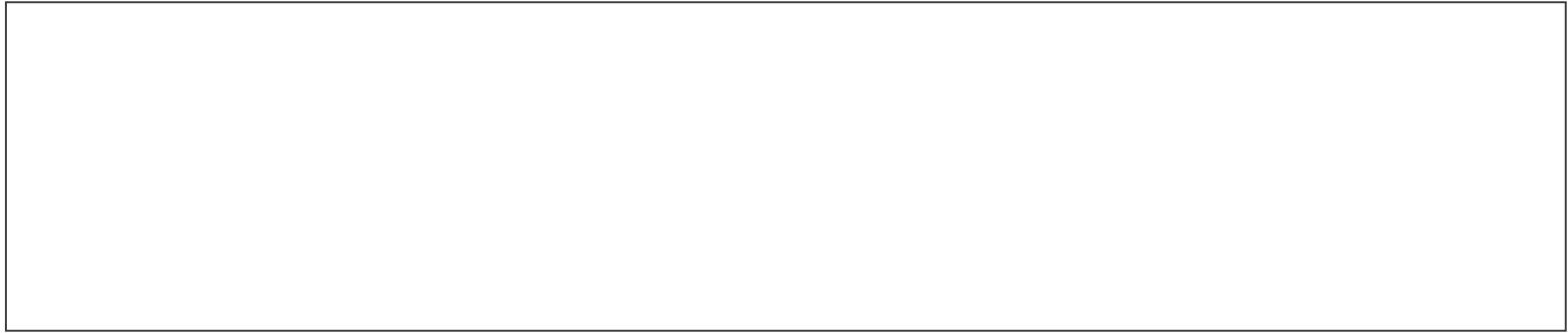


Architecture Diagram for RDB2RDF processing





Overview





Overview

- **Basis** for defining an RDFS/OWL class and its properties based on relational data stored in a source database:
 - A **SQL query** specification against the source DB
 - **No restriction on query complexity**
 - Related foreign and unique key constraint definitions.



Overview

- **Basis** for defining an RDFS/OWL class and its properties based on relational data stored in a source database:
 - A **SQL query** specification against the source DB
 - **No restriction on query complexity**
 - Related foreign and unique key constraint definitions.
- Client side capabilities may vary
 - **Strong-DB**: An RDBMS with support for Views and Constraints on Views. (Example: Oracle database server)
 - **No-DB**: Only connects to database using JDBC or ODBC.



Overview

- **Basis** for defining an RDFS/OWL class and its properties based on relational data stored in a source database:
 - A **SQL query** specification against the source DB
 - **No restriction on query complexity**
 - Related foreign and unique key constraint definitions.
- Client side capabilities may vary
 - **Strong-DB**: An RDBMS with support for Views and Constraints on Views. (Example: Oracle database server)
 - **No-DB**: Only connects to database using JDBC or ODBC.
- Proposed mapping language
(Note: **SQL is not a part of the mapping language**)
 - **Strong-DB** → No new language. Employ some *conventions* with SQL (e.g., when naming Views, View cols, and constraints).
 - **No-DB** → Simple language to specify mapping between RDF classes, properties and SQL queries, query projections, constraints.

DB View → RDF Class and Properties

| empno | ename | job | deptno |
|-------|-------|-------|--------|
| 100 | John | R&D | 11 |
| 200 | Jane | R&D | 11 |
| 300 | Brad | R&D | 22 |
| 400 | Glen | Sales | 33 |
| 500 | Kurt | Sales | 44 |

Diagram illustrating the mapping of a database view to RDF Class and Properties:






- The column **empno** is mapped to the property **c_prm_empno**.
- The column **deptno** is mapped to the property **c_ref_deptno**.

