The SPARQL queries for the lemonEXT and OBANmod ontologies: results in Table 9

The following three SPARQL SELECT queries exploit the lemonEXT core ontology once it is populated (the OWL file). The SELECT SPARQL queries q1 and q2 have in bold the UMLS Semantic Types for them. The SELECT SPARQL query q3 has in bold the UMLS Semantic Group to which is applied, i.e. CHEM.

Query(parameters)	SPARQL SELECT Query
q1(OWLfile,targetTerm)	SELECT (COUNT(DISTINCT ?C) AS ?count) FROM <file:./InOWL/diso11_onto/lemonEXT_core__xmlRdf.owl> FROM NAMED <file: OWLfile> { { GRAPH <file: OWLfile> { ?lx rdf:type lemon:Lexicon ; lemon:entry ?y ; lemon:topic ?targetTerm . ?targetTerm rdf:type owl:NamedIndividual; rdfs:label ?lbtargetTerm . ?y rdf:type owl:NamedIndividual; ontolex:denotes ?C . ?C rdfs:subClassOf ?ST . FILTER (**(?ST = ustg:T059 || ?ST = ustg:T060 || ?ST = ustg:T061 || ?ST = ustg:T058)** && ?lbtargetTerm = "targetTerm"@en) } } }
q2(OWLfile,targetTerm)	SELECT (COUNT(DISTINCT ?C) AS ?count) FROM <file:./InOWL/diso11_onto/lemonEXT_core__xmlRdf.owl> FROM NAMED <file: OWLfile> { { GRAPH <file: OWLfile> { ?lx rdf:type lemon:Lexicon ; lemon:entry ?y ; lemon:topic ?targetTerm . ?targetTerm rdf:type owl:NamedIndividual; rdfs:label ?lbtargetTerm . ?y rdf:type owl:NamedIndividual; ontolex:denotes ?C . ?C rdfs:subClassOf ?ST . FILTER (**(?ST = ustg:T034 || ?ST = ustg:T184 || ?ST = ustg:T033)** && ?lbtargetTerm = "targetTerm"@en) } } }
q3(OWLfile,targetTerm)	SELECT (COUNT(DISTINCT ?C) AS ?count) FROM <file:./InOWL/diso11_onto/lemonEXT_core__xmlRdf.owl> FROM NAMED <file: OWLfile> { { GRAPH <file: OWLfile> { ?lx rdf:type lemon:Lexicon ; lemon:entry ?y ; lemon:topic ?targetTerm . ?targetTerm rdf:type owl:NamedIndividual; rdfs:label ?lbtargetTerm . ?y rdf:type owl:NamedIndividual; ontolex:denotes ?C . ?C rdfs:subClassOf ?ST . ?ST rdf:type owl:Class ; rdfs:subClassOf [a owl:Restriction ; owl:onProperty obo:BFO_0000050; owl:someValuesFrom ?SG] . FILTER (**?SG = ustg:CHEM** && ?lbtargetTerm = "targetTerm"@en) } } }

The next three SPARQL SELECT queries exploit the OBANmod core ontology once it is populated (the OWL file). The SELECT SPARQL queries q1V and q2V have in bold the UMLS Semantic Types for them. The SELECT SPARQL query q3V has in bold the UMLS Semantic Group to which is applied, i.e. CHEM.

