Ontology-based Data Access: Theory and Practice

Guohui Xiao

KRDB Research Centre

Free University of Bozen-Bolzano

Roman Kontchakov

Department of Computer Science & Inf. Systems

Birkbeck, University of London

http://ontop.inf.unibz.it/ijcai-2018-tutorial

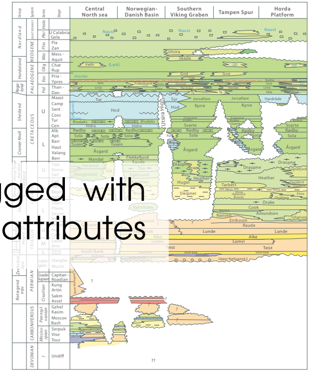
Motivation: Geologists at Statoil (Equinor)

900 geologists & geophysicists in Statoil Exploration develop **stratigraphic models** of unexplored areas on the basis of data acquired from previous operations at nearby locations

Motivation: Geologists at Statoil (Equinor)

900 geologists & geophysicists in Statoil Exploration develop **stratigraphic models** of unexplored areas on the basis of data acquired from previous operations at nearby locations

LITHOSTRATIGRAPHIC CHART
NORWEGIAN NORTH SEA



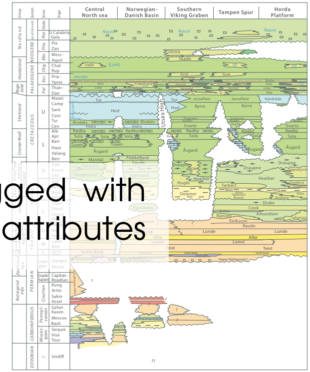
G&G's **information need (009)**

In my area of interest, return all pressure data tagged with key stratigraphy information with understandable QC attributes (and suitable for further filtering).

Motivation: Geologists at Statoil (Equinor)

900 geologists & geophysicists in Statoil Exploration develop
stratigraphic models of unexplored areas on the basis of
data acquired from previous operations at nearby locations

LITHOSTRATIGRAPHIC CHART
NORWEGIAN NORTH SEA



G&G's **information need (009)**

In my area of interest, return all pressure data tagged with key stratigraphy information with understandable QC attributes (and suitable for further filtering).

Slegge database contains **1545** tables and 1727 views

table WELLBORE has 38 columns

formation pressure: a join of PTY_PRESSURE, ACTIVITY, ACTIVITY_CLASS and WELLBORE

stratigraphic information: a join of PICKED_STRATIGRAPHIC_ZONES, PTY_LOCATION_1D, PTY_PRESSURE, ACTIVITY, ACTIVITY_CLASS and WELLBORE

Data Gathering at Statoil (1)

data gathering is a huge problem in industry:

search for data and quality assessment,

e.g., in oil&gas takes **30–70%** of engineers' time *(Crompton, 2008)*

Data Gathering at Statoil (1)

data gathering is a huge problem in industry:

search for data and quality assessment,

e.g., in oil&gas takes **30–70%** of engineers' time *(Crompton, 2008)*

solution 1

use **predefined SQL queries** of the in-house system to retrieve information about

(a) pressure measurements,

(b) lithostratigraphy of wellbores, etc.

and integrate the results using a **spreadsheet**

Data Gathering at Statoil (1)

data gathering is a huge problem in industry:

search for data and quality assessment,

e.g., in oil&gas takes **30–70%** of engineers' time *(Crompton, 2008)*

solution 1

use **predefined SQL queries** of the in-house system to retrieve information about

(a) pressure measurements,

(b) lithostratigraphy of wellbores, etc.

and integrate the results using a **spreadsheet**

time-consuming, **error-prone** (e.g., different units of measurement), **difficult to repeat**

Data Gathering at Statoil (2)

solution 2

ask an **IT expert** to translate the information need into SQL

Data Gathering at Statoil (2)

solution 2

ask an **IT expert** to translate the information need into SQL

```
SELECT
    WELLBORE.IDENTIFIER,
    PTY_PRESSURE.PTY_PRESSURE_S,
    STRATIGRAPHIC_ZONE.STRAT_COLUMN_IDENTIFIER,
    STRATIGRAPHIC_ZONE.STRAT_UNIT_IDENTIFIER
FROM WELLBORE,
    PTY_PRESSURE,
    ACTIVITY FP_DEPTH_DATA
        LEFT JOIN (PTY_LOCATION_1D FP_DEPTH_PT1_LOC
            INNER JOIN PICKED_STRATIGRAPHIC_ZONES ZS
                ON ZS.STRAT_ZONE_ENTRY_MD <= FP_DEPTH_PT1_LOC.DATA_VALUE_1_O AND
                    ZS.STRAT_ZONE_EXIT_MD >= FP_DEPTH_PT1_LOC.DATA_VALUE_1_O AND
                    ZS.STRAT_ZONE_DEPTH_UOM = FP_DEPTH_PT1_LOC.DATA_VALUE_1_OU
            INNER JOIN STRATIGRAPHIC_ZONE
                ON ZS.WELLBORE = STRATIGRAPHIC_ZONE.WELLBORE AND
                    ZS.STRAT_COLUMN_IDENTIFIER = STRATIGRAPHIC_ZONE.STRAT_COLUMN_IDENTIFIER AND
                    ZS.STRAT_INTERP_VERSION = STRATIGRAPHIC_ZONE.STRAT_INTERP_VERSION AND
                    ZS.STRAT_ZONE_IDENTIFIER = STRATIGRAPHIC_ZONE.STRAT_ZONE_IDENTIFIER)
            ON FP_DEPTH_DATA.FACILITY_S = ZS.WELLBORE AND
                FP_DEPTH_DATA.ACTIVITY_S = FP_DEPTH_PT1_LOC.ACTIVITY_S,
    ACTIVITY_CLASS FORM_PRESSURE_CLASS
WHERE WELLBORE.WELLBORE_S = FP_DEPTH_DATA.FACILITY_S AND
    FP_DEPTH_DATA.ACTIVITY_S = PTY_PRESSURE.ACTIVITY_S AND
    FP_DEPTH_DATA.KIND_S = FORM_PRESSURE_CLASS.ACTIVITY_CLASS_S AND
    WELLBORE.REF_EXISTENCE_KIND = 'actual' AND
    FORM_PRESSURE_CLASS.NAME = 'formation pressure depth data'
```

Data Gathering at Statoil (2)

solution 2

ask an **IT expert** to translate the information need into SQL

```
SELECT
    WELLBORE.IDENTIFIER,
    PTY_PRESSURE.PTY_PRESSURE_S,
    STRATIGRAPHIC_ZONE.STRAT_COLUMN_IDENTIFIER,
    STRATIGRAPHIC_ZONE.STRAT_UNIT_IDENTIFIER
FROM WELLBORE,
    PTY_PRESSURE,
    ACTIVITY FP_DEPTH_DATA
        LEFT JOIN (PTY_LOCATION_1D FP_DEPTH_PT1_LOC
            INNER JOIN PICKED_STRATIGRAPHIC_ZONES ZS
                ON ZS.STRAT_ZONE_ENTRY_MD <= FP_DEPTH_PT1_LOC.DATA_VALUE_1_O AND
                    ZS.STRAT_ZONE_EXIT_MD >= FP_DEPTH_PT1_LOC.DATA_VALUE_1_O AND
                    ZS.STRAT_ZONE_DEPTH_UOM = FP_DEPTH_PT1_LOC.DATA_VALUE_1_OU
            INNER JOIN STRATIGRAPHIC_ZONE
                ON ZS.WELLBORE = STRATIGRAPHIC_ZONE.WELLBORE AND
                    ZS.STRAT_COLUMN_IDENTIFIER = STRATIGRAPHIC_ZONE.STRAT_COLUMN_IDENTIFIER AND
                    ZS.STRAT_INTERP_VERSION = STRATIGRAPHIC_ZONE.STRAT_INTERP_VERSION AND
                    ZS.STRAT_ZONE_IDENTIFIER = STRATIGRAPHIC_ZONE.STRAT_ZONE_IDENTIFIER)
            ON FP_DEPTH_DATA.FACILITY_S = ZS.WELLBORE AND
                FP_DEPTH_DATA.ACTIVITY_S = FP_DEPTH_PT1_LOC.ACTIVITY_S,
    ACTIVITY_CLASS FORM_PRESSURE_CLASS
WHERE WELLBORE.WELLBORE_S = FP_DEPTH_DATA.FACILITY_S AND
    FP_DEPTH_DATA.ACTIVITY_S = PTY_PRESSURE.ACTIVITY_S AND
    FP_DEPTH_DATA.KIND_S = FORM_PRESSURE_CLASS.ACTIVITY_CLA
    WELLBORE.REF_EXISTENCE_KIND = 'actual' AND
    FORM_PRESSURE_CLASS.NAME = 'formation pressure depth da
```

takes **4 days**
on average (with Slegge only)