DB View → RDF Class and Properties

| InstURI | empno | ename | job | deptno |
|---------|-------|-------|-------|--------|
| <100> | 100 | John | R&D | 11 |
| <200> | 200 | Jane | R&D | 11 |
| <300> | 300 | Brad | R&D | 22 |
| <400> | 400 | Glen | Sales | 33 |
| <500> | 500 | Kurt | Sales | 44 |

The diagram illustrates the mapping of a database view to RDF classes and properties. The table above shows the data. The label 'c_prm_empno' is positioned above the 'empno' column. The label 'c_ref_deptno' is positioned above the 'deptno' column. The label 'class' is positioned below the 'job' column, with an arrow pointing to it.

DB View → RDF Class and Properties

| InstURI | empno | ename | job | deptno | rdf:type | status | rdf:type |
|---------|-------|-------|-------|--------|---|--------|----------|
| <100> | 100 | John | R&D | 11 |  | part | <part> |
| <200> | 200 | Jane | R&D | 11 |  | full | <full> |
| <300> | 300 | Brad | R&D | 22 |  | intrn | <intrn> |
| <400> | 400 | Glen | Sales | 33 |  | full | <full> |
| <500> | 500 | Kurt | Sales | 44 |  | full | <full> |

c_prm_empno
c_ref_deptno
job - based classes
status - based classes

class

Example: DEPT view defines Class and Properties

```
CREATE VIEW "<xyz.com/dept>" AS
```

```
SELECT
```

```
'<xyz.com/dept/' || deptno || '>'
```

```
, deptno
```

```
, dname
```

```
, loc
```

```
FROM
```

```
dept
```

Example: DEPT view defines Class and Properties

CREATE VIEW “<xyz.com/dept>” AS

SELECT

| | |
|-----------------------------------|---------------------------|
| ‘<xyz.com/dept/’ deptno ‘>’ | InstURI |
| , deptno | “<xyz.com/dept/deptno>” |
| , dname | “<xyz.com/dept/Name>” |
| , loc | “<xyz.com/dept/Location>” |

FROM

dept

Example: DEPT view defines Class and Properties

CREATE VIEW “<xyz.com/dept>” AS

SELECT

| | |
|-----------------------------------|---------------------------|
| ‘<xyz.com/dept/’ deptno ‘>’ | InstURI |
| , deptno | “<xyz.com/dept/deptno>” |
| , dname | “<xyz.com/dept/Name>” |
| , loc | “<xyz.com/dept/Location>” |

FROM

dept

ALTER VIEW “<xyz.com/dept>”
add constraint “<xyz.com/dept/c_unq_deptno>”
unique (“<xyz.com/dept/deptno>”)
disable novalidate;



Example: EMP view defines Class and Properties

```
CREATE VIEW "<xyz.com/emp>" AS
```

```
SELECT
```

```
'<xyz.com/emp/' || empno || '>'
```

```
, empno
```

```
, ename
```

```
, '<xyz.com/emp/job/' || job || '>'
```

```
, job
```

```
, deptno
```

```
FROM
```

```
emp
```


Example: EMP view defines Class and Properties

CREATE VIEW “<xyz.com/emp>” AS

SELECT

| | |
|-------------------------------------|-------------------------|
| ‘<xyz.com/emp/’ empno ‘>’ | InstURI |
| , empno | “<xyz.com/emp/empno>” |
| , ename | “<xyz.com/emp/Name>” |
| , ‘<xyz.com/emp/job/’ job ‘>’ | “rdf:type” |
| , job | “<xyz.com/emp/job>” |
| , deptno | “<xyz.com/emp/deptNum>” |

FROM

emp

Example: EMP view defines Class and Properties

CREATE VIEW “<xyz.com/emp>” AS

SELECT

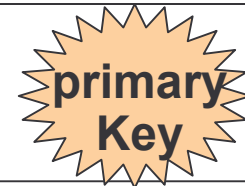
| | |
|-------------------------------------|-------------------------|
| ‘<xyz.com/emp/’ empno ‘>’ | InstURI |
| , empno | “<xyz.com/emp/empno>” |
| , ename | “<xyz.com/emp/Name>” |
| , ‘<xyz.com/emp/job/’ job ‘>’ | “rdf:type” |
| , job | “<xyz.com/emp/job>” |
| , deptno | “<xyz.com/emp/deptNum>” |

