MIE1501 Project Part 2

Team 6  
Qisheng He 999502275

Abhishek Das 1006374353

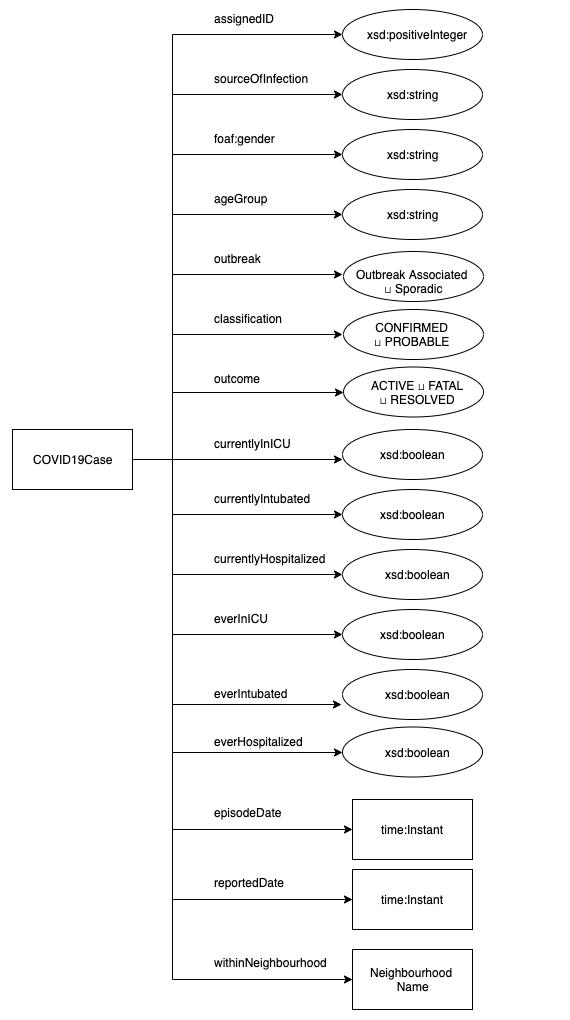
# Revised Ontology

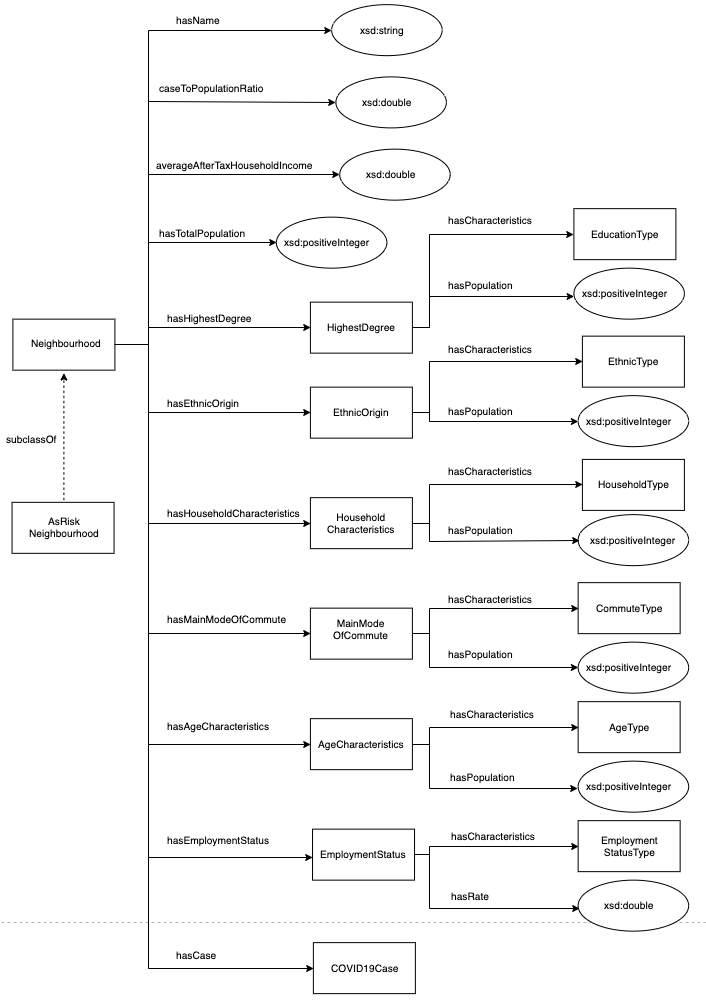
We modified our ontology based on the feedback from part 1. The major change was to explicitly define the demographic classes (e.g. Highest Degree, Ethnic Origin etc) as a union of all acceptable individuals. Previously they were defined as Literals which led to low fidelity.

Another change was to remove the “foaf:Person” class from our ontology, and only keep the “foaf:gender” property that we used.

We also imported more instances from the csv files using the Cellfie plugin. For more information please refer to their github link <https://github.com/protegeproject/cellfie-plugin>.

With all the updates, now the ontology is as illustrated in the graphs below.





# 1.1 Description Logic

The classes are listed below, and a design graph is attached in the next page to illustrate the structure.

Neighbourhood ⊑ Thing

AtRiskNeighbourhood ⊑ Neighbourhood

COVID19Case ⊑ Thing

HighestDegree ⊑ Demographics

EthnicOrigin ⊑ Demographics

HouseholdCharacteristics ⊑ Demographics

MainModeOfCommute ⊑ Demographics

AgeCharacteristics ⊑ Demographics

EmploymentStatus ⊑ Demographics

The classes are defined in description logic below. All classes are primitive classes because all the properties are necessary for them but not sufficient. For Neighbourhood classes we have added hasCase property since the objective requires us to incorporate COVID19 cases.