Data Gathering at Statoil (2)

solution 2

ask an **IT expert** to translate the information need into SQL

```
SELECT
    WELLBORE.IDENTIFIER,
    PTY_PRESSURE.PTY_PRESSURE_S,
    STRATIGRAPHIC_ZONE.STRAT_COLUMN_IDENTIFIER
```

knowledge of the **geological domain** and **database structure**
(1545 tables and 1727 views, magic values, extensive denormalisation)

```
            ON ZS.STRAT_ZONE_ENTRY_MD <= FP_DEPTH_PT1_LOC.DATA_VALUE_1_O AND
                ZS.STRAT_ZONE_EXIT_MD >= FP_DEPTH_PT1_LOC.DATA_VALUE_1_O AND
                ZS.STRAT_ZONE_DEPTH_UOM = FP_DEPTH_PT1_LOC.DATA_VALUE_1_OU
        INNER JOIN STRATIGRAPHIC_ZONE
            ON ZS.WELLBORE = STRATIGRAPHIC_ZONE.WELLBORE AND
                ZS.STRAT_COLUMN_IDENTIFIER = STRATIGRAPHIC_ZONE.STRAT_COLUMN_IDENTIFIER AND
                ZS.STRAT_INTERP_VERSION = STRATIGRAPHIC_ZONE.STRAT_INTERP_VERSION AND
                ZS.STRAT_ZONE_IDENTIFIER = STRATIGRAPHIC_ZONE.STRAT_ZONE_IDENTIFIER)
        ON FP_DEPTH_DATA.FACILITY_S = ZS.WELLBORE AND
            FP_DEPTH_DATA.ACTIVITY_S = FP_DEPTH_PT1_LOC.ACTIVITY_S,
    ACTIVITY_CLASS FORM_PRESSURE_CLASS
WHERE WELLBORE.WELLBORE_S = FP_DEPTH_DATA.FACILITY_S AND
    FP_DEPTH_DATA.ACTIVITY_S = PTY_PRESSURE.ACTIVITY_S AND
    FP_DEPTH_DATA.KIND_S = FORM_PRESSURE_CLASS.ACTIVITY_CLA
    WELLBORE.REF_EXISTENCE_KIND = 'actual' AND
    FORM_PRESSURE_CLASS.NAME = 'formation pressure depth da
```

takes **4 days**
on average (with Slegge only)

Data Gathering at Statoil (2)

solution 2

ask an **IT expert** to translate the information need into SQL

```
SELECT
      WELLBORE.IDENTIFIER,
      PTY_PRESSURE.PTY_PRESSURE_S,
      STRATIGRAPHIC_ZONE.STRAT_COLUMN_IDENTIFIER
```

knowledge of the **geological domain** and **database structure**

(1545 tables and 1727 views, magic values, extensive denormalisation)

```
                  ON ZS.STRAT_ZONE_ENTRY_MD <= FP_DEPTH_PT1_LOC.DATA_VALUE_1_O AND
                     ZS.STRAT_ZONE_EXIT_MD >= FP_DEPTH_PT1_LOC.DATA_VALUE_1_O AND
                     ZS.STRAT_ZONE_DEPTH_UOM = FP_DEPTH_PT1_LOC.DATA_VALUE_1_OU
```

encode the domain knowledge in an ontology

```
                     ZS.STRAT_COLUMN_IDENTIFIER = STRATIGRAPHIC_ZONE.STRAT_COLUMN_IDENTIFIER AND
                     ZS.STRAT_INTERP_VERSION = STRATIGRAPHIC_ZONE.STRAT_INTERP_VERSION AND
                     ZS.STRAT_ZONE_IDENTIFIER = STRATIGRAPHIC_ZONE.STRAT_ZONE_IDENTIFIER)
            ON FP_DEPTH_DATA.FACILITY_S = ZS.WELLBORE AND
                  FP_DEPTH_DATA.ACTIVITY_S = FP_DEPTH_PT1_LOC.ACTIVITY_S,
      ACTIVITY_CLASS FORM_PRESSURE_CLASS
WHERE WELLBORE.WELLBORE_S = FP_DEPTH_DATA.FACILITY_S AND
      FP_DEPTH_DATA.ACTIVITY_S = PTY_PRESSURE.ACTIVITY_S AND
      FP_DEPTH_DATA.KIND_S = FORM_PRESSURE_CLASS.ACTIVITY_CLA
      WELLBORE.REF_EXISTENCE_KIND = 'actual' AND
      FORM_PRESSURE_CLASS.NAME = 'formation pressure depth data
```

takes **4 days**
on average (with Slegge only)

Data Gathering at Statoil (2)

solution 2

ask an **IT expert** to translate the information need into SQL

```
SELECT
    WELLBORE.IDENTIFIER,
    PTY_PRESSURE.PTY_PRESSURE_S,
    STRATIGRAPHIC_ZONE.STRAT_COLUMN_IDENTIFIER
```

knowledge of the **geological domain** and **database structure**
(1545 tables and 1727 views, magic values, extensive denormalisation)

```
            ON ZS.STRAT_ZONE_ENTRY_MD <= FP_DEPTH_PT1_LOC.DATA_VALUE_1_O AND
               ZS.STRAT_ZONE_EXIT_MD >= FP_DEPTH_PT1_LOC.DATA_VALUE_1_O AND
```

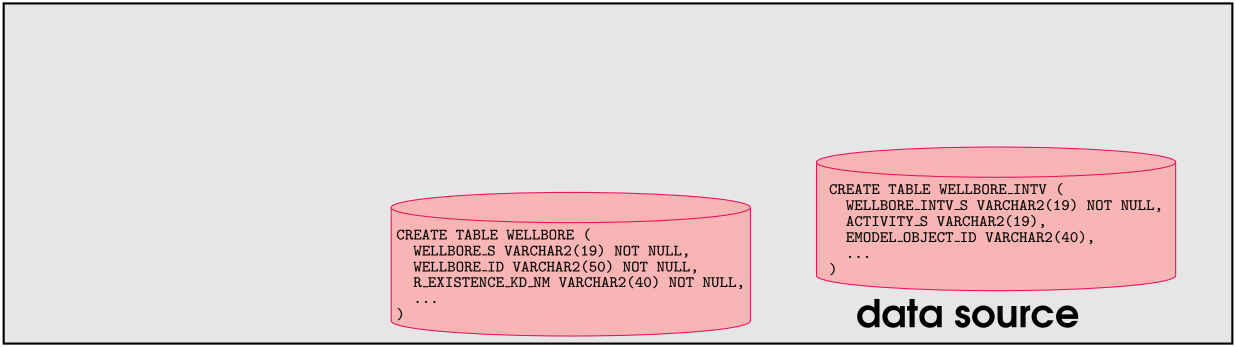
encode the domain knowledge in an ontology

and database structure in mappings

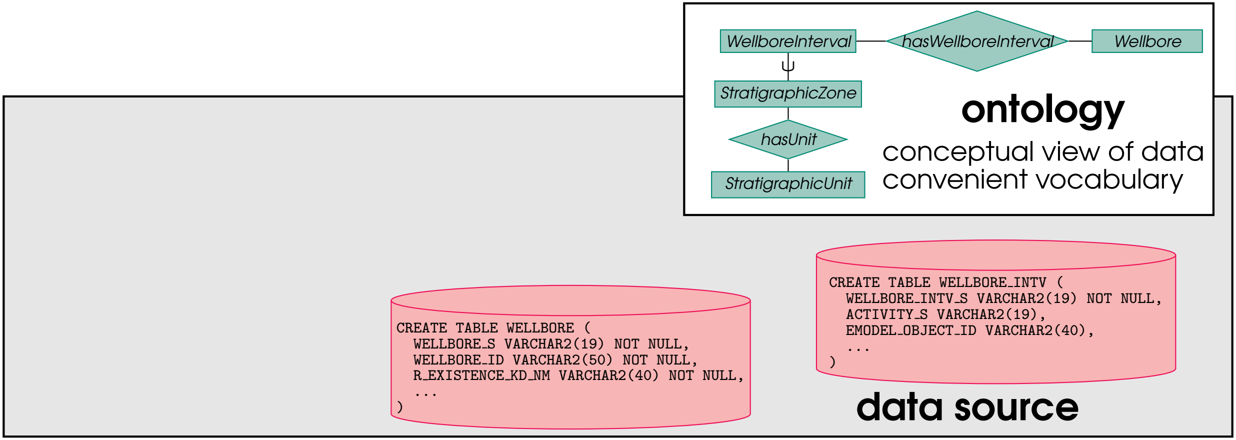
```
         ON
            FP_DEPTH_DATA.ACTIVITY_S = FP_DEPTH_PT1_LOC.ACTIVITY_S,
    ACTIVITY_CLASS FORM_PRESSURE_CLASS
WHERE WELLBORE.WELLBORE_S = FP_DEPTH_DATA.FACILITY_S AND
    FP_DEPTH_DATA.ACTIVITY_S = PTY_PRESSURE.ACTIVITY_S AND
    FP_DEPTH_DATA.KIND_S = FORM_PRESSURE_CLASS.ACTIVITY_CLA
    WELLBORE.REF_EXISTENCE_KIND = 'actual' AND
    FORM_PRESSURE_CLASS.NAME = 'formation pressure depth da
```

takes **4 days**
on average (with Slegge only)

Ontology-Based Data Access



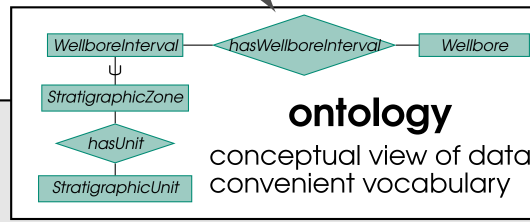
Ontology-Based Data Access



Ontology-Based Data Access

```
SELECT ?w ?depth ?strat_unit
WHERE {
    ?w a :Wellbore . ?w :hasMeasurement ?p .
    ?p a :Pressure . ?p :hasDepth ?depth
    OPTIONAL {
        ?depth :inWellboreInterval ?strat_zone .
        ?strat_zone :hasUnit ?strat_unit
    }
}
```

