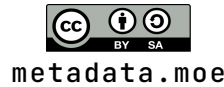


# SPARQL Cheat Sheet



SPARQL 1.1: <http://www.w3.org/TR/sparql11-query/>

	cardinality	
Prologue	0-1	BASE <...>
	0-N	PREFIX ex: <...>
Select Query	1	SELECT ...
	0-N	FROM <...>
	0-N	FROM NAMED <...>
	0-1	WHERE
Graph pattern and Subquery	1	{
	0-N	...
	0-N	{ ... }
	1	}
Solution modifier	0-1	GROUP BY ...
	0-1	HAVING ...
	0-1	ORDER BY ...
	0-1	LIMIT ...
	0-1	OFFSET ...
0-1	VALUES ...	

## SELECT Query

指定した変数の値を返す:  
`SELECT ?s (isIRI(?o) AS ?l)  
WHERE { ?s ?p ?o . }`

すべての変数の値を返す:  
`SELECT *  
WHERE { ?s ?p ?o . }`

重複を排除する:  
`SELECT DISTINCT *  
WHERE { ... }`

重複を排除する(結果が一意とは限らない):  
`SELECT REDUCED *  
WHERE { ... }`

## CONSTRUCT Query

テンプレートに従ってRDFグラフを返す:  
`CONSTRUCT { ?s rdfs:label ?t }  
WHERE { ?s ex:title ?t . }`

## ASK Query

パターンがマッチするかどうかをtrueかfalseで返す:  
`ASK { ?s ?p ?o . }`

## DESCRIBE Query

リソースに関するRDFグラフを返す:  
`DESCRIBE <http://example.com/>`

`DESCRIBE ?x ?y  
WHERE { ?x foaf:knows ?y . }`

## RDF Dataset

デフォルトグラフを指定する:  
`SELECT *  
FROM ex:g1  
FROM ex:g2  
WHERE { ... }`

名前付きグラフを指定する:  
`SELECT *  
FROM NAMED ex:g3  
WHERE { GRAPH ex:g3 { ... } }`

## RDF Syntax

### リテラル

"cat"  
"ネコ"@ja  
"2023-12-31"^^xsd:dateTime  
1234  
true

### 変数

次の表記は等価:  
`?var $var`

### 空白ノード

次のトリプルは等価:  
`[ :p "v" ] .  
[ ] :p "v" .  
_:b57 :p "v" .`

### トリプルパターン

#### 共通の主語

`?x foaf:name ?name .  
?x foaf:mbox ?mbox .`

「;」でまとめて記述できる:

`?x foaf:name ?name ;  
foaf:mbox ?mbox .`

#### 共通の主語と述語

`?x foaf:nick "Alice" .  
?x foaf:nick "Alice_" .`

「,」でまとめて記述できる:

`?x foaf:nick "Alice" , "Alice_" .`

#### rdf:type

次のトリプルは等価:  
`?x a ex:Class1 .  
?x rdf:type ex:Class1 .`

## Graph Patterns

### 省略可能パターン

`{ ... OPTIONAL { ... } }`

### 代替パターン

`{ ... } UNION { ... }`

### 除外パターン

`{ ... MINUS { ... } }`

### サブクエリ

`{ ... { SELECT ... } }`

## 統合クエリ

`{ ...  
SERVICE ex:sparql { ... } }`

## フィルタ

評価結果がtrueとならない場合を除外:  
`{ ... FILTER (?age < 20) }  
{ ... FILTER isIRI(?x) }`

## BIND

`BIND (?min * 60 AS ?sec)`

## VALUES

1つの変数に複数の値:  
`VALUES ?x { "a" "b" "c" }`

複数の変数に複数の値:

`VALUES (?x ?y) {  
("a" 1) ("b" 2) ("c" 3)  
}`

## プロパティパス

`^ex:p1+/(ex:p2|ex:p3)/ex:p4?`

## 優先順:

- iri IRIまたは「a」
- !iri 否定
- !(iri<sub>1</sub>|iri<sub>2</sub>|...) 否定のプロパティ集合
- (elt) グループ
- elt\* 0回以上
- elt? 0回または1回
- elt+ 1回以上
- ^elt 逆向き
- elt<sub>1</sub>/elt<sub>2</sub> 順方向
- elt<sub>1</sub>|elt<sub>2</sub> 代替

## Solution Modifiers

### GROUP BY / HAVING

グループ条件に従い結果を集約:

`SELECT ?y (COUNT(?s) AS ?c)  
WHERE {  
?s ex:published ?y  
}  
GROUP BY ?y  
HAVING (?c > 100)`

### ORDER BY / LIMIT / OFFSET

結果の順序、取得上限、取得開始位置を指定:  
`SELECT ?y (COUNT(?s) AS ?c)  
WHERE {  
?s ex:published ?y  
}  
ORDER BY DESC(?y)  
LIMIT 25  
OFFSET 50`