Query(parameters)	**SPARQL SELECT Query**
q1V(OWLfile,CUItargetTerm)	SELECT (COUNT(DISTINCT ?C) AS ?count) FROM <file:./InOWL/diso11_onto/OBANmod_core__xmlRdf.owl> FROM NAMED <file:OWLfile> { { GRAPH <file:OWLfile> { ?x rdf:type oban:association ; oban:association_has_object ?C ; oban:association_has_subject ustg:CUItargetTerm . ?C rdfs:subClassOf ?ST . ?y rdf:type oban:provenance ; obo:IAO_0000136 ?x ; obo:RO_0002558 ?lbV . FILTER (**(?ST = ustg:T059 || ?ST = ustg:T060 || ?ST = ustg:T061 || ?ST = ustg:T058)** && (?lbV = OBANmod:OBANmod_2 || ?lbV = OBANmod:OBANmod_3 || ?lbV = OBANmod:OBANmod_4 || ?lbV = OBANmod:OBANmod_5)) } } }
q2V(OWLfile, CUItargetTerm)	SELECT (COUNT(DISTINCT ?C) AS ?count) FROM <file:./InOWL/diso11_onto/OBANmod_core__xmlRdf.owl> FROM NAMED <file:OWLfile> { { GRAPH <file:OWLfile> { ?x rdf:type oban:association ; oban:association_has_object ?C ; oban:association_has_subject ustg:CUItargetTerm . ?C rdfs:subClassOf ?ST . ?y rdf:type oban:provenance ; obo:IAO_0000136 ?x ; obo:RO_0002558 ?lbV . FILTER (**(?ST = ustg:T034 || ?ST = ustg:T184 || ?ST = ustg:T033)** && (?lbV = OBANmod:OBANmod_2 || ?lbV = OBANmod:OBANmod_3 || ?lbV = OBANmod:OBANmod_4 || ?lbV = OBANmod:OBANmod_5)) } } }
q3V(OWLfile, CUItargetTerm)	SELECT (COUNT(DISTINCT ?C) AS ?count) FROM <file:./InOWL/diso11_onto/OBANmod_core__xmlRdf.owl> FROM NAMED <file:OWLfile> { { GRAPH <file:OWLfile> { ?x rdf:type oban:association ; oban:association_has_object ?C ; oban:association_has_subject ustg:CUItargetTerm . ?C rdfs:subClassOf ?ST . ?ST rdf:type owl:Class ; rdfs:subClassOf [a owl:Restriction ; owl:onProperty obo:BFO_0000050; owl:someValuesFrom ?SG] . ?y rdf:type oban:provenance ; obo:IAO_0000136 ?x ; obo:RO_0002558 ?lbV . FILTER (**?SG = ustg:CHEM** && (?lbV = OBANmod:OBANmod_2 || ?lbV = OBANmod:OBANmod_3 || ?lbV = OBANmod:OBANmod_4 || ?lbV = OBANmod:OBANmod_5)) } } }

“One Health” queries in SPARQL: results in Table 11

The following three SPARQL SELECT queries exploit the OBANmod core ontology once it is populated for both datasets, i.e. the VetCN dataset (the OWL file 1) and the PMSB dataset (the OWL file 2). The SELECT SPARQL queries q1VU and q2VU have in bold the UMLS Semantic Types for them. The SELECT SPARQL query q3VU has in bold the UMLS Semantic Group to which is applied, i.e. CHEM.

Query(parameters)	SPARQL SELECT Query

q1VU(OWLfile1, OWLfile2, CUItargetTerm)	SELECT (COUNT(DISTINCT ?C) AS ?count)
	FROM <file:./InOWL/diso11_onto/OBANmod_core__xmlRdf.owl>
	FROM NAMED <file:OWLfile1>
	FROM NAMED <file:OWLfile2>
	{ { GRAPH <file:OWLfile1> {
	
	?x rdf:type oban:association ; oban:association_has_object ?C ;
	oban:association_has_subject ustg:CUItargetTerm .
	
	?C rdfs:subClassOf ?ST .
	
	?y rdf:type oban:provenance ; obo:IAO_0000136 ?x ; obo:RO_0002558 ?lbV .
	
	FILTER (**(?ST = ustg:T059 || ?ST = ustg:T060 || ?ST = ustg:T061 || ?ST = ustg:T058)**
	&& (?lbV = OBANmod:OBANmod_2 || ?lbV = OBANmod:OBANmod_3 ||
	?lbV = OBANmod:OBANmod_4 || ?lbV = OBANmod:OBANmod_5)) } }
	
	UNION
	
	{ GRAPH <file:OWLfile2 > {
	
	?x rdf:type oban:association ; oban:association_has_object ?C ;
	oban:association_has_subject ustg:CUItargetTerm .
	
	?C rdfs:subClassOf ?ST .
	
	?y rdf:type oban:provenance ; obo:IAO_0000136 ?x ; obo:RO_0002558 ?lbV .
	