FROM emp

ALTER VIEW “<xyz.com/emp>”

add constraint “<xyz.com/emp/c_prm_empno>”

primary key (“<xyz.com/emp/empno>”) disable novalidate;



Example: EMP view defines Class and Properties

CREATE VIEW “<xyz.com/emp>” AS

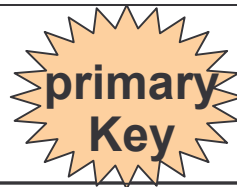
SELECT

| | |
|-------------------------------------|-------------------------|
| ‘<xyz.com/emp/’ empno ‘>’ | InstURI |
| , empno | “<xyz.com/emp/empno>” |
| , ename | “<xyz.com/emp/Name>” |
| , ‘<xyz.com/emp/job/’ job ‘>’ | “rdf:type” |
| , job | “<xyz.com/emp/job>” |
| , deptno | “<xyz.com/emp/deptNum>” |

FROM emp

ALTER VIEW “<xyz.com/emp>”

add constraint “<xyz.com/emp/c_prm_empno>”
primary key (“<xyz.com/emp/empno>”) disable novalidate;



ALTER VIEW “<xyz.com/emp>”

add constraint “<xyz.com/emp/c_ref_deptno>”
foreign key (“<xyz.com/emp/deptNum>”)
references “<xyz.com/dept>” (“<xyz.com/dept/deptno>”)
disable novalidate;



Example: EMP view with multiple rdf:type columns

CREATE VIEW “<xyz.com/emp>” AS

SELECT

| | |
|---|--------------------------------|
| ‘<xyz.com/emp/’ empno ‘>’ | InstURI |
| , empno | “<xyz.com/emp/empno>” |
| , ename | “<xyz.com/emp/Name>” |
| , ‘<xyz.com/emp/job/’ job ‘>’ | “<xyz.com/emp/job/rdf:type>” |
| , job | “<xyz.com/emp/job>” |
| , deptno | “<xyz.com/emp/deptNum>” |
| , ‘<xyz.com/emp/etype/’ etype ‘>’ | “<xyz.com/emp/etype/rdf:type>” |
| , etype | “<xyz.com/emp/etype>” |

FROM

emp

Example: EMP view with multiple rdf:type columns

CREATE VIEW “<xyz.com/emp>” AS

SELECT

| | |
|---|--------------------------------|
| ‘<xyz.com/emp/’ empno ‘>’ | InstURI |
| , empno | “<xyz.com/emp/empno>” |
| , ename | “<xyz.com/emp/Name>” |
| , ‘<xyz.com/emp/job/’ job ‘>’ | “<xyz.com/emp/job/rdf:type>” |
| , job | “<xyz.com/emp/job>” |
| , deptno | “<xyz.com/emp/deptNum>” |
| , ‘<xyz.com/emp/etype/’ etype ‘>’ | “<xyz.com/emp/etype/rdf:type>” |
| , etype | “<xyz.com/emp/etype>” |

FROM

emp

ALTER VIEW “<xyz.com/emp>”
add constraint “<xyz.com/emp/c_prm_empno>”
primary key (“<xyz.com/emp/empno>”) disable novalidate;

**primary
Key**

Example: EMP view with multiple rdf:type columns

CREATE VIEW “<xyz.com/emp>” AS

SELECT

| | |
|---|--------------------------------|
| ‘<xyz.com/emp/’ empno ‘>’ | InstURI |
| , empno | “<xyz.com/emp/empno>” |
| , ename | “<xyz.com/emp/Name>” |
| , ‘<xyz.com/emp/job/’ job ‘>’ | “<xyz.com/emp/job/rdf:type>” |
| , job | “<xyz.com/emp/job>” |
| , deptno | “<xyz.com/emp/deptNum>” |
| , ‘<xyz.com/emp/etype/’ etype ‘>’ | “<xyz.com/emp/etype/rdf:type>” |
| , etype | “<xyz.com/emp/etype>” |

