# Prefixes

PREFIX rdfs:<http://www.w3.org/2000/01/rdf-schema#>

PREFIX dc-term:<http://purl.org/dc/terms/>

PREFIX rdf:<http://www.w3.org/1999/02/22-rdf-syntax-ns#>

PREFIX dc:<http://purl.org/dc/elements/1.1/>

PREFIX foaf:<http://xmlns.com/foaf/0.1/>

PREFIX xsd:<http://www.w3.org/2001/XMLSchema#>

PREFIX owl:<http://www.w3.org/2002/07/owl#>

PREFIX oup:<http://oxfordindex.oup.com/metadata/>

Stock SPARQL Queries

# Get Auto-Approved Links (approved by Accuracy)

SELECT (COUNT(DISTINCT ?link) as ?count)

WHERE{

?link a oup:isPrimaryTopicOfLink;

oup:hasMatchStatus "approved";

oup:hasAccuracy ?acc.

FILTER regex(str(?link), "benz")

FILTER (xsd:integer(?acc) >= 50)

}

# Get New SameEntityAs Records by Product Pair via Regex

## ALL

SELECT DISTINCT ?record1 ?record2

WHERE{

?auth a oup:Authority;

oup:hasLink ?plink1;

oup:hasLink ?plink2.

?plink1 a oup:isPrimaryTopicOfLink.

?plink2 a oup:isPrimaryTopicOfLink.

FILTER (?plink1 != ?plink2)

FILTER regex(str(?plink1), "10\\.1093\\/benz")

FILTER regex(str(?plink2), "10\\.1093\\/www?")

?plink1 oup:hasTarget ?record1.

?plink2 oup:hasTarget ?record2.

FILTER NOT EXISTS {

?record1 oup:hasLink/rdf:type oup:sameEntityAsLink.

?record2 oup:hasLink/rdf:type oup:sameEntityAsLink.

?record1 ^oup:hasTarget/rdf:type oup:sameEntityAsLink.

?record2 ^oup:hasTarget/rdf:type oup:sameEntityAsLink.

}

}

## Approved

SELECT DISTINCT ?record1 ?record2

WHERE{

?auth a oup:Authority;

oup:hasLink ?plink1;

oup:hasLink ?plink2.

?plink1 a oup:isPrimaryTopicOfLink;

oup:hasMatchStatus “approved”.

?plink2 a oup:isPrimaryTopicOfLink;

oup:hasMatchStatus “approved”.

FILTER (?plink1 != ?plink2)

FILTER regex(str(?plink1), "10\\.1093\\/benz")

FILTER regex(str(?plink2), "10\\.1093\\/www?")

?plink1 oup:hasTarget ?record1.

?plink2 oup:hasTarget ?record2.

FILTER NOT EXISTS {

?record1 oup:hasLink/rdf:type oup:sameEntityAsLink.

?record2 oup:hasLink/rdf:type oup:sameEntityAsLink.

?record1 ^oup:hasTarget/rdf:type oup:sameEntityAsLink.

?record2 ^oup:hasTarget/rdf:type oup:sameEntityAsLink.

}

}

# Get New SameEntityAs Records by Product Pair via ProductCode

## ALL

### LongForm All

SELECT DISTINCT ?record1 ?record2

WHERE{

?auth a oup:Authority;

oup:hasLink ?plink1;

oup:hasLink ?plink2.

?plink1 a oup:isPrimaryTopicOfLink.

?plink2 a oup:isPrimaryTopicOfLink.

FILTER (!sameTerm(?plink1, ?plink2))

?plink1 oup:hasTarget ?record1.

?record1 oup:hasProductCode "ANB ".

?plink2 oup:hasTarget ?record2.

?record2 oup:hasProductCode "OAO ".

FILTER NOT EXISTS {

?record1 oup:hasLink ?sameLink1.

?sameLink1 a oup:sameEntityAsLink;

oup:hasTarget ?record2.

?record2 oup:hasLink ?sameLink2.

?sameLink2 a oup:sameEntityAsLink;

oup:hasTarget ?record1.

}

}

## Approved

### Longform Approved

SELECT DISTINCT ?record1 ?record2

WHERE{

?auth a oup:Authority;

oup:hasLink ?plink1;

oup:hasLink ?plink2.

?plink1 a oup:isPrimaryTopicOfLink;

oup:hasMatchStatus "approved ".

?plink2 a oup:isPrimaryTopicOfLink;

oup:hasMatchStatus "approved ".

FILTER (!sameTerm(?plink1, ?plink2))

?plink1 oup:hasTarget ?record1.