	FILTER (**(?ST = ustg:T059 || ?ST = ustg:T060 || ?ST = ustg:T061 || ?ST = ustg:T058)**
	&& (?lbV = OBANmod:OBANmod_2 || ?lbV = OBANmod:OBANmod_3 ||
	?lbV = OBANmod:OBANmod_4 || ?lbV = OBANmod:OBANmod_5))
	} } }

q2VU(OWLfile1, OWLfile2, CUItargetTerm)	SELECT (COUNT(DISTINCT ?C) AS ?count) FROM <file:./InOWL/diso11_onto/OBANmod_core__xmlRdf.owl> FROM NAMED <file:OWLfile1> FROM NAMED <file:OWLfile2> { { GRAPH <file:OWLfile1> { ?x rdf:type oban:association ; oban:association_has_object ?C ; oban:association_has_subject ustg:CUItargetTerm . ?C rdfs:subClassOf ?ST . ?y rdf:type oban:provenance ; obo:IAO_0000136 ?x ; obo:RO_0002558 ?lbV . FILTER (**(?ST = ustg:T034 || ?ST = ustg:T184 || ?ST = ustg:T033)** && (?lbV = OBANmod:OBANmod_2 || ?lbV = OBANmod:OBANmod_3 || ?lbV = OBANmod:OBANmod_4 || ?lbV = OBANmod:OBANmod_5)) } } UNION

	{ GRAPH <file:OWLfile2 > { ?x rdf:type oban:association ; oban:association_has_object ?C ; oban:association_has_subject ustg:CUItargetTerm . ?C rdfs:subClassOf ?ST . ?y rdf:type oban:provenance ; obo:IAO_0000136 ?x ; obo:RO_0002558 ?lbV . FILTER ((**?ST = ustg:T034 || ?ST = ustg:T184 || ?ST = ustg:T033)** && (?lbV = OBANmod:OBANmod_2 || ?lbV = OBANmod:OBANmod_3 || ?lbV = OBANmod:OBANmod_4 || ?lbV = OBANmod:OBANmod_5)) } } }

q3VU(OWLfile1,
OWLfile2,
CUItargetTerm)
	SELECT (COUNT(DISTINCT ?C) AS ?count)
	FROM <file:./InOWL/diso11_onto/OBANmod_core__xmlRdf.owl>
	FROM NAMED <file:OWLfile1>
	FROM NAMED <file:OWLfile2>
	{ { GRAPH <file:OWLfile1> {
	
	?x rdf:type oban:association ; oban:association_has_object ?C ;
	oban:association_has_subject ustg:CUItargetTerm .
	
	?C rdfs:subClassOf ?ST .
	
	?ST rdf:type owl:Class ; rdfs:subClassOf [a owl:Restriction ;
	owl:onProperty obo:BFO_0000050; owl:someValuesFrom ?SG] .
	
	?y rdf:type oban:provenance ; obo:IAO_0000136 ?x ; obo:RO_0002558 ?lbV .
	
	FILTER (**?SG = ustg:CHEM**
	&& (?lbV = OBANmod:OBANmod_2 || ?lbV = OBANmod:OBANmod_3 ||
	?lbV = OBANmod:OBANmod_4 || ?lbV = OBANmod:OBANmod_5)) } }
	
	UNION
	
	{ GRAPH <file:OWLfile2 > {
	
	?x rdf:type oban:association ; oban:association_has_object ?C ;
	oban:association_has_subject ustg:CUItargetTerm .
	
	?C rdfs:subClassOf ?ST .
	
	?ST rdf:type owl:Class ; rdfs:subClassOf [a owl:Restriction ;
	owl:onProperty obo:BFO_0000050; owl:someValuesFrom ?SG] .
	
	?y rdf:type oban:provenance ; obo:IAO_0000136 ?x ; obo:RO_0002558 ?lbV .
	
	FILTER (**?SG = ustg:CHEM**
	&& (?lbV = OBANmod:OBANmod_2 || ?lbV = OBANmod:OBANmod_3 ||
	?lbV = OBANmod:OBANmod_4 || ?lbV = OBANmod:OBANmod_5))
	} } }

The following three SPARQL SELECT queries exploit the OBANmod core ontology once it is populated for both datasets, i.e. the VetCN dataset (the OWL file 1) and the PMSB dataset (the OWL file 2). The SELECT SPARQL queries q1VM and q2VM have in bold the UMLS Semantic Types for them. The SELECT SPARQL query q3VM has in bold the UMLS Semantic Group to which is applied, i.e. CHEM.