FROM

emp

ALTER VIEW “<xyz.com/emp>”

add constraint “<xyz.com/emp/c_prm_empno>”
primary key (“<xyz.com/emp/empno>”) disable novalidate;

**primary
Key**

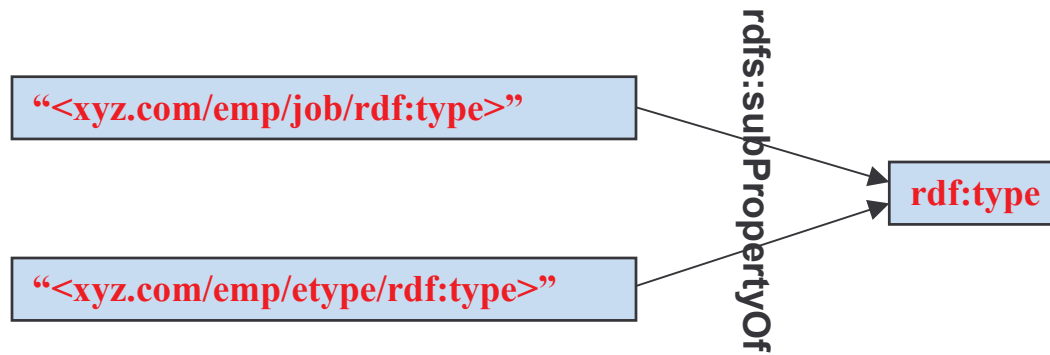
ALTER VIEW “<xyz.com/emp>”

add constraint “<xyz.com/emp/c_ref_deptno>”
foreign key (“<xyz.com/emp/deptNum>”)
references “<xyz.com/dept>” (“<xyz.com/dept/deptno>”)
disable novalidate;

**foreign
Key**



Subproperties of rdf:type





Prefix-based DB identifiers (stay within length-limits)

RDF_PREFIX_MAP\$

| View | Prefix | Expansion | |
|------------|--------|-----------------|-----------------------|
| “xyz:emp” | xyz: | <xyz.com/> | Prefix for class name |
| “xyz:emp” | emp: | <xyz.com/emp/> | Prefix for cols/cons |
| “xyz:dept” | xyz: | <xyz.com/> | Prefix for class name |
| “xyz:dept” | dept: | <xyz.com/dept/> | Prefix for cols/cons |

- ADD Unique constraint (View, Prefix)
- ADD Unique constraint (View, Expansion)

No DB: <ViewName, SQLdefString> info

| ViewName | SQLdefString |
|------------|---|
| "xyz:dept" | <pre> SELECT '<xyz.com/dept/' deptno '>' , deptno , dname , loc FROM dept </pre> |
| "xyz:emp" | <pre> SELECT '<xyz.com/emp/' empno '>' , empno , ename , '<xyz.com/emp/job/' job '>' , job , deptno FROM emp </pre> |

No DB: <ViewName, SQLdefString> info

| ViewName | SQLdefString | | | | | | | | | | | | | | | |
|------------|--|-------------------------------------|-----------------------------------|---------|---------|----------|---------------|--|-------------------------------------|-------------|-------|-------|-----------------|------|------|--|
| "xyz:dept" | <table border="1"> <tr> <td>SELECT</td> <td>'<xyz.com/dept/' deptno '>'</td> <td>InstURI</td> </tr> <tr> <td></td> <td>, deptno</td> <td>"dept:deptno"</td> </tr> <tr> <td></td> <td>, dname</td> <td>"dept:Name"</td> </tr> <tr> <td></td> <td>, loc</td> <td>"dept:Location"</td> </tr> <tr> <td>FROM</td> <td>dept</td> <td></td> </tr> </table> | SELECT | '<xyz.com/dept/' deptno '>' | InstURI | | , deptno | "dept:deptno" | | , dname | "dept:Name" | | , loc | "dept:Location" | FROM | dept | |
| | SELECT | '<xyz.com/dept/' deptno '>' | InstURI | | | | | | | | | | | | | |
| | | , deptno | "dept:deptno" | | | | | | | | | | | | | |
| | | , dname | "dept:Name" | | | | | | | | | | | | | |
| | | , loc | "dept:Location" | | | | | | | | | | | | | |
| FROM | dept | | | | | | | | | | | | | | | |
| "xyz:emp" | <table border="1"> <tr> <td>SELECT</td> <td>'<xyz.com/emp/' empno '>'</td> </tr> <tr> <td></td> <td>, empno</td> </tr> <tr> <td></td> <td>, ename</td> </tr> <tr> <td></td> <td>, '<xyz.com/emp/job/' job '>'</td> </tr> <tr> <td></td> <td>, job</td> </tr> <tr> <td></td> <td>, deptno</td> </tr> <tr> <td>FROM</td> <td>emp</td> </tr> </table> | SELECT | '<xyz.com/emp/' empno '>' | | , empno | | , ename | | , '<xyz.com/emp/job/' job '>' | | , job | | , deptno | FROM | emp | |
| | SELECT | '<xyz.com/emp/' empno '>' | | | | | | | | | | | | | | |
| | | , empno | | | | | | | | | | | | | | |
| | | , ename | | | | | | | | | | | | | | |
| | | , '<xyz.com/emp/job/' job '>' | | | | | | | | | | | | | | |
| | | , job | | | | | | | | | | | | | | |
| | , deptno | | | | | | | | | | | | | | | |
| FROM | emp | | | | | | | | | | | | | | | |

No DB: <ViewName, SQLdefString> info

| ViewName | SQLdefString | | | | | | | | | | | | | | | | | | | | | |
|-------------|--|-------------------------------------|-----------------------------------|---------|--|----------|---------------|--|---------|-------------|--|-------------------------------------|-----------------|-------------|-------|-----------|--|----------|---------------|-------------|-----|--|
| "xyz:dept" | <table border="1"> <tr> <td>SELECT</td> <td>'<xyz.com/dept/' deptno '>'</td> <td>InstURI</td> </tr> <tr> <td></td> <td>, deptno</td> <td>"dept:deptno"</td> </tr> <tr> <td></td> <td>, dname</td> <td>"dept:Name"</td> </tr> <tr> <td></td> <td>, loc</td> <td>"dept:Location"</td> </tr> <tr> <td>FROM</td> <td>dept</td> <td></td> </tr> </table> | SELECT | '<xyz.com/dept/' deptno '>' | InstURI | | , deptno | "dept:deptno" | | , dname | "dept:Name" | | , loc | "dept:Location" | FROM | dept | | | | | | | |
| | SELECT | '<xyz.com/dept/' deptno '>' | InstURI | | | | | | | | | | | | | | | | | | | |
| | | , deptno | "dept:deptno" | | | | | | | | | | | | | | | | | | | |
| | | , dname | "dept:Name" | | | | | | | | | | | | | | | | | | | |
| | | , loc | "dept:Location" | | | | | | | | | | | | | | | | | | | |
| FROM | dept | | | | | | | | | | | | | | | | | | | | | |
| "xyz:emp" | <table border="1"> <tr> <td>SELECT</td> <td>'<xyz.com/emp/' empno '>'</td> <td>InstURI</td> </tr> <tr> <td></td> <td>, empno</td> <td>"emp:empno"</td> </tr> <tr> <td></td> <td>, ename</td> <td>"emp:Name"</td> </tr> <tr> <td></td> <td>, '<xyz.com/emp/job/' job '>'</td> <td>"rdf:type"</td> </tr> <tr> <td></td> <td>, job</td> <td>"emp:job"</td> </tr> <tr> <td></td> <td>, deptno</td> <td>"emp:deptNum"</td> </tr> <tr> <td>FROM</td> <td>emp</td> <td></td> </tr> </table> | SELECT | '<xyz.com/emp/' empno '>' | InstURI | | , empno | "emp:empno" | | , ename | "emp:Name" | | , '<xyz.com/emp/job/' job '>' | "rdf:type" | | , job | "emp:job" | | , deptno | "emp:deptNum" | FROM | emp | |
| | SELECT | '<xyz.com/emp/' empno '>' | InstURI | | | | | | | | | | | | | | | | | | | |
| | | , empno | "emp:empno" | | | | | | | | | | | | | | | | | | | |
| | | , ename | "emp:Name" | | | | | | | | | | | | | | | | | | | |
| | | , '<xyz.com/emp/job/' job '>' | "rdf:type" | | | | | | | | | | | | | | | | | | | |
| | | , job | "emp:job" | | | | | | | | | | | | | | | | | | | |
| | , deptno | "emp:deptNum" | | | | | | | | | | | | | | | | | | | | |
| FROM | emp | | | | | | | | | | | | | | | | | | | | | |