?record1 oup:hasProductCode "ANB ".

?plink2 oup:hasTarget ?record2.

?record2 oup:hasProductCode "OAO ".

FILTER NOT EXISTS {

?record1 oup:hasLink ?sameLink1.

?sameLink1 a oup:sameEntityAsLink;

oup:hasTarget ?record2.

?record2 oup:hasLink ?sameLink2.

?sameLink2 a oup:sameEntityAsLink;

oup:hasTarget ?record1.

}

}

# Get New SameEntityAs Records

SELECT DISTINCT ?record1 ?record2

WHERE{

?auth a oup:Authority;

oup:hasLink ?plink1;

oup:hasLink ?plink2.

?plink1 a oup:isPrimaryTopicOfLink.

?plink2 a oup:isPrimaryTopicOfLink.

FILTER (!sameTerm(?plink1, ?plink2))

?plink1 oup:hasTarget ?record1.

?plink2 oup:hasTarget ?record2.

FILTER NOT EXISTS {

?record1 oup:hasLink/rdf:type oup:sameEntityAsLink.

?record2 oup:hasLink/rdf:type oup:sameEntityAsLink.

?record1 ^oup:hasTarget/rdf:type oup:sameEntityAsLink.

?record2 ^oup:hasTarget/rdf:type oup:sameEntityAsLink.

}

}

## Linux Direct Version

SELECT DISTINCT ?auth ?record1 ?record2

WHERE{

?auth a oup:Authority;

oup:hasLink ?plink1;

oup:hasLink ?plink2.

?plink1 a oup:isPrimaryTopicOfLink;

oup:hasMatchStatus "approved".

?plink2 a oup:isPrimaryTopicOfLink;

oup:hasMatchStatus "approved".

FILTER (!sameTerm(?plink1, ?plink2))

?plink1 oup:hasTarget ?record1.

?plink2 oup:hasTarget ?record2.

FILTER NOT EXISTS {

?record1 oup:hasLink ?same1.

?same1 a oup:sameEntityAsLink;

oup:hasTarget ?record2.

?record2 oup:hasLink ?same2.

?same2 a oup:sameEntityAsLink;

oup:hasTarget ?record1.

}

}

PREFIX oup:<http://oxfordindex.oup.com/metadata/> SELECT DISTINCT ?auth ?record1 ?record2 WHERE{?auth a oup:Authority; oup:hasLink ?plink1; oup:hasLink ?plink2. ?plink1 a oup:isPrimaryTopicOfLink; oup:hasMatchStatus "approved". ?plink2 a oup:isPrimaryTopicOfLink; oup:hasMatchStatus "approved". FILTER (!sameTerm(?plink1, ?plink2)) ?plink1 oup:hasTarget ?record1. ?plink2 oup:hasTarget ?record2. FILTER NOT EXISTS {?record1 oup:hasLink ?same1. ?same1 a oup:sameEntityAsLink; oup:hasTarget ?record2. ?record2 oup:hasLink ?same2. ?same2 a oup:sameEntityAsLink; oup:hasTarget ?record1.}}.

### Get Count of the same

SELECT (COUNT(DISTINCT \*) AS ?count){

SELECT DISTINCT ?record1 ?record2

WHERE{

?auth a oup:Authority;

oup:hasLink ?plink1;

oup:hasLink ?plink2.

?plink1 a oup:isPrimaryTopicOfLink.

?plink2 a oup:isPrimaryTopicOfLink.

FILTER (!sameTerm(?plink1, ?plink2))

?plink1 oup:hasTarget ?record1.

?plink2 oup:hasTarget ?record2.

FILTER NOT EXISTS {

?record1 oup:hasLink/rdf:type oup:sameEntityAsLink.

?record2 oup:hasLink/rdf:type oup:sameEntityAsLink.

?record1 ^oup:hasTarget/rdf:type oup:sameEntityAsLink.

?record2 ^oup:hasTarget/rdf:type oup:sameEntityAsLink.

}

}

}

# Get Existing SameEntityAs Links

### Longform Approved

SELECT DISTINCT ?same1

WHERE{

?record1 a oup:Record;

oup:hasProductCode "ANB";

oup:hasLink ?same1.

?same1 a oup:sameEntityAsLink;

oup:hasMatchStatus "approved";

oup:hasTarget ?record2.

?record2 oup:hasProductCode "OAO".

}

# GET SAME ENTITY AS BY DISTINCT PRODUCT

SELECT DISTINCT ?prodSource ?prodTarg

WHERE {

?same a oup:sameEntityAsLink;

oup:hasTarget ?recTarg.