Query(parameters)	**SPARQL SELECT Query**

q1VM(OWLfile1, OWLfile2, CUItargetTerm)	SELECT (COUNT(DISTINCT ?C) AS ?count)
	FROM <file:./InOWL/diso11_onto/OBANmod_core__xmlRdf.owl>
	FROM NAMED <file:OWLfile1>
	FROM NAMED <file:OWLfile2>
	{ { GRAPH <file:OWLfile1> {
	
	?x rdf:type oban:association ; oban:association_has_object ?C ;
	oban:association_has_subject ustg:CUItargetTerm .
	
	?C rdfs:subClassOf ?ST ; ustg:hasDbXrefInSCT ?SCTid .
	
	?y rdf:type oban:provenance ; obo:IAO_0000136 ?x ; obo:RO_0002558 ?lbV .
	
	FILTER (**(?ST = ustg:T059 || ?ST = ustg:T060 || ?ST = ustg:T061 || ?ST = ustg:T058)**
	&& (?lbV = OBANmod:OBANmod_2 || ?lbV = OBANmod:OBANmod_3 ||
	?lbV = OBANmod:OBANmod_4 || ?lbV = OBANmod:OBANmod_5)) } }
	
	UNION
	
	{ GRAPH <file:OWLfile2 > {
	
	?x rdf:type oban:association ; oban:association_has_object ?C ;
	oban:association_has_subject ustg:CUItargetTerm .
	
	?C rdfs:subClassOf ?ST ; ustg:hasDbXrefInSCT ?SCTid .
	
	?y rdf:type oban:provenance ; obo:IAO_0000136 ?x ; obo:RO_0002558 ?lbV .
	
	FILTER (**(?ST = ustg:T059 || ?ST = ustg:T060 || ?ST = ustg:T061 || ?ST = ustg:T058)**
	&& (?lbV = OBANmod:OBANmod_2 || ?lbV = OBANmod:OBANmod_3 ||
	?lbV = OBANmod:OBANmod_4 || ?lbV = OBANmod:OBANmod_5))
	} } }

q2VM(OWLfile1, OWLfile2, CUItargetTerm)	SELECT (COUNT(DISTINCT ?C) AS ?count) FROM <file:./InOWL/diso11_onto/OBANmod_core__xmlRdf.owl> FROM NAMED <file:OWLfile1> FROM NAMED <file:OWLfile2> { { GRAPH <file:OWLfile1> { ?x rdf:type oban:association ; oban:association_has_object ?C ; oban:association_has_subject ustg:CUItargetTerm . ?C rdfs:subClassOf ?ST ; ustg:hasDbXrefInSCT ?SCTid . ?y rdf:type oban:provenance ; obo:IAO_0000136 ?x ; obo:RO_0002558 ?lbV . FILTER (**(?ST = ustg:T034 || ?ST = ustg:T184 || ?ST = ustg:T033)** && (?lbV = OBANmod:OBANmod_2 || ?lbV = OBANmod:OBANmod_3 || ?lbV = OBANmod:OBANmod_4 || ?lbV = OBANmod:OBANmod_5)) } } UNION

	{ GRAPH <file:OWLfile2 > { ?x rdf:type oban:association ; oban:association_has_object ?C ; oban:association_has_subject ustg:CUItargetTerm . ?C rdfs:subClassOf ?ST ; ustg:hasDbXrefInSCT ?SCTid . ?y rdf:type oban:provenance ; obo:IAO_0000136 ?x ; obo:RO_0002558 ?lbV . FILTER ((**?ST = ustg:T034 || ?ST = ustg:T184 || ?ST = ustg:T033)** && (?lbV = OBANmod:OBANmod_2 || ?lbV = OBANmod:OBANmod_3 || ?lbV = OBANmod:OBANmod_4 || ?lbV = OBANmod:OBANmod_5)) } } }

q3VM(OWLfile1,
OWLfile2,
CUItargetTerm)
	SELECT (COUNT(DISTINCT ?C) AS ?count)
	FROM <file:./InOWL/diso11_onto/OBANmod_core__xmlRdf.owl>
	FROM NAMED <file:OWLfile1>
	FROM NAMED <file:OWLfile2>
	{ { GRAPH <file:OWLfile1> {
	
	?x rdf:type oban:association ; oban:association_has_object ?C ;
	oban:association_has_subject ustg:CUItargetTerm .
	