No DB: View Constraint info

| ConsName | ConsType | ViewName | RefConsName |
|---------------------|-----------|------------|---------------------|
| “dept:c_unq_deptno” | Unique | “xyz:dept” | |
| “emp:c_prm_empno” | Primary | “xyz:emp” | |
| “emp:c_ref_deptno” | Reference | “xyz:emp” | “dept:c_unq_deptno” |

| ConsName | ViewName | ColName | ColPosInKey |
|---------------------|------------|---------------|-------------|
| “dept:c_unq_deptno” | “xyz:dept” | “dept:deptno” | 1 |
| “emp:c_prm_empno” | “xyz:emp” | “emp:empno” | 1 |
| “emp:c_ref_deptno” | “xyz:emp” | “emp:deptNum” | 1 |



No DB: <ViewName, ColPos, Property> info

| ViewName | ColPos | Property |
|------------|--------|------------------|
| “xyz:dept” | 1 | InstURI |
| “xyz:dept” | 2 | dept:deptno |
| “xyz:dept” | 3 | dept:Name |
| “xyz:dept” | 4 | dept:Location |
| “xyz:emp” | 1 | InstURI |
| “xyz:emp” | 2 | emp:empno |
| “xyz:emp” | 3 | emp:Name |
| “xyz:emp” | 4 | rdf:type |
| “xyz:emp” | 5 | emp:job |
| “xyz:emp” | 6 | emp:deptNum |
| “xyz:emp” | 0 | emp:c_ref_deptno |



No DB: A simple syntax for Classes

```
Class (ViewName, SQLdefString)
```

```
Class ("xyz:dept", <sql-def-for-dept>)
```

```
Class ("xyz:emp", <sql-def-for-emp>)
```

No DB: A simple syntax for Properties¹

Property (PropertyName, ViewName, ColPos, Range)

Property (“dept:InstURI”, “xyz:dept”, 1, “xyz:dept”)

Property (“dept:deptno”, “xyz:dept”, 2, “xsd:positiveInteger”)

Property (“dept:Name”, “xyz:dept”, 3, “xsd:string”)

Property (“dept:location”, “xyz:dept”, 4, “xsd:string”)

Property (“emp:InstURI”, “xyz:emp”, 1, “xyz:emp”)

Property (“emp:empno”, “xyz:emp”, 2, “xsd:positiveInteger”)

Property (“emp:Name”, “xyz:emp”, 3, “xsd:string”)

Property (“emp:rdf:type”, “xyz:emp”, 4, “rdfs:Class”)

Property (“emp:job”, “xyz:emp”, 5, “xsd:string”)

Property (“emp:deptNum”, “xyz:emp”, 6, “xsd:positiveInteger”)

Property (“emp:c_ref_deptno”, “xyz:emp”, 0, “xyz:dept”)

¹ Inverse function specification for properties not shown, but can be added easily.