Neighbourhood ⊑ =1 hasTotalPopulation.xsd:positiveInteger

⊓ =1 hasName.xsd:string

⊓ ∃ hasCase.COVID19Case

⊓ =1 COVID19CaseToPopulationRatio.xsd:double

⊓ ∃ hasEducation.HighestDegree

⊓ ∃ hasEthnicOrigin.EthnicOrigin

⊓ ∃ hasHouseholdCharacteristics.HouseholdCharacteristics

⊓ =1 averageAfterTaxHouseholdIncome.xsd:double

⊓ ∃ hasMainModeOfCommute.MainModeOfCommute

⊓ ∃ hasAgeCharacteristics.AgeCharacteristics

⊓ ∃ hasEmploymentStatus.EmploymentStatus

**Note:** The calculation of COVID19CaseToPopulationRatio will be done in Python and output as Turtle and the Turtle file will then be evaluated for competency questions using Apache Fuseki and SparQL.

AtRiskNeighbourhood is a defined class since we need the HermiT reasoner to infer if a neighbour is at risk or not. The definition for this class is a neighbour with a COVID19CaseToPopulationRatio greater than a certain threshold value.

AtRiskNeighbourhood ≡ Neighbourhood

⊓ >=r COVID19CaseToPopulationRatio

**Note:** r is a pre-defined threshold ratio. In our ontology we assume r to be 0.05. It can be adjusted in the future.

The COVID19Case is a primitive class since the properties mentioned below are necessary based on the data source provided but not sufficient for COVID19Case classsfication.

COVID19Case ⊑ =1 assignedID.xsd:positiveInteger

⊓ <=1 sourceOfInfection.rdfs:Literal

⊓ <=1 ageGroup.rdfs:Literal

⊓ =1 outbreak.({Sporadic} ⊔ {Outbreak\_Associated})

⊓ =1 classification.({CONFIRMED} ⊔ {PROBABLE})

⊓ =1 episodeDate.time:Instant

⊓ =1 reportedTime.time:Instant

⊓ =1 outcome.({ACTIVE} ⊔ {FATAL} ⊔ {RESOLVED})

⊓ <=1 foaf:gender.rdfs:Literal

⊓ =1 currentlyHospitalized.xsd:boolean

⊓ =1 currentlyInICU.xsd:boolean

⊓ =1 currentlyIntubated.xsd:boolean

⊓ =1 everHospitalized.xsd:boolean

⊓ =1 everInICU.xsd:boolean

⊓ =1 everIntubated.xsd:boolean

⊓ =1 inNeighbourhood.Neighbourhood

Demographics is a subset of top owl class Thing.

Demographics ⊑ Thing

HighestDegree is a subset of Demographics and has exactly one characteristics and each distinct characteristic has exactly one population value.

HighestDegree ⊑ Demographics

⊓ =1 hasCharacteristics.EducationType

⊓ =1 population.xsd:positiveInteger

EducationType ≡ {'University\_certificate,\_diploma\_or\_degree\_at\_bachelor\_level\_or\_above'} ⊔ {Earned\_doctorate} ⊔ {Trades\_certificate\_or\_diploma\_other\_than\_Certificate\_of\_Apprenticeship\_or\_Certificate\_of\_Qualification} ⊔

{'Postsecondary\_certificate,\_diploma\_or\_degree'} ⊔

{'College,\_CEGEP\_or\_other\_non-university\_certificate\_or\_diploma'} ⊔

{University\_certificate\_or\_diploma\_below\_bachelor\_level} ⊔

{Apprenticeship\_or\_trades\_certificate\_or\_diploma} ⊔

{Certificate\_of\_Apprenticeship\_or\_Certificate\_of\_Qualification} ⊔

{'Degree\_in\_medicine,\_dentistry,\_veterinary\_medicine\_or\_optometry'} ⊔

{University\_certificate\_or\_diploma\_above\_bachelor\_level} ⊔

{'Secondary\_(high)\_school\_diploma\_or\_equivalency\_certificate'} ⊔

{Masters\_degree} ⊔

{Bachelors\_degree} ⊔

{'No\_certificate,\_diploma\_or\_degree'}

EthnicOrigin is a subset of Demographics and exactly one characteristics and each distinct characteristic has exactly one population value.

EthnicOrigin ⊑ Demographics

⊓ =1 hasCharacteristics.EthnicType

⊓ =1 population.xsd:positiveInteger

EthnicType ≡ {NorthAmericanAboriginalOrigins} ⊔ {'FirstNations(NorthAmericanIndian)'} ⊔ {Inuit} ⊔ {Mtis} ⊔ {OtherNorthAmericanOrigins} ⊔ {Acadian} ⊔ {American} ⊔ {Canadian} ⊔ {NewBrunswicker} ⊔ Newfoundlander} ⊔ {NovaScotian} ⊔ {Ontarian} ⊔ {Qubcois} ⊔ {Portuguese} ⊔ {OtherNorthAmericanorigins;n.i.e.} ⊔ {EuropeanOrigins} ⊔ {BritishIslesOrigins} ⊔ {ChannelIslander} ⊔ {Cornish} ⊔ {English} ⊔ {Irish} ⊔ {Manx} ⊔ {Welsh} ⊔ {BritishIslesorigins;n.i.e.} ⊔ {FrenchOrigins} ⊔ {Alsatian} ⊔ {Breton} ⊔ {Corