

## 2. Unit: SPARQL Formal Semantics

**Exercise 2.1 (SPARQL Formal Semantics)** Consider the SPARQL Formal Semantics.

- Define a “null-tolerant join” for the relational algebra that acts like the  $\bowtie$  of the SPARQL algebra.
- Which SQL construct is similar to the “\” operator in the SPARQL algebra?
- In the SPARQL algebra, OPT is expressed via left outer join, which is defined via “\” (while a corresponding MINUS does not exist in the SPARQL syntax).  
Such a MINUS (cf. part (b) of this exercise) provides a more intuitive idea of negation than “! bound( $x$ )”. Give a general pattern how to express ( $P_1$  MINUS  $P_2$ ) in SPARQL 1.0 syntax.
- Recall the definition of  $\bowtie$  in the relational algebra (DB lecture) and define SPARQL’s  $\bowtie$  in a similar way.

**Exercise 2.2 (Outer Join)** Recall that SPARQL’s OPTIONAL corresponds to a left outer join.

- Give a general pattern how to express a *full* outer join (i.e., “outer” to both sides) in the SPARQL algebra (consider as input two mappings  $R$  and  $S$  and give an expression for  $R \bowtie S$ ) and in SPARQL.
- Give all cities (name as ?XN) that are the capital of a country (:capital) or that are located at a river (:locatedAt) or both (return the names ?CN of the country and/or the river (?RN)).

**Exercise 2.3 (SPARQL Formal Semantics: OPTIONAL)** Consider the SPARQL Formal Semantics.

Prove or show a counterexample:

The statement (from W3C SPARQL Working Draft 20061004)

If  $\text{OPT}(A, B)$  is an optional graph pattern, where  $A$  and  $B$  are graph patterns, then  $S$  is a solution of  $\text{OPT}(A, B)$  if

- $S$  is a pattern solution of  $A$  and of  $B$ , or
- $S$  is a solution to  $A$ , but not to  $B$ .

describes the same semantics as above.

**Exercise 2.4 (SPARQL: Filter-Safe Expressions)** Consider the following definition:

**Definition 1 ([PAG06, AG 08])** A SPARQL expression is *filter-safe*, if for every subexpression of the form  $(P \text{ FILTER } R)$ ,  $\text{var}(R) \subseteq \text{var}(P)$ .

- a) Give a SPARQL query for the following: For each country, give the capital and its population, and, if exist, all cities in that country that have a higher population, as a result table of the following form

C	CAP	CapPop	City	CityPop
D	Berlin	3472009		
CH	Bern	128848	Zürich	384786
CH	Bern	128848	Genf	191557
:	:	:	:	

- b) Make your query filter-safe (if it is not yet filter-safe).
- c) Give the same query in SQL (hint: use the LEFT OUTER JOIN ... ON ... construct)
- d) Sketch an algorithm that rewrites non-filter safe queries into safe ones. First, try it on your own, then maybe look in [AG08].
- e) Give a SPARQL query for “Give the names of all countries, such that there is some city in that country where more than 1/4 of the population are living in”, and make it filter-safe.
- f) Give a SPARQL query for “Give the names of all countries, such that there is *no* city in that country where more than 1/4 of the population are living in”, and make it filter-safe.
- g) Is there a similar thing in SQL, in the relational algebra, and in the relational calculus (DBT Lecture)?