No DB: A simple syntax for Constraints

Constraint (ConsName, ConsType, ViewName, RefConsName)

Constraint ("dept:c_unq_deptno", Unique, "xyz:dept", NULL)

Constraint ("emp:c_prm_empno", Primary, "xyz:emp", NULL)

Constraint ("emp:c_ref_deptno", Reference, "xyz:emp", "dept:c_unq_deptno")

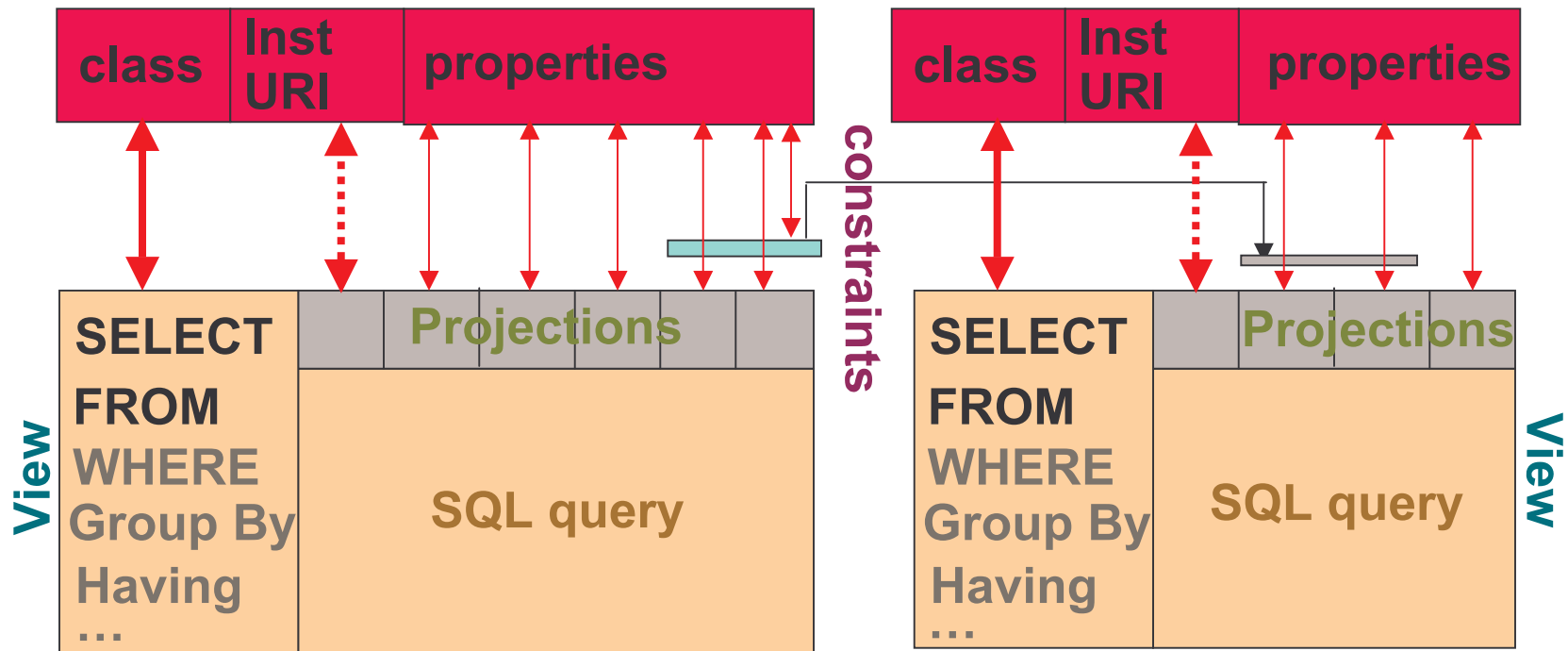
ConstraintColumn (ConsName, ViewName, ColName, ColPosInKey)

ConstraintColumn ("dept:c_unq_deptno", "xyz:dept", "dept:deptno", 1)

ConstraintColumn ("emp:c_prm_empno", "xyz:emp", "emp:empno", 1)

ConstraintColumn ("emp:c_ref_deptno", "xyz:emp", "emp:deptNum", 1)

Summary



- Proposed mapping language
(Note: SQL is *not* a part of the mapping language)
 - Strong-DB → No new language. Employ some *conventions* with SQL (e.g., when naming Views, View cols, and constraints).
 - No-DB → Simple language to specify mapping between RDF classes, properties and SQL queries, query projections, constraints.



ORACLE IS THE INFORMATION COMPANY