	?C rdfs:subClassOf ?ST ; ustg:hasDbXrefInSCT ?SCTid .
	
	?ST rdf:type owl:Class ; rdfs:subClassOf [a owl:Restriction ;
	owl:onProperty obo:BFO_0000050; owl:someValuesFrom ?SG] .
	
	?y rdf:type oban:provenance ; obo:IAO_0000136 ?x ; obo:RO_0002558 ?lbV .
	
	FILTER (**?SG = ustg:CHEM**
	&& (?lbV = OBANmod:OBANmod_2 || ?lbV = OBANmod:OBANmod_3 ||
	?lbV = OBANmod:OBANmod_4 || ?lbV = OBANmod:OBANmod_5)) } }
	
	UNION
	
	{ GRAPH <file:OWLfile2 > {
	
	?x rdf:type oban:association ; oban:association_has_object ?C ;
	oban:association_has_subject ustg:CUItargetTerm .
	
	?C rdfs:subClassOf ?ST ; ustg:hasDbXrefInSCT ?SCTid .
	
	?ST rdf:type owl:Class ; rdfs:subClassOf [a owl:Restriction ;
	owl:onProperty obo:BFO_0000050; owl:someValuesFrom ?SG] .
	
	?y rdf:type oban:provenance ; obo:IAO_0000136 ?x ; obo:RO_0002558 ?lbV .
	
	FILTER (**?SG = ustg:CHEM**
	&& (?lbV = OBANmod:OBANmod_2 || ?lbV = OBANmod:OBANmod_3 ||
	?lbV = OBANmod:OBANmod_4 || ?lbV = OBANmod:OBANmod_5))
	} } }

The following three SPARQL SELECT queries exploit: a) the asserted information within the locality-based modules created per target term from the SNOMED CT ontology; and b) the OBANmod core ontology once it is populated for both datasets, i.e. the VetCN dataset (the OWL file 1) and the PMSB dataset (the OWL file 2). The SELECT SPARQL queries q1VS and q2VS have in bold the UMLS Semantic Types for them. The SELECT SPARQL query q3VS has in bold the UMLS Semantic Group to which is applied, i.e. CHEM.

Query(parameters)	**SPARQL SELECT Query**

q1VS(OWLfile1, OWLfile2, CUItargetTerm, OWLtargetTermModule)	SELECT (COUNT(DISTINCT ?SCT) AS ?count)
	FROM <file:OWLtargetTermModule>
	FROM NAMED <file:OWLfile1>
	FROM NAMED <file:OWLfile2>
	{ { GRAPH <file:OWLfile1> {
	
	?x rdf:type oban:association ; oban:association_has_object ?C ;
	oban:association_has_subject ustg:CUItargetTerm .
	
	?C rdfs:subClassOf ?ST ; ustg:hasDbXrefInSCT ?SCTid .
	
	?y rdf:type oban:provenance ; obo:IAO_0000136 ?x ; obo:RO_0002558 ?lbV .
	
	FILTER (**(?ST = ustg:T059 || ?ST = ustg:T060 || ?ST = ustg:T061 || ?ST = ustg:T058)**
	&& (?lbV = OBANmod:OBANmod_2 || ?lbV = OBANmod:OBANmod_3 ||
	?lbV = OBANmod:OBANmod_4 || ?lbV = OBANmod:OBANmod_5))
	}
	
	?SCT a owl:Class .
	
	FILTER (fn:ends-with(xsd:string(?SCT), "/" + ?SCTid))
	
	}
	
	UNION
	
	{ GRAPH <file:OWLfile2 > {
	
	?x rdf:type oban:association ; oban:association_has_object ?C ;
	oban:association_has_subject ustg:CUItargetTerm .
	
	?C rdfs:subClassOf ?ST ; ustg:hasDbXrefInSCT ?SCTid .
	
	?y rdf:type oban:provenance ; obo:IAO_0000136 ?x ; obo:RO_0002558 ?lbV .
	
	FILTER (**(?ST = ustg:T059 || ?ST = ustg:T060 || ?ST = ustg:T061 || ?ST = ustg:T058)**
	&& (?lbV = OBANmod:OBANmod_2 || ?lbV = OBANmod:OBANmod_3 ||
	?lbV = OBANmod:OBANmod_4 || ?lbV = OBANmod:OBANmod_5))
	}
	
	?SCT a owl:Class .
	