SPARQL query

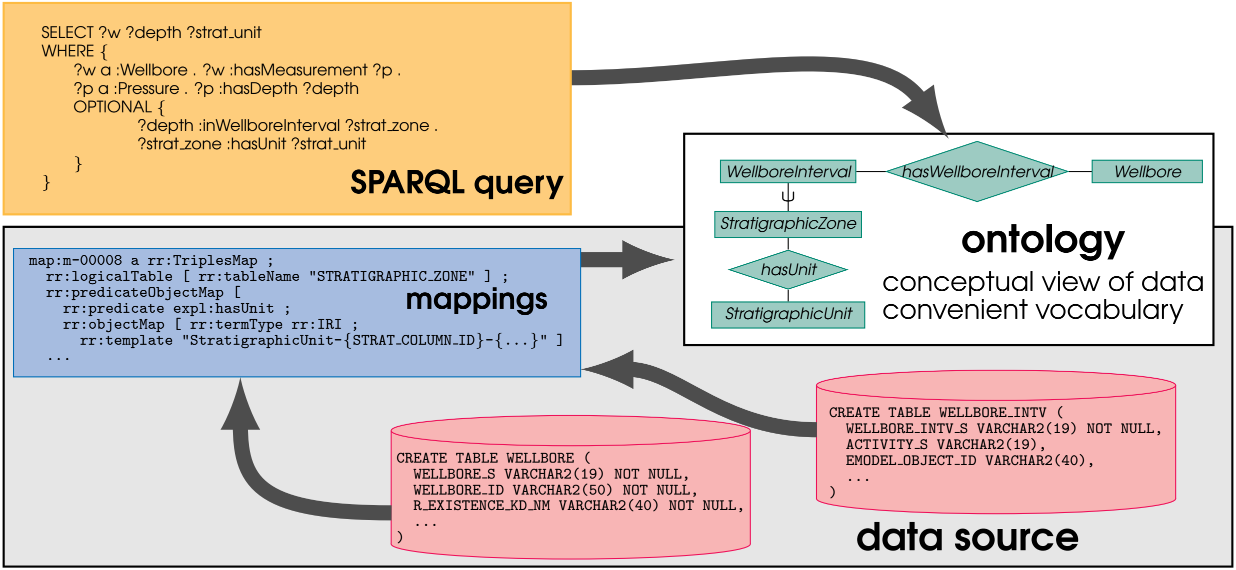



```
CREATE TABLE WELLBORE (
  WELLBORE_S VARCHAR2(19) NOT NULL,
  WELLBORE_ID VARCHAR2(50) NOT NULL,
  R_EXISTENCE_KD_NM VARCHAR2(40) NOT NULL,
  ...
)
```

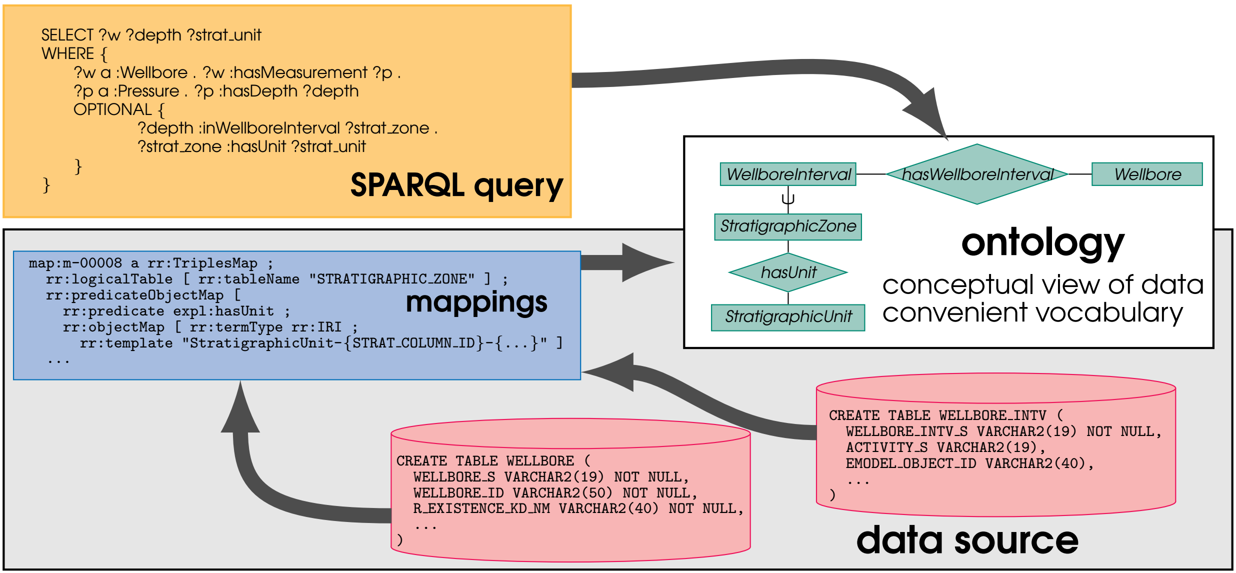
```
CREATE TABLE WELLBORE_INTV (
  WELLBORE_INTV_S VARCHAR2(19) NOT NULL,
  ACTIVITY_S VARCHAR2(19),
  EMODEL_OBJECT_ID VARCHAR2(40),
  ...
)
```

data source

Ontology-Based Data Access



Ontology-Based Data Access



reduced time for translating information needs into (SPARQL) queries

days ⟶ minutes

Tutorial Plan

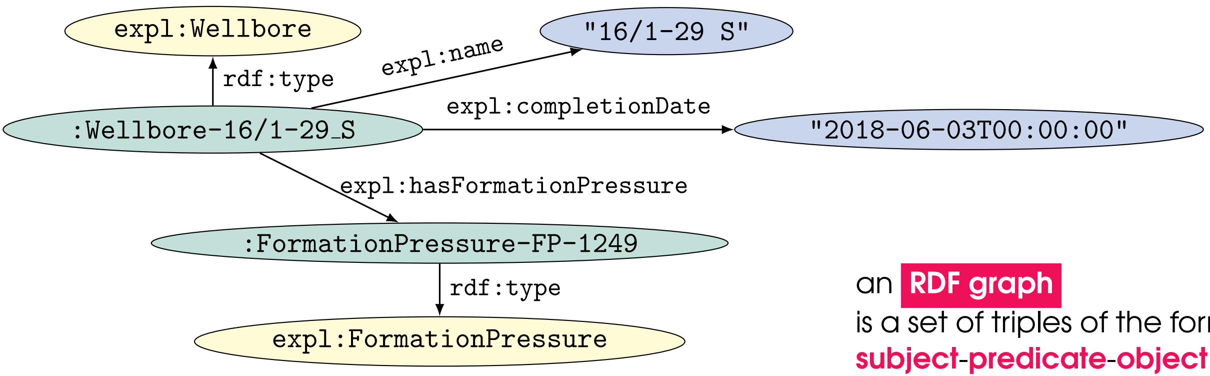
- Semantic Web standards
  - RDF (Data Model)
  - OWL 2 QL (Ontology Language)
  - SPARQL (Query Language)
  - R2RML (Mapping Language)
- Tutorial with Ontop
- coffee break
- Using OBDA in Practice
- Basics: Query Rewriting and Optimisation
- Extending the Foundations
  - Approximating Expressive Ontologies
  - Dealing with Identity: SameAs
  - Ontology-Mediated Query Answering and Circuit Complexity
- Recent Advances / Challenges

RDF: Resource Description Framework (RDF in 2004, RDF 1.1 in 2014)

IRIs are identifiers: $\underbrace{\texttt{slegge:}}_{\textbf{prefix}\ \texttt{http://slegger.gitlab.io/data\#}}\texttt{Wellbore-16/1-29\_S}$

RDF: Resource Description Framework (RDF in 2004, RDF 1.1 in 2014)

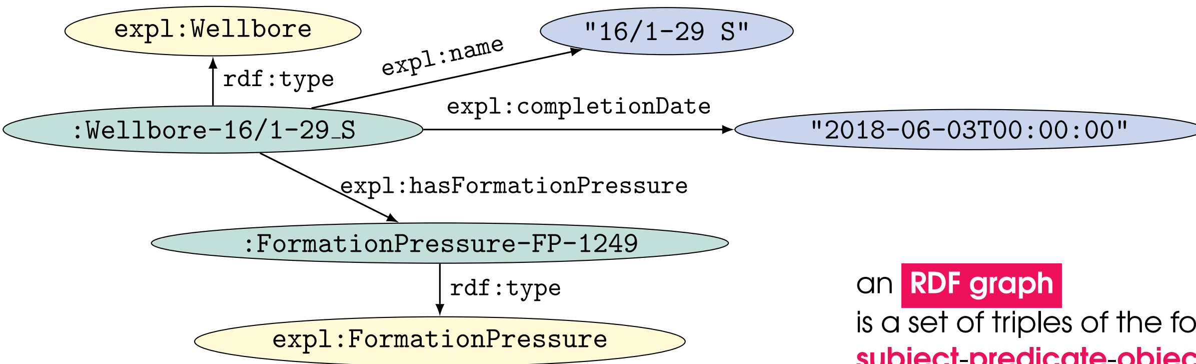
IRIs are identifiers: $\underbrace{\texttt{slegge:}}_{\textbf{prefix}\ \texttt{http://slegger.gitlab.io/data\#}}\texttt{Wellbore-16/1-29\_S}$



an **RDF graph**
is a set of triples of the form
subject-predicate-object

RDF: Resource Description Framework (RDF in 2004, RDF 1.1 in 2014)

IRIs are identifiers: $\underbrace{\texttt{slegge:}}_{\textbf{prefix}\ \texttt{http://slegger.gitlab.io/data\#}}\texttt{Wellbore-16/1-29\_S}$



an **RDF graph**
is a set of triples of the form
subject-predicate-object

```
@prefix :  <http://slegger.gitlab.io/data#> .
@prefix expl:  <http://slegger.gitlab.io/slegge-obda/ontology/subsurface-exploration#> .

:Wellbore-16/1-29_S rdf:type expl:Wellbore .
:Wellbore-16/1-29_S expl:name "16/1-29 S" .
:Wellbore-16/1-29_S expl:completionDate "2018-06-03T00:00:00"^^xsd:dateTime .
:Wellbore-16/1-29_S expl:hasFormationPressure :FormationPressure-FP-1249 .
:FormationPressure-FP-1249 rdf:type expl:FormationPressure .
```

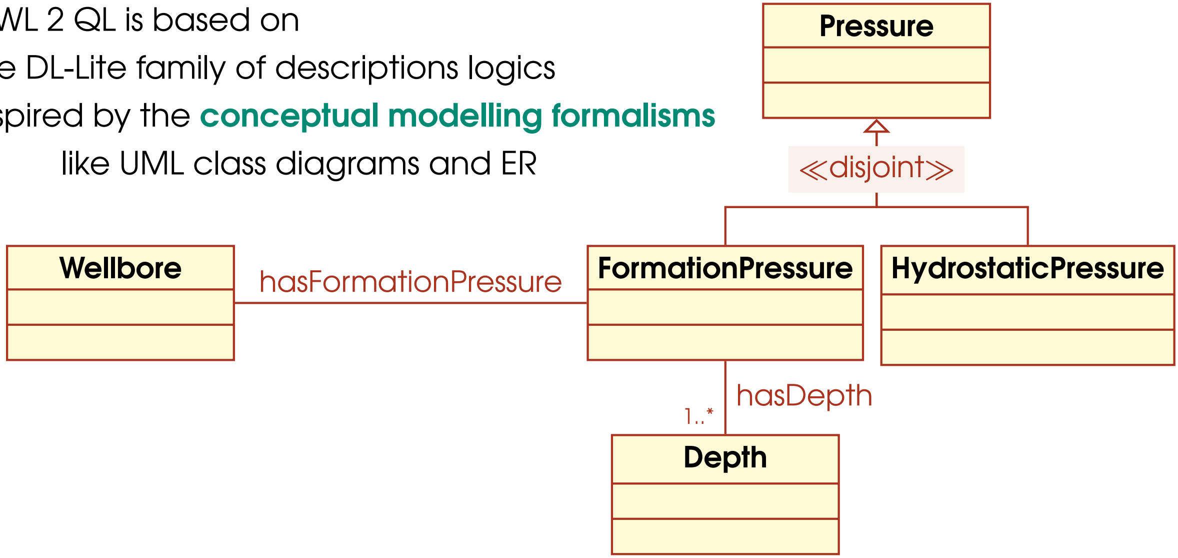
Turtle syntax

OWL 2 QL

OWL 2 QL is based on
the DL-Lite family of descriptions logics
inspired by the **conceptual modelling formalisms**
like UML class diagrams and ER

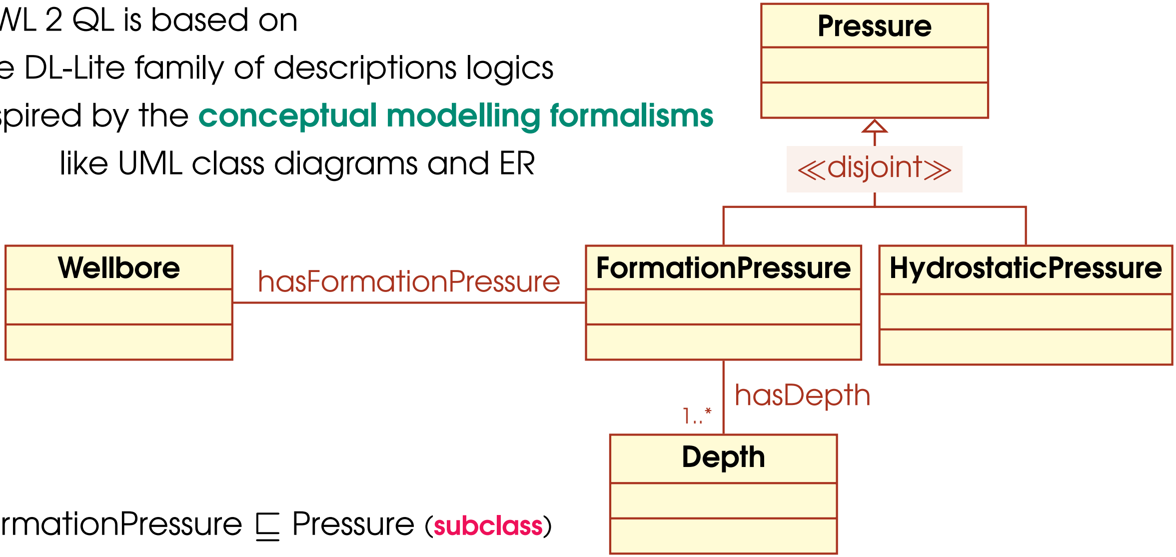
OWL 2 QL

OWL 2 QL is based on
the DL-Lite family of descriptions logics
inspired by the **conceptual modelling formalisms**
like UML class diagrams and ER



OWL 2 QL

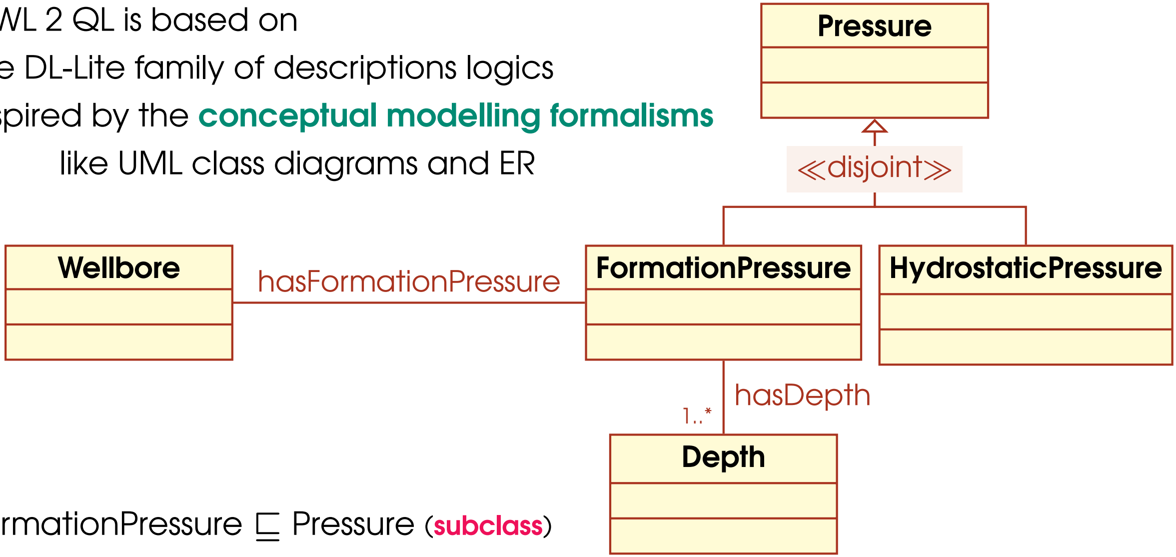
OWL 2 QL is based on
the DL-Lite family of descriptions logics
inspired by the **conceptual modelling formalisms**
like UML class diagrams and ER



FormationPressure $\sqsubseteq$ Pressure (**subclass**)

OWL 2 QL

OWL 2 QL is based on
the DL-Lite family of descriptions logics
inspired by the **conceptual modelling formalisms**
like UML class diagrams and ER

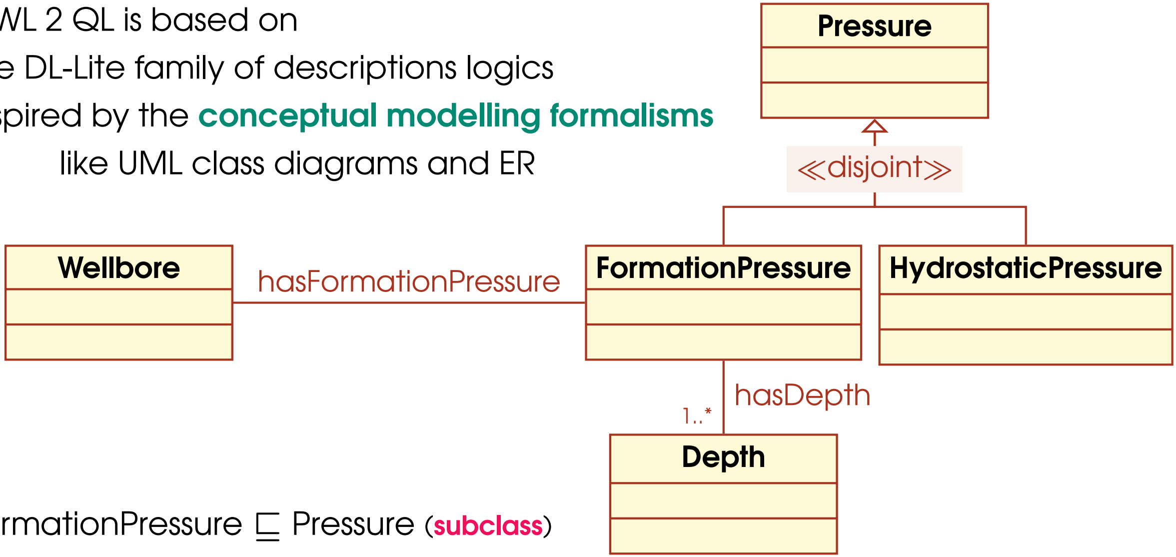


FormationPressure $\sqsubseteq$ Pressure (**subclass**)

FormationPressure $\sqcap$ HydrostaticPressure $\sqsubseteq$ $\bot$ (**disjointness**)

OWL 2 QL

OWL 2 QL is based on
the DL-Lite family of descriptions logics
inspired by the **conceptual modelling formalisms**
like UML class diagrams and ER



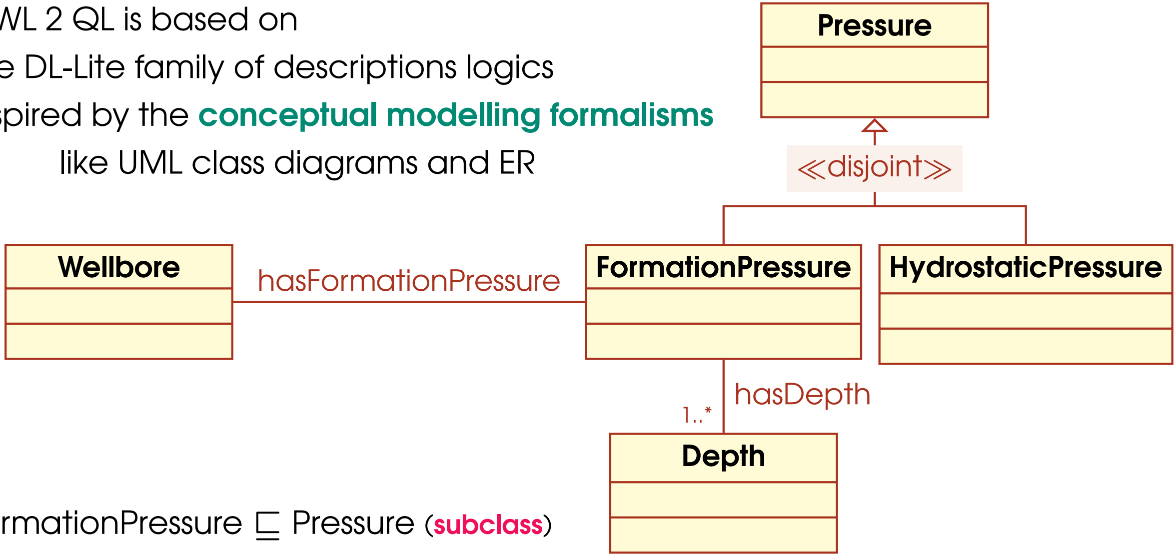
FormationPressure $\sqsubseteq$ Pressure (**subclass**)

FormationPressure $\sqcap$ HydrostaticPressure $\sqsubseteq \bot$ (**disjointness**)

hasFormationPressure $\sqsubseteq$ hasMeasurement (**sub-association**)

OWL 2 QL

OWL 2 QL is based on
the DL-Lite family of descriptions logics
inspired by the **conceptual modelling formalisms**
like UML class diagrams and ER



FormationPressure $\sqsubseteq$ Pressure (**subclass**)

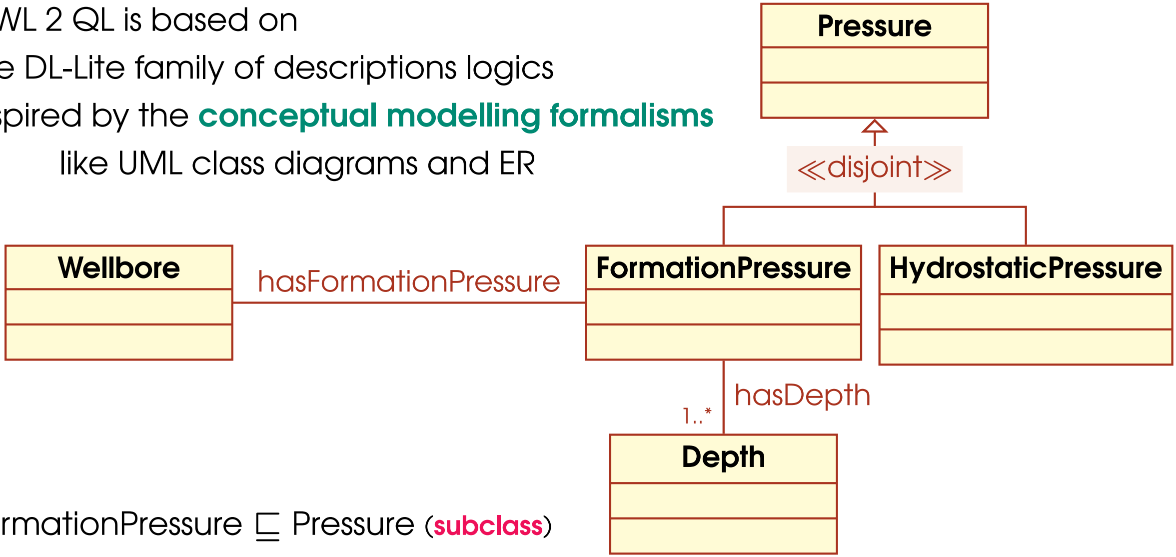
FormationPressure $\sqcap$ HydrostaticPressure $\sqsubseteq \bot$ (**disjointness**)

hasFormationPressure $\sqsubseteq$ hasMeasurement (**sub-association**)

$\exists$hasFormationPressure$^-$ $\sqsubseteq$ FormationPressure (**range/domain**)

OWL 2 QL

OWL 2 QL is based on
the DL-Lite family of descriptions logics
inspired by the **conceptual modelling formalisms**
like UML class diagrams and ER



$\mathrm{FormationPressure} \sqsubseteq \mathrm{Pressure}$ (**subclass**)

$\mathrm{FormationPressure} \sqcap \mathrm{HydrostaticPressure} \sqsubseteq \bot$ (**disjointness**)

$\mathrm{hasFormationPressure} \sqsubseteq \mathrm{hasMeasurement}$ (**sub-association**)

$\exists \mathrm{hasFormationPressure}^- \sqsubseteq \mathrm{FormationPressure}$ (**range/domain**)

$\mathrm{FormationPressure} \sqsubseteq \exists \mathrm{hasDepth}.\mathrm{Depth}$ (**mandatory participation/multiplicity**)

Semantics of OWL 2 QL

DLs are **fragments of First-Order Logic**:

classes are **unary predicates** / properties are **binary predicates**

Semantics of OWL 2 QL

DLs are **fragments of First-Order Logic**:

classes are **unary predicates** / properties are **binary predicates**

FormationPressure $\sqsubseteq$ Pressure

Semantics of OWL 2 QL

DLs are **fragments of First-Order Logic**:

classes are **unary predicates** / properties are **binary predicates**

FormationPressure $\sqsubseteq$ Pressure

$$\forall x\ (\text{FormationPressure}(x) \to \text{Pressure}(x))$$

Semantics of OWL 2 QL

DLs are **fragments of First-Order Logic**:

classes are **unary predicates** / properties are **binary predicates**

FormationPressure $\sqsubseteq$ Pressure

$$\forall x\ (\text{FormationPressure}(x) \to \text{Pressure}(x))$$

FormationPressure $\sqcap$ HydrostaticPressure $\sqsubseteq \bot$

Semantics of OWL 2 QL

DLs are **fragments of First-Order Logic**:

classes are **unary predicates** / properties are **binary predicates**

FormationPressure $\sqsubseteq$ Pressure

$$\forall x\ (\mathsf{FormationPressure}(x) \to \mathsf{Pressure}(x))$$

FormationPressure $\sqcap$ HydrostaticPressure $\sqsubseteq \bot$

$$\forall x\ (\mathsf{FormationPressure}(x) \land \mathsf{HydrostaticPressure}(x) \to \bot)$$

Semantics of OWL 2 QL

DLs are **fragments of First-Order Logic**:

classes are **unary predicates** / properties are **binary predicates**

FormationPressure $\sqsubseteq$ Pressure

$$\forall x\ (\mathsf{FormationPressure}(x) \to \mathsf{Pressure}(x))$$

FormationPressure $\sqcap$ HydrostaticPressure $\sqsubseteq \bot$

$$\forall x\ (\mathsf{FormationPressure}(x) \land \mathsf{HydrostaticPressure}(x) \to \bot)$$

hasFormationPressure $\sqsubseteq$ hasMeasurement

Semantics of OWL 2 QL

DLs are **fragments of First-Order Logic**:

classes are **unary predicates** / properties are **binary predicates**

FormationPressure $\sqsubseteq$ Pressure

$$\forall x\ (\text{FormationPressure}(x) \to \text{Pressure}(x))$$

FormationPressure $\sqcap$ HydrostaticPressure $\sqsubseteq \bot$

$$\forall x\ (\text{FormationPressure}(x) \land \text{HydrostaticPressure}(x) \to \bot)$$

hasFormationPressure $\sqsubseteq$ hasMeasurement

$$\forall xy\ (\text{hasFormationPressure}(x, y) \to \text{hasMeasurement}(x, y))$$

Semantics of OWL 2 QL

DLs are **fragments of First-Order Logic**:

classes are **unary predicates** / properties are **binary predicates**

FormationPressure $\sqsubseteq$ Pressure

$$\forall x\ (\text{FormationPressure}(x) \rightarrow \text{Pressure}(x))$$

FormationPressure $\sqcap$ HydrostaticPressure $\sqsubseteq \bot$

$$\forall x\ (\text{FormationPressure}(x) \land \text{HydrostaticPressure}(x) \rightarrow \bot)$$

hasFormationPressure $\sqsubseteq$ hasMeasurement

$$\forall xy\ (\text{hasFormationPressure}(x, y) \rightarrow \text{hasMeasurement}(x, y))$$

$\exists$hasFormationPressure$^-$ $\sqsubseteq$ FormationPressure

Semantics of OWL 2 QL

DLs are **fragments of First-Order Logic**:

classes are **unary predicates** / properties are **binary predicates**

FormationPressure $\sqsubseteq$ Pressure

$$\forall x\ (\mathsf{FormationPressure}(x) \to \mathsf{Pressure}(x))$$

FormationPressure $\sqcap$ HydrostaticPressure $\sqsubseteq \bot$

$$\forall x\ (\mathsf{FormationPressure}(x) \land \mathsf{HydrostaticPressure}(x) \to \bot)$$

hasFormationPressure $\sqsubseteq$ hasMeasurement

$$\forall xy\ (\mathsf{hasFormationPressure}(x, y) \to \mathsf{hasMeasurement}(x, y))$$

$\exists$hasFormationPressure$^-$ $\sqsubseteq$ FormationPressure

$$\forall xy\ (\mathsf{hasFormationPressure}(x, y) \to \mathsf{FormationPressure}(y))$$

Semantics of OWL 2 QL

DLs are **fragments of First-Order Logic**:

classes are **unary predicates** / properties are **binary predicates**

FormationPressure $\sqsubseteq$ Pressure

$$\forall x\ (\text{FormationPressure}(x) \rightarrow \text{Pressure}(x))$$

FormationPressure $\sqcap$ HydrostaticPressure $\sqsubseteq$ $\perp$

$$\forall x\ (\text{FormationPressure}(x) \wedge \text{HydrostaticPressure}(x) \rightarrow \perp)$$

hasFormationPressure $\sqsubseteq$ hasMeasurement

$$\forall xy\ (\text{hasFormationPressure}(x, y) \rightarrow \text{hasMeasurement}(x, y))$$

$\exists$hasFormationPressure$^-$ $\sqsubseteq$ FormationPressure

$$\forall xy\ (\text{hasFormationPressure}(x, y) \rightarrow \text{FormationPressure}(y))$$

FormationPressure $\sqsubseteq$ $\exists$hasDepth.Depth

Semantics of OWL 2 QL

DLs are **fragments of First-Order Logic**:

classes are **unary predicates** / properties are **binary predicates**

FormationPressure $\sqsubseteq$ Pressure

$$\forall x\ (\mathsf{FormationPressure}(x) \to \mathsf{Pressure}(x))$$

FormationPressure $\sqcap$ HydrostaticPressure $\sqsubseteq \bot$

$$\forall x\ (\mathsf{FormationPressure}(x) \land \mathsf{HydrostaticPressure}(x) \to \bot)$$

hasFormationPressure $\sqsubseteq$ hasMeasurement

$$\forall xy\ (\mathsf{hasFormationPressure}(x, y) \to \mathsf{hasMeasurement}(x, y))$$

$\exists$hasFormationPressure^{-} $\sqsubseteq$ FormationPressure

$$\forall xy\ (\mathsf{hasFormationPressure}(x, y) \to \mathsf{FormationPressure}(y))$$

FormationPressure $\sqsubseteq$ $\exists$hasDepth.Depth

$$\forall x\ (\mathsf{FormationPressure}(x) \to \exists y\ (\mathsf{hasDepth}(x, y) \land \mathsf{Depth}(y)))$$

Semantics of OWL 2 QL

DLs are **fragments of First-Order Logic**:

classes are **unary predicates** / properties are **binary predicates**

FormationPressure $\sqsubseteq$ Pressure

$$\forall x\,(\mathsf{FormationPressure}(x) \to \mathsf{Pressure}(x))$$

FormationPressure $\sqcap$ HydrostaticPressure $\sqsubseteq \bot$

$$\forall x\,(\mathsf{FormationPressure}(x) \land \mathsf{HydrostaticPressure}(x) \to \bot)$$

hasFormationPressure $\sqsubseteq$ hasMeasurement

$$\forall xy\,(\mathsf{hasFormationPressure}(x,y) \to \mathsf{hasMeasurement}(x,y))$$

$\exists$hasFormationPressure$^-$ $\sqsubseteq$ FormationPressure

$$\forall xy\,(\mathsf{hasFormationPressure}(x,y) \to \mathsf{FormationPressure}(y))$$

FormationPressure $\sqsubseteq$ $\exists$hasDepth.Depth

$$\forall x\,(\mathsf{FormationPressure}(x) \to \exists y\,(\mathsf{hasDepth}(x,y) \land \mathsf{Depth}(y)))$$

tuple-generating dependencies / existential rules

OWL

Web Ontology Language (OWL) was standardised in 2004; OWL 2 in 2012

OWL 2 QL is one of the three **profiles** of OWL 2

(fragments tailored for specific applications)

OWL 2 QL contains

- subclass axioms, where

  existential quantification (ObjectSomeValuesFrom) is limited to **owl:Thing**

OWL

Web Ontology Language (OWL) was standardised in 2004; OWL 2 in 2012

OWL 2 QL is one of the three **profiles** of OWL 2
(fragments tailored for specific applications)

OWL 2 QL contains

- subclass axioms, where
  existential quantification (ObjectSomeValuesFrom) is limited to **owl:Thing**
- **inverse** object properties (InverseObjectProperties)

OWL

Web Ontology Language (OWL) was standardised in 2004; OWL 2 in 2012

OWL 2 QL is one of the three **profiles** of OWL 2

(fragments tailored for specific applications)

OWL 2 QL contains

- subclass axioms, where
  existential quantification (ObjectSomeValuesFrom) is limited to **owl:Thing**
- **inverse** object properties (InverseObjectProperties)
- property domain axioms (ObjectPropertyDomain and DataPropertyDomain)
- property range axioms (ObjectPropertyRange and DataPropertyRange)

OWL

Web Ontology Language (OWL) was standardised in 2004; OWL 2 in 2012

OWL 2 QL is one of the three **profiles** of OWL 2

(fragments tailored for specific applications)

OWL 2 QL contains

- subclass axioms, where
  existential quantification (ObjectSomeValuesFrom) is limited to **owl:Thing**
- **inverse** object properties (InverseObjectProperties)
- property domain axioms (ObjectPropertyDomain and DataPropertyDomain)
- property range axioms (ObjectPropertyRange and DataPropertyRange)
- **property inclusions** (SubObjectPropertyOf but no property chains)

OWL

Web Ontology Language (OWL) was standardised in 2004; OWL 2 in 2012

OWL 2 QL is one of the three **profiles** of OWL 2
(fragments tailored for specific applications)

OWL 2 QL contains

- subclass axioms, where
  existential quantification (ObjectSomeValuesFrom) is limited to **owl:Thing**
- **inverse** object properties (InverseObjectProperties)
- property domain axioms (ObjectPropertyDomain and DataPropertyDomain)
- property range axioms (ObjectPropertyRange and DataPropertyRange)
- **property inclusions** (SubObjectPropertyOf but no property chains)

does not contain

- **universal quantification** (ObjectAllValuesFrom, DataAllValuesFrom)
- **enumeration** of individuals and literals (ObjectOneOf, DataOneOf)
- **disjunction** (ObjectUnionOf, DisjointUnion and DataUnionOf)
- individual equality assertions (**sameAs**) (SameIndividual)

SPARQL: Basic Graph Patterns

SPARQL is the query language for RDF (SPARQL 1.0 in 2008, SPARQL 1.1 in 2013)

```
SELECT ?w ?d
WHERE {
    ?w a expl:Wellbore .
    ?w expl:hasMeasurement ?p .
    ?p a expl:Pressure .
    ?p expl:hasDepth ?d
}
```

SPARQL: Basic Graph Patterns

SPARQL is the query language for RDF (SPARQL 1.0 in 2008, SPARQL 1.1 in 2013)

```
SELECT ?w ?d
WHERE {
    ?w a expl:Wellbore .
    ?w expl:hasMeasurement ?p .
    ?p a expl:Pressure .
    ?p expl:hasDepth ?d
}
```

variables to appear in the **query results**

triple patterns separated by .

a abbreviates **rdf:type**

'find all depths of pressure measurements for all wellbores'

```
:Wellbore-16/1-29_S a expl:Wellbore .
:Wellbore-30/8-5 a expl:Wellbore .
:Wellbore-16/1-29_S expl:hasMeasurement :FormationPressure-FP-1249 .
:FormationPressure-FP-1249 a expl:Pressure .
:FormationPressure-FP-1249 expl:hasDepth :TVD-FP-1249 .
:Wellbore-16/1-29_S expl:hasMeasurement :FormationPressure-FP-1377 .
:FormationPressure-FP-1377 a expl:Pressure .
:FormationPressure-FP-1377 expl:hasDepth :TVD-FP-1377 .
```

SPARQL: Basic Graph Patterns

SPARQL is the query language for RDF (SPARQL 1.0 in 2008, SPARQL 1.1 in 2013)

```
SELECT ?w ?d
WHERE {
    ?w a expl:Wellbore .
    ?w expl:hasMeasurement ?p .
    ?p a expl:Pressure .
    ?p expl:hasDepth ?d
}
```

variables to appear in the **query results**

triple patterns separated by .

a abbreviates **rdf:type**

'find all depths of pressure measurements for all wellbores'

```
:Wellbore-16/1-29_S a expl:Wellbore .
:Wellbore-30/8-5 a expl:Wellbore .
:Wellbore-16/1-29_S expl:hasMeasurement :FormationPressure-FP-1249 .
:FormationPressure-FP-1249 a expl:Pressure .
:FormationPressure-FP-1249 expl:hasDepth :TVD-FP-1249 .
:Wellbore-16/1-29_S expl:hasMeasurement :FormationPressure-FP-1377 .
:FormationPressure-FP-1377 a expl:Pressure .
:FormationPressure-FP-1377 expl:hasDepth :TVD-FP-1377 .
```

answer:

?w	?d
:Wellbore-16/1-29_S	:TVD-FP-1249
:Wellbore-16/1-29_S	:TVD-FP-1377

SPARQL: Basic Graph Patterns

SPARQL is the query language for RDF (SPARQL 1.0 in 2008, SPARQL 1.1 in 2013)

```
SELECT ?w ?d
WHERE {
    ?w a expl:Wellbore .
    ?w expl:hasMeasurement ?p .
    ?p a expl:Pressure .
    ?p expl:hasDepth ?d
}
```

variables to appear in the **query results**

triple patterns separated by .

a abbreviates **rdf:type**

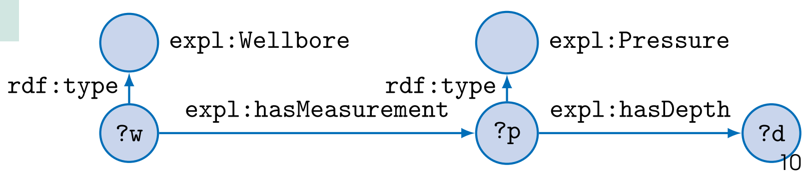
'find all depths of pressure measurements for all wellbores'

```
:Wellbore-16/1-29_S a expl:Wellbore .
:Wellbore-30/8-5 a expl:Wellbore .
:Wellbore-16/1-29_S expl:hasMeasurement :FormationPressure-FP-1249 .
:FormationPressure-FP-1249 a expl:Pressure .
:FormationPressure-FP-1249 expl:hasDepth :TVD-FP-1249 .
:Wellbore-16/1-29_S expl:hasMeasurement :FormationPressure-FP-1377 .
:FormationPressure-FP-1377 a expl:Pressure .
:FormationPressure-FP-1377 expl:hasDepth :TVD-FP-1377 .
```

answer:

?w	?d
:Wellbore-16/1-29_S	:TVD-FP-1249
:Wellbore-16/1-29_S	:TVD-FP-1377

graph matching



SPARQL: Union

```
SELECT ?w
WHERE {
    { ?w a expl:Wellbore }
    UNION
    { ?w a expl:ExplorationWellbore }
}
```

SPARQL: Union

```
SELECT ?w
WHERE {
    { ?w a expl:Wellbore }
    UNION
    { ?w a expl:ExplorationWellbore }
}
```

matches one of **alternative** graph patterns

SPARQL: Union

```
SELECT ?w
WHERE {
    { ?w a expl:Wellbore }
    UNION
    { ?w a expl:ExplorationWellbore }
}
```

matches one of **alternative** graph patterns

```
:Wellbore-16/1-29_S a expl:ExplorationWellbore .
:Wellbore-1/2-U-3 a expl:Wellbore .
```

SPARQL: Union

```
SELECT ?w
WHERE {
    { ?w a expl:Wellbore }
    UNION
    { ?w a expl:ExplorationWellbore }
}
```

matches one of **alternative** graph patterns

```
:Wellbore-16/1-29_S a expl:ExplorationWellbore .
:Wellbore-1/2-U-3 a expl:Wellbore .
```

answer:

?w
:Wellbore-16/1-29_S
:Wellbore-1/2-U-3

SPARQL: Union

```
SELECT ?w
WHERE {
    { ?w a expl:Wellbore }
    UNION
    { ?w a expl:ExplorationWellbore }
}
```

matches one of **alternative** graph patterns

```
:Wellbore-16/1-29_S a expl:ExplorationWellbore .
:Wellbore-1/2-U-3 a expl:Wellbore .
```

answer:

?w
:Wellbore-16/1-29_S
:Wellbore-1/2-U-3

SELECT with Basic Graph Patterns = Conjunctive Queries (CQs)

$$\exists p \, [\text{Wellbore}(w) \land \text{hasMeasurement}(w, p) \land \text{Pressure}(p) \land \text{hasDepth}(p, d)]$$

SELECT with Basic Graph Patterns + UNION $\approx$ Unions of Conjunctive Queries (UCQs)

$$\text{ExplorationWellbore}(w) \lor \text{Wellbore}(w)$$

SPARQL: Union

```
SELECT ?w
WHERE {
    { ?w a expl:Wellbore }
    UNION
    { ?w a expl:ExplorationWellbore }
}
```

matches one of **alternative** graph patterns

```
:Wellbore-16/1-29_S a expl:ExplorationWellbore .
:Wellbore-1/2-U-3 a expl:Wellbore .
```

answer:

?w
:Wellbore-16/1-29_S
:Wellbore-1/2-U-3

SELECT with Basic Graph Patterns = Conjunctive Queries (CQs)

$\exists \boldsymbol{p}\,[\mathsf{Wellbore}(\boldsymbol{w}) \land \mathsf{hasMeasurement}(\boldsymbol{w},\boldsymbol{p}) \land \mathsf{Pressure}(\boldsymbol{p}) \land \mathsf{hasDepth}(\boldsymbol{p},\boldsymbol{d})]$

SELECT with Basic Graph Patterns + UNION ≈ Unions of Conjunctive Queries (UCQs)

$\mathsf{ExplorationWellbore}(\boldsymbol{w}) \lor \mathsf{Wellbore}(\boldsymbol{w})$

observe that `expl:ExplorationWellbore` is naturally a **subclass** of `expl:Wellbore`

why do we need the UNION above then?

Entailment Regime

OWL 2 Direct Semantics Entailment Regime is part of SPARQL 1.1 recommendation

it allows to take account of the **ontology** (subclass & subproperty axioms, etc.) when **answering** SPARQL queries

Entailment Regime

OWL 2 Direct Semantics Entailment Regime is part of SPARQL 1.1 recommendation

it allows to take account of the **ontology** (subclass & subproperty axioms, etc.) when **answering** SPARQL queries

if expl:ExplorationWellbore $\sqsubseteq$ expl:Wellbore, then

```
SELECT DISTINCT ?w
WHERE {
    ?w a expl:Wellbore
}
```

is equivalent (under the entailment regime) to

```
SELECT DISTINCT ?w
WHERE {
    { ?w a expl:Wellbore }
    UNION
    { ?w a expl:ExplorationWellbore }
}
```

Entailment Regime

OWL 2 Direct Semantics Entailment Regime is part of SPARQL 1.1 recommendation

it allows to take account of the **ontology** (subclass & subproperty axioms, etc.) when **answering** SPARQL queries

if expl:ExplorationWellbore $\sqsubseteq$ expl:Wellbore , then

```
SELECT DISTINCT ?w
WHERE {
    ?w a expl:Wellbore
}
```

is equivalent (under the entailment regime) to

```
SELECT DISTINCT ?w
WHERE {
    { ?w a expl:Wellbore }
    UNION
    { ?w a expl:ExplorationWellbore }
}
```

more precisely, when evaluating basic graph patterns (BGPs), graph matching is replaced by **'entailed by the ontology'**

the semantics of all other constructs of SPARQL is the same

SPARQL: Optional Matching

```
SELECT ?w ?d ?u
WHERE {
    ?w a expl:Wellbore . ?w expl:hasMeasurement ?p .
    ?p a expl:Pressure . ?p expl:hasDepth ?d
    OPTIONAL {
        ?d expl:inWellboreInterval ?z .
        ?z expl:hasUnit ?u
    }
}
```

SPARQL: Optional Matching

```
SELECT ?w ?d ?u
WHERE {
    ?w a expl:Wellbore . ?w expl:hasMeasurement ?p .
    ?p a expl:Pressure . ?p expl:hasDepth ?d
    OPTIONAL {
        ?d expl:inWellboreInterval ?z .
        ?z expl:hasUnit ?u
    }
}
```

if the optional part does not match,
it creates **no bindings**
but **does not eliminate** the solution

SPARQL: Optional Matching

```
SELECT ?w ?d ?u
WHERE {
    ?w a expl:Wellbore . ?w expl:hasMeasurement ?p .
    ?p a expl:Pressure . ?p expl:hasDepth ?d
    OPTIONAL {
        ?d expl:inWellboreInterval ?z .
        ?z expl:hasUnit ?u
    }
}
```

if the optional part does not match, it creates **no bindings** but **does not eliminate** the solution

```
:Wellbore-16/1-29_S a expl:Wellbore .
:Wellbore-30/8-5 a expl:Wellbore .
:Wellbore-16/1-29_S expl:hasMeasurement :FormationPressure-FP-1249 .
:FormationPressure-FP-1249 a expl:Pressure .
:FormationPressure-FP-1249 expl:hasDepth :TVD-FP-1249 .
:Wellbore-16/1-29_S expl:hasMeasurement :FormationPressure-FP-1377 .
:FormationPressure-FP-1377 a expl:Pressure .
:FormationPressure-FP-1377 expl:hasDepth :TVD-FP-1377 .
:TVD-FP-1377 expl:inWellboreInterval :SZ-4 .
:SZ-4 expl:hasUnit :Stratigraphic-Unit-ROGALAND .
```

SPARQL: Optional Matching

```
SELECT ?w ?d ?u
WHERE {
    ?w a expl:Wellbore . ?w expl:hasMeasurement ?p .
    ?p a expl:Pressure . ?p expl:hasDepth ?d
    OPTIONAL {
        ?d expl:inWellboreInterval ?z .
        ?z expl:hasUnit ?u
    }
}
```

if the optional part does not match, it creates **no bindings** but **does not eliminate** the solution

```
:Wellbore-16/1-29_S a expl:Wellbore .
:Wellbore-30/8-5 a expl:Wellbore .
:Wellbore-16/1-29_S expl:hasMeasurement :FormationPressure-FP-1249 .
:FormationPressure-FP-1249 a expl:Pressure .
:FormationPressure-FP-1249 expl:hasDepth :TVD-FP-1249 .
:Wellbore-16/1-29_S expl:hasMeasurement :FormationPressure-FP-1377 .
:FormationPressure-FP-1377 a expl:Pressure .
:FormationPressure-FP-1377 expl:hasDepth :TVD-FP-1377 .
:TVD-FP-1377 expl:inWellboreInterval :SZ-4 .
:SZ-4 expl:hasUnit :Stratigraphic-Unit-ROGALAND .
```

answer:

?w	?d	?u
:Wellbore-16/1-29_S	:TVD-FP-1249	—
:Wellbore-16/1-29_S	:TVD-FP-1377	:Stratigraphic-Unit-ROGALAND