?recTarg oup:hasProductCode ?prodTarg.

?recSource oup:hasLink ?same.

?recSource oup:hasProductCode ?prodSource.

}

# DELETE SELF LINKING SAME ENTITY AS

DELETE{

?same rdf:type oup:sameEntityAsLink.

?same oup:hasTarget ?recTarget.

?recSource oup:hasLink ?same.

}

WHERE{

?same a oup:sameEntityAsLink;

oup:hasTarget ?recTarget.

?recSource oup:hasLink ?same.

FILTER ( ?recSource = ?recTarget)

}

# Create Same Entity As

MAKE SURE TO CHECK THE “FILTER NOT EXIST” PART GOING BOTH WAYS!!!!!

This SPARQL only checks to make sure no sameAs link already exists from Source to Target, but a second query is needed to check from Target to Source because today 2016-08-25 I discovered Atticus is reciprocating sameAs links dynamically, despite that we were told it doesn’t.

## TEST

SELECT ?same ?recSource ?recTarget

WHERE{

?auth a oup:Authority;

oup:hasLink ?plink1;

oup:hasLink ?plink2.

?plink1 a oup:isPrimaryTopicOfLink;

oup:hasMatchStatus "approved".

?plink2 a oup:isPrimaryTopicOfLink;

oup:hasMatchStatus "approved".

FILTER (!sameTerm(?plink1, ?plink2))

?plink1 oup:hasTarget ?recSource.

?recSource oup:hasProductCode "ANB".

?plink2 oup:hasTarget ?recTarget.

?recTarget oup:hasProductCode "OAO".

FILTER NOT EXISTS {

?recSource oup:hasLink ?sameLink1.

?sameLink1 a oup:sameEntityAsLink;

oup:hasTarget ?recTarget.

}

BIND(IRI(CONCAT(STR(?recSource), "~sameEntityAs~",SUBSTR(STR(?recTarget),37)))

AS ?same)

}

### Links between GAO and Benezit

They don’t appear to be approved yet.

SELECT ?same ?recSource ?recTarget

WHERE{

?auth a oup:Authority;

oup:hasLink ?plink1;

oup:hasLink ?plink2.

?plink1 a oup:isPrimaryTopicOfLink.

?plink2 a oup:isPrimaryTopicOfLink.

FILTER (!sameTerm(?plink1, ?plink2))

?plink1 oup:hasTarget ?recSource.

?recSource oup:hasProductCode "OAO".

FILTER regex(str(?recSource) , "gao")

?plink2 oup:hasTarget ?recTarget.

?recTarget oup:hasProductCode "OAO".

FILTER regex(str(?recTarget) , "benz")

FILTER NOT EXISTS {

?recSource oup:hasLink ?sameLink1.

?sameLink1 a oup:sameEntityAsLink;

oup:hasTarget ?recTarget.

}

BIND(IRI(CONCAT(STR(?recSource), "~sameEntityAs~",SUBSTR(STR(?recTarget),37)))

AS ?same)

}

## UPDATE

INSERT{

?same rdf:type oup:sameEntityAsLink.

?same oup:hasTarget ?recTarget.

?same oup:hasProvenanceOrigin "mcdonalds".

?same oup:hasProvenanceDate "2016-08-25T00:00:00.000".

?recSource oup:hasLink ?same.

}

WHERE{

?auth a oup:Authority;

oup:hasLink ?plink1;

oup:hasLink ?plink2.

?plink1 a oup:isPrimaryTopicOfLink;

oup:hasMatchStatus "approved".

?plink2 a oup:isPrimaryTopicOfLink;

oup:hasMatchStatus "approved".

FILTER (!sameTerm(?plink1, ?plink2))

?plink1 oup:hasTarget ?recSource.

?recSource oup:hasProductCode "ANB".

?plink2 oup:hasTarget ?recTarget.

?recTarget oup:hasProductCode "OAO".

FILTER NOT EXISTS {

?recSource oup:hasLink ?sameLink1.

?sameLink1 a oup:sameEntityAsLink;

oup:hasTarget ?recTarget.

}

BIND(IRI(CONCAT(STR(?recSource), "~sameEntityAs~",SUBSTR(STR(?recTarget),37))) AS ?same)

}

# Get ODNB Links To Groves

SELECT ?record1

WHERE {

{

?record1 a oup:Record.

?record1 oup:hasProductCode "ODNB".

?record1 oup:hasLink ?link1.

?link1 a oup:sameEntityAsLink.

?link1 oup:hasTarget ?record2.