	FILTER (fn:ends-with(xsd:string(?SCT), "/" + ?SCTid))
	
	} } }

q2VS(OWLfile1, OWLfile2, CUItargetTerm, OWLtargetTermModule)	SELECT (COUNT(DISTINCT ?SCT) AS ?count)
	FROM <file:OWLtargetTermModule>
	FROM NAMED <file:OWLfile1>
	FROM NAMED <file:OWLfile2>
	{ { GRAPH <file:OWLfile1> {
	
	?x rdf:type oban:association ; oban:association_has_object ?C ;
	oban:association_has_subject ustg:CUItargetTerm .
	
	?C rdfs:subClassOf ?ST ; ustg:hasDbXrefInSCT ?SCTid .
	
	?y rdf:type oban:provenance ; obo:IAO_0000136 ?x ; obo:RO_0002558 ?lbV .
	
	FILTER (**(?ST = ustg:T034 || ?ST = ustg:T184 || ?ST = ustg:T033)**
	&& (?lbV = OBANmod:OBANmod_2 || ?lbV = OBANmod:OBANmod_3 ||
	?lbV = OBANmod:OBANmod_4 || ?lbV = OBANmod:OBANmod_5))
	}
	
	?SCT a owl:Class .
	
	FILTER (fn:ends-with(xsd:string(?SCT), "/" + ?SCTid))
	
	}
	
	UNION
	
	{ GRAPH <file:OWLfile2 > {
	
	?x rdf:type oban:association ; oban:association_has_object ?C ;
	oban:association_has_subject ustg:CUItargetTerm .
	
	?C rdfs:subClassOf ?ST ; ustg:hasDbXrefInSCT ?SCTid .
	
	?y rdf:type oban:provenance ; obo:IAO_0000136 ?x ; obo:RO_0002558 ?lbV .
	
	FILTER (**(?ST = ustg:T034 || ?ST = ustg:T184 || ?ST = ustg:T033)**
	&& (?lbV = OBANmod:OBANmod_2 || ?lbV = OBANmod:OBANmod_3 ||
	?lbV = OBANmod:OBANmod_4 || ?lbV = OBANmod:OBANmod_5))
	
	}
	
	?SCT a owl:Class .
	
	FILTER (fn:ends-with(xsd:string(?SCT), "/" + ?SCTid))
	
	} }

q3VS(OWLfile1, OWLfile2, CUItargetTerm, OWLtargetTermModule)	SELECT (COUNT(DISTINCT ?SCT) AS ?count)
	FROM <file:OWLtargetTermModule>
	FROM NAMED <file:OWLfile1>
	FROM NAMED <file:OWLfile2>
	{ { GRAPH <file:OWLfile1> {
	
	?x rdf:type oban:association ; oban:association_has_object ?C ;
	oban:association_has_subject ustg:CUItargetTerm .
	
	?C rdfs:subClassOf ?ST ; ustg:hasDbXrefInSCT ?SCTid .
	
	?ST rdf:type owl:Class ; rdfs:subClassOf [a owl:Restriction ;
	owl:onProperty obo:BFO_0000050; owl:someValuesFrom ?SG] .
	
	?y rdf:type oban:provenance ; obo:IAO_0000136 ?x ; obo:RO_0002558 ?lbV .
	
	FILTER (**?SG = ustg:CHEM**
	&& (?lbV = OBANmod:OBANmod_2 || ?lbV = OBANmod:OBANmod_3 ||
	?lbV = OBANmod:OBANmod_4 || ?lbV = OBANmod:OBANmod_5))
	
	}
	
	?SCT a owl:Class .
	
	FILTER (fn:ends-with(xsd:string(?SCT), "/" + ?SCTid))
	
	}
	
	UNION
	
	{ GRAPH <file:OWLfile2 > {
	
	?x rdf:type oban:association ; oban:association_has_object ?C ;
	oban:association_has_subject ustg:CUItargetTerm .
	
	?C rdfs:subClassOf ?ST ; ustg:hasDbXrefInSCT ?SCTid .
	