SPARQL: Optional Matching

```
SELECT ?w ?d ?u
WHERE {
    ?w a expl:Wellbore . ?w expl:hasMeasurement ?p .
    ?p a expl:Pressure . ?p expl:hasDepth ?d
    OPTIONAL {
        ?d expl:inWellboreInterval ?z .
        ?z expl:hasUnit ?u
    }
}
```

if the optional part does not match,
it creates **no bindings**
but **does not eliminate** the solution

```
:Wellbore-16/1-29_S a expl:Wellbore .
:Wellbore-30/8-5 a expl:Wellbore .
:Wellbore-16/1-29_S expl:hasMeasurement :FormationPressure-FP-1249 .
:FormationPressure-FP-1249 a expl:Pressure .
:FormationPressure-FP-1249 expl:hasDepth :TVD-FP-1249 .
:Wellbore-16/1-29_S expl:hasMeasurement :FormationPressure-FP-1377 .
:FormationPressure-FP-1377 a expl:Pressure .
:FormationPressure-FP-1377 expl:hasDepth :TVD-FP-1377 .
:TVD-FP-1377 expl:inWellboreInterval :SZ-4 .
:SZ-4 expl:hasUnit :Stratigraphic-Unit-ROGALAND .
```

answer:

?w	?d	?u
:Wellbore-16/1-29_S	:TVD-FP-1249	—
:Wellbore-16/1-29_S	:TVD-FP-1377	:Stratigraphic-Unit-ROGALAND

NB: similar to `LEFT JOIN` in SQL

Remarks on SPARQL 1.1

we have seen the following features of SPARQL:

- Basic Graph Patterns
- UNION
- OPTIONAL

Remarks on SPARQL 1.1

we have seen the following features of SPARQL:

- Basic Graph Patterns
- UNION
- OPTIONAL

SPARQL 1.1 has many additional features:

- complex FILTER conditions
- GROUP BY, to express aggregations and support aggregation operators
- MINUS, to remove possible solutions
- FILTER NOT EXISTS, to test for the absence of a pattern
- property paths (regular expressions)
- solution modifiers (LIMIT, ORDER BY)
- CONSTRUCT / ASK / DESCRIBE queries
- ...

Mappings Relational Data to RDF

```
SELECT IDENTIFIER FROM WELLBORE
WHERE REF_EXISTENCE_KIND = 'actual'
  ⇝ Wellbore(iri("Wellbore-", IDENTIFIER))
```

Mappings Relational Data to RDF

```
SELECT IDENTIFIER FROM WELLBORE
WHERE REF_EXISTENCE_KIND = 'actual'
  ⇝ Wellbore(iri("Wellbore-", IDENTIFIER))
```

source query
(in SQL for relational databases)

target query
(atoms in the ontology vocabulary)

Mappings Relational Data to RDF

```
SELECT IDENTIFIER FROM WELLBORE
WHERE REF_EXISTENCE_KIND = 'actual'
  ⇝ Wellbore(iri("Wellbore-", IDENTIFIER))
```

source query
(in SQL for relational databases)

target query
(atoms in the ontology vocabulary)

the `iri` function constructs IRIs by concatenating any number of strings

`WELLBORE` table

IDENTIFIER	REF_EXISTENCE_KIND	...
16/1-29_S	actual	...
30/8-5	actual	...
33/10-12	planned	...

Mappings Relational Data to RDF

```
SELECT IDENTIFIER FROM WELLBORE
WHERE REF_EXISTENCE_KIND = 'actual'
  ⇝ Wellbore(iri("Wellbore-", IDENTIFIER))
```

source query
(in SQL for relational databases)
target query
(atoms in the ontology vocabulary)

the `iri` function constructs IRIs by concatenating any number of strings

`WELLBORE` table

IDENTIFIER	REF_EXISTENCE_KIND	...
16/1-29_S	actual	...
30/8-5	actual	...
33/10-12	planned	...

result:

```
Wellbore(Wellbore-16/1-29_S)
Wellbore(Wellbore-30/8-5)
```
↑ **IRIs**

R2RML

```
map:m-00001
    a rr:TriplesMap ;
    rr:logicalTable [
        a rr:R2RMLView ;
        rr:sqlQuery " SELECT * FROM WELLBORE WHERE REF_EXISTENCE_KIND = 'actual' "
    ] ;
    rr:predicateObjectMap [
        a rr:PredicateObjectMap ;
        rr:objectMap [
            a rr:ObjectMap, rr:TermMap ;
            rr:template "http://slegger.gitlab.io/data#TotalCoreLength-{IDENTIFIER}" ;
            rr:termType rr:IRI
        ] ;
        rr:predicate expl:hasTotalCoreLength
    ] ;
    rr:subjectMap [
        a rr:SubjectMap, rr:TermMap ;
        rr:class expl:Wellbore ;
        rr:template " http://slegger.gitlab.io/data#Wellbore-{IDENTIFIER} " ;
        rr:termType rr:IRI
    ] .
```

a language for expressing **customised mappings** from relational databases to RDF datasets

(W3C recommendation 2012)

R2RML

a language for expressing **customised mappings** from relational databases to RDF datasets (W3C recommendation 2012)

```
map:m-00001
    a rr:TriplesMap ;
    rr:logicalTable [
        a rr:R2RMLView ;
        rr:sqlQuery " SELECT * FROM WELLBORE WHERE REF_EXISTENCE_KIND = 'actual' "
    ] ;
    rr:predicateObjectMap [
        a rr:PredicateObjectMap ;
        rr:objectMap [
            a rr:ObjectMap, rr:TermMap ;
            rr:template "http://slegger.gitlab.io/data#TotalCoreLength-{IDENTIFIER}" ;
            rr:termType rr:IRI
        ] ;
        rr:predicate expl:hasTotalCoreLength
    ] ;
    rr:subjectMap [
        a rr:SubjectMap, rr:TermMap ;
        rr:class expl:Wellbore ;
        rr:template " http://slegger.gitlab.io/data#Wellbore-{IDENTIFIER} " ;
        rr:termType rr:IRI
    ] .
```

produces RDF graph

```
:Wellbore-16/1-29_S a expl:Wellbore .
:Wellbore-30/8-5 a expl:Wellbore .
```

R2RML

a language for expressing **customised mappings** from relational databases to RDF datasets

(W3C recommendation 2012)

```
map:m-00001
    a rr:TriplesMap ;
    rr:logicalTable [
        a rr:R2RMLView ;
        rr:sqlQuery " SELECT * FROM WELLBORE WHERE REF_EXISTENCE_KIND = 'actual' "
    ] ;
    rr:predicateObjectMap [
        a rr:PredicateObjectMap ;
        rr:objectMap [
            a rr:ObjectMap, rr:TermMap ;
            rr:template "http://slegger.gitlab.io/data#TotalCoreLength-{IDENTIFIER}" ;
            rr:termType rr:IRI
        ] ;
        rr:predicate expl:hasTotalCoreLength
    ] ;
    rr:subjectMap [
        a rr:SubjectMap, rr:TermMap ;
        rr:class expl:Wellbore ;
        rr:template " http://slegger.gitlab.io/data#Wellbore-{IDENTIFIER} " ;
        rr:termType rr:IRI
    ] .
```

produces RDF graph

```
:Wellbore-16/1-29_S a expl:Wellbore .
:Wellbore-30/8-5 a expl:Wellbore .
```

NB: is this a GAV (Global-As-View) mapping?

R2RML

a language for expressing **customised mappings** from relational databases to RDF datasets

(W3C recommendation 2012)

```
map:m-00001
    a rr:TriplesMap ;
    rr:logicalTable [
        a rr:R2RMLView ;
        rr:sqlQuery " SELECT * FROM WELLBORE WHERE REF_EXISTENCE_KIND = 'actual' "
    ] ;
    rr:predicateObjectMap [
        a rr:PredicateObjectMap ;
        rr:objectMap [
            a rr:ObjectMap, rr:TermMap ;
            rr:template "http://slegger.gitlab.io/data#TotalCoreLength-{IDENTIFIER}" ;
            rr:termType rr:IRI
        ] ;
        rr:predicate expl:hasTotalCoreLength
    ] ;
    rr:subjectMap [
        a rr:SubjectMap, rr:TermMap ;
        rr:class expl:Wellbore ;
        rr:template " http://slegger.gitlab.io/data#Wellbore-{IDENTIFIER} " ;
        rr:termType rr:IRI
    ] .
```

produces RDF graph

```
:Wellbore-16/1-29_S a expl:Wellbore .
:Wellbore-30/8-5 a expl:Wellbore .
```

NB: is this a GAV (Global-As-View) mapping?

not quite — the IRI function can simulate GLAV (more in part 2...)

References

1. G. Xiao, D. Calvanese, R. Kontchakov, D. Lembo, A. Poggi, R. Rosati & M. Zakharyaschev. "Ontology-Based Data Access: A Survey". In: Proc. of the 27th Int. Joint Conf. on Artificial Intelligence, IJCAI-ECAI 2018, 5511–5519. ijcai.org, 2018.

2. D. Calvanese, G. De Giacomo, D. Lembo, M. Lenzerini & R. Rosati. "Tractable reasoning and efficient query answering in description logics: The DL-Lite family." In: JAR, 39(3):385–429, 2007.

3. "RDF 1.1 Concepts and Abstract Syntax" R. Cyganiak, D. Wood, M. Lanthaler, editors. W3C, 2014.
   https://www.w3.org/TR/rdf11-concepts

4. "OWL 2 Web Ontology Language Profiles (Second Edition)" B. Motik, B. Cuenca Grau, I. Horrocks, Z. Wu, A. Fokoue, C. Lutz, editors. W3C, 2012.
   https://www.w3.org/TR/owl2-profiles

5. "SPARQL 1.1 Query Language" S. Harris & A. Seaborne, editors. W3C, 2013.
   https://www.w3.org/TR/sparql11-query

6. "R2RML: RDB to RDF Mapping Language" S. Das, S. Sundara, R. Cyganiak, editors. W3C, 2012.
   https://www.w3.org/TR/r2rml