?record2 oup:hasProductCode "OMO".

}

UNION

{

?record1 a oup:Record.

?record1 oup:hasProductCode "ODNB".

?record1 oup:hasLink ?link1.

?link1 a oup:sameEntityAsLink.

?link1 oup:hasTarget ?record2.

?record2 oup:hasProductCode "OAO".

}

}

# Get Authorities with only one isPrimaryTopicOf Link

SELECT ?auth (COUNT(DISTINCT ?link) AS ?count)

WHERE {

?auth a oup:Authority.

?auth oup:hasLink ?link.

}

GROUP BY ?auth

HAVING (COUNT(DISTINCT ?link) = 1)

## Filter

Only those made in 2014.

FILTER regex(str(?auth), "\\.2014")

# Get Authorities with 3+ isPrimaryTopicOf Links

10.1093/oi/authority.20110803095342715

10.1093/oi/authority.20110803095343523

10.1093/oi/authority.20110803095432721

10.1093/oi/authority.20110803095431849

10.1093/oi/authority.20110803095343215

10.1093/oi/authority.20110803095431799

10.1093/oi/authority.20110803095343129

10.1093/oi/authority.20110803095343127

10.1093/oi/authority.20110803095343104

10.1093/oi/authority.20110803095342719

SELECT DISTINCT?authority

Where{

?authority a oup:Authority.

?authority oup:hasLink ?link1.

?link1 a oup:isPrimaryTopicOfLink.

?authority oup:hasLink ?link2.

?link2 a oup:isPrimaryTopicOfLink.

?authority oup:hasLink ?link3.

?link3 a oup:isPrimaryTopicOfLink.

FILTER (?link1 != ?link2 && ?link1 != ?link3 && ?link2 != ?link3)

} LIMIT 100

# Get Records with Multiple PrimaryTopic Links

SELECT (count(?doi) as ?countDOI)

WHERE {

?link1 oup:hasTarget ?doi.

?link1 a oup:isPrimaryTopicOfLink.

?link2 oup:hasTarget ?doi.

?link2 a oup:isPrimaryTopicOfLink.

FILTER (?link1 != ?link2)

}ORDER BY ?link1

SELECT ?doi

WHERE {

?link1 oup:hasTarget ?doi.

?link1 a oup:isPrimaryTopicOfLink.

?link2 oup:hasTarget ?doi.

?link2 a oup:isPrimaryTopicOfLink.

?link3 oup:hasTarget ?doi.

?link3 a oup:isPrimaryTopicOfLink.

?link4 oup:hasTarget ?doi.

?link4 a oup:isPrimaryTopicOfLink.

FILTER (?link1 != ?link2 && ?link1 != ?link3 && ?link2 != ?link3 && ?link4 != ?link1 && ?link4 && ?link2 && ?link4 != ?link3)

}

# Get PrimaryTopic Target

SELECT ?doi1

where {

<http://oxfordindex.oup.com/metadata/10.1093/oi/authority.20110803095413160> oup:hasLink ?link1.

?link1 a oup:isPrimaryTopicOfLink.

?link1 oup:hasTarget ?doi1.

}

# Get isRelatedTo Authorities

SELECT ?auth1 ?auth2

WHERE{

?auth1 oup:hasLink ?link1.

?link1 a oup:isPrimaryTopicOfLink.

?link1 oup:hasTarget ?content\_record1.

?link2 oup:hasTarget ?content\_record1.

?link2 a oup:referenceLink.

?content\_record2 oup:hasLink ?link2.

?auth2 oup:hasLink ?link3.

?link3 a oup:isPrimaryTopicOfLink.

?link3 oup:hasTarget ?content\_record2.

FILTER(!sameTerm(?auth1, ?auth2))

}

GROUP BY ?auth1 ?content\_record2 ?auth2

HAVING(COUNT(?content\_record1) > 2)

SELECT ?auth1 ?auth2

WHERE{

?auth1 oup:hasLink ?link1.

?link1 a oup:isPrimaryTopicOfLink.

?link1 oup:hasTarget ?content\_record1.

?content\_record1 oup:hasLink ?link2.

?link2 a oup:referenceLink.

?link2 oup:hasTarget ?content\_record2.

?content\_record2 oup:hasLink ?link2.

?auth2 oup:hasLink ?link3.

?link3 a oup:isPrimaryTopicOfLink.

?link3 oup:hasTarget ?content\_record2.

FILTER(!sameTerm(?auth1, ?auth2))

}

GROUP BY ?auth1 ?content\_record2 ?auth2

HAVING(COUNT(?content\_record1) > 2)