	?ST rdf:type owl:Class ; rdfs:subClassOf [a owl:Restriction ;
	owl:onProperty obo:BFO_0000050; owl:someValuesFrom ?SG] .
	
	?y rdf:type oban:provenance ; obo:IAO_0000136 ?x ; obo:RO_0002558 ?lbV .
	
	FILTER (**?SG = ustg:CHEM**
	&& (?lbV = OBANmod:OBANmod_2 || ?lbV = OBANmod:OBANmod_3 ||
	?lbV = OBANmod:OBANmod_4 || ?lbV = OBANmod:OBANmod_5))
	
	}
	
	?SCT a owl:Class .
	
	FILTER (fn:ends-with(xsd:string(?SCT), "/" + ?SCTid))
	
	} }

The following three SPARQL SELECT queries exploit: a) the inferred information obtained (with the reasoner FaCT++) from the axioms within the locality-based modules created per target term from the SNOMED CT ontology; and b) the OBANmod core ontology once it is populated for both datasets, i.e. the VetCN dataset (the OWL file 1) and the PMSB dataset (the OWL file 2). The SELECT SPARQL queries q1VR and q2VR have in bold the UMLS Semantic Types for them. The SELECT SPARQL query q3VR has in bold the UMLS Semantic Group to which is applied, i.e. CHEM.

Query(parameters)	**SPARQL SELECT Query**

q1VR(OWLfile1, OWLfile2, CUItargetTerm, inferred_ OWLtargetTermModule)	SELECT (COUNT(DISTINCT ?SCTdes) AS ?count)
	FROM <file:inferredOWLtargetTermModule>
	FROM NAMED <file:OWLfile1>
	FROM NAMED <file:OWLfile2>
	{ { GRAPH <file:OWLfile1> {
	
	?x rdf:type oban:association ; oban:association_has_object ?C ;
	oban:association_has_subject ustg:CUItargetTerm .
	
	?C rdfs:subClassOf ?ST ; ustg:hasDbXrefInSCT ?SCTid .
	
	?y rdf:type oban:provenance ; obo:IAO_0000136 ?x ; obo:RO_0002558 ?lbV .
	
	FILTER (**(?ST = ustg:T059 || ?ST = ustg:T060 || ?ST = ustg:T061 || ?ST = ustg:T058)**
	&& (?lbV = OBANmod:OBANmod_2 || ?lbV = OBANmod:OBANmod_3 || ?lbV = OBANmod:OBANmod_4 || ?lbV = OBANmod:OBANmod_5))
	}
	
	?SCTdes rdfs:subClassOf+ ?SCT .
	
	FILTER (fn:ends-with(xsd:string(?SCT), "/" + ?SCTid))
	
	}
	
	UNION
	
	{ GRAPH <file:OWLfile2 > {
	
	?x rdf:type oban:association ; oban:association_has_object ?C ;
	oban:association_has_subject ustg:CUItargetTerm .
	
	?C rdfs:subClassOf ?ST ;ustg:hasDbXrefInSCT ?SCTid .
	
	?y rdf:type oban:provenance ; obo:IAO_0000136 ?x ; obo:RO_0002558 ?lbV .
	
	FILTER (**(?ST = ustg:T059 || ?ST = ustg:T060 || ?ST = ustg:T061 || ?ST = ustg:T058)**
	&& (?lbV = OBANmod:OBANmod_2 || ?lbV = OBANmod:OBANmod_3 || ?lbV = OBANmod:OBANmod_4 || ?lbV = OBANmod:OBANmod_5))
	}
	
	?SCTdes rdfs:subClassOf+ ?SCT .
	
	FILTER (fn:ends-with(xsd:string(?SCT), "/" + ?SCTid))
	
	} } }

q2VR(OWLfile1, OWLfile2, CUItargetTerm, inferred_ OWLtargetTermModule)	SELECT (COUNT(DISTINCT ?SCTdes) AS ?count)
	FROM <file:inferredOWLtargetTermModule>
	FROM NAMED <file:OWLfile1>
	FROM NAMED <file:OWLfile2>
	{ { GRAPH <file:OWLfile1> {
	
	?x rdf:type oban:association ; oban:association_has_object ?C ;
	oban:association_has_subject ustg:CUItargetTerm .
	
	?C rdfs:subClassOf ?ST ; ustg:hasDbXrefInSCT ?SCTid .
	
	?y rdf:type oban:provenance ; obo:IAO_0000136 ?x ; obo:RO_0002558 ?lbV .
	
	FILTER (**(?ST = ustg:T034 || ?ST = ustg:T184 || ?ST = ustg:T033)**
	&& (?lbV = OBANmod:OBANmod_2 || ?lbV = OBANmod:OBANmod_3 ||
	?lbV = OBANmod:OBANmod_4 || ?lbV = OBANmod:OBANmod_5))
	}
	
	?SCTdes rdfs:subClassOf+ ?SCT .
	
	FILTER (fn:ends-with(xsd:string(?SCT), "/" + ?SCTid))
	
	}
	
	UNION
	
	{ GRAPH <file:OWLfile2 > {
	
	?x rdf:type oban:association ; oban:association_has_object ?C ;
	oban:association_has_subject ustg:CUItargetTerm .
	
	?C rdfs:subClassOf ?ST ; ustg:hasDbXrefInSCT ?SCTid .
	
	?y rdf:type oban:provenance ; obo:IAO_0000136 ?x ; obo:RO_0002558 ?lbV .
	
	FILTER (**(?ST = ustg:T034 || ?ST = ustg:T184 || ?ST = ustg:T033)**
	&& (?lbV = OBANmod:OBANmod_2 || ?lbV = OBANmod:OBANmod_3 ||
	?lbV = OBANmod:OBANmod_4 || ?lbV = OBANmod:OBANmod_5))
	
	}
	
	?SCTdes rdfs:subClassOf+ ?SCT .
	
	FILTER (fn:ends-with(xsd:string(?SCT), "/" + ?SCTid))
	
	} }

q3VR(OWLfile1, OWLfile2, CUItargetTerm, inferred_ OWLtargetTermModule)	SELECT (COUNT(DISTINCT ?SCTdes) AS ?count)
	FROM <file:inferredOWLtargetTermModule>
	FROM NAMED <file:OWLfile1>
	FROM NAMED <file:OWLfile2>
	{ { GRAPH <file:OWLfile1> {
	
	?x rdf:type oban:association ; oban:association_has_object ?C ;
	oban:association_has_subject ustg:CUItargetTerm .
	
	?C rdfs:subClassOf ?ST ; ustg:hasDbXrefInSCT ?SCTid .
	
	?ST rdf:type owl:Class ; rdfs:subClassOf [a owl:Restriction ;
	owl:onProperty obo:BFO_0000050; owl:someValuesFrom ?SG] .
	
	?y rdf:type oban:provenance ; obo:IAO_0000136 ?x ; obo:RO_0002558 ?lbV .
	
	FILTER (**?SG = ustg:CHEM**
	&& (?lbV = OBANmod:OBANmod_2 || ?lbV = OBANmod:OBANmod_3 ||
	?lbV = OBANmod:OBANmod_4 || ?lbV = OBANmod:OBANmod_5))
	
	}
	
	?SCTdes rdfs:subClassOf+ ?SCT .
	
	FILTER (fn:ends-with(xsd:string(?SCT), "/" + ?SCTid))
	
	}
	
	UNION
	
	{ GRAPH <file:OWLfile2 > {
	
	?x rdf:type oban:association ; oban:association_has_object ?C ;
	oban:association_has_subject ustg:CUItargetTerm .
	
	?C rdfs:subClassOf ?ST ; ustg:hasDbXrefInSCT ?SCTid .
	
	?ST rdf:type owl:Class ; rdfs:subClassOf [a owl:Restriction ;
	owl:onProperty obo:BFO_0000050; owl:someValuesFrom ?SG] .
	
	?y rdf:type oban:provenance ; obo:IAO_0000136 ?x ; obo:RO_0002558 ?lbV .
	
	FILTER (**?SG = ustg:CHEM**
	&& (?lbV = OBANmod:OBANmod_2 || ?lbV = OBANmod:OBANmod_3 ||
	?lbV = OBANmod:OBANmod_4 || ?lbV = OBANmod:OBANmod_5))
	
	}
	
	?SCTdes rdfs:subClassOf+ ?SCT .
	
	FILTER (fn:ends-with(xsd:string(?SCT), "/" + ?SCTid))
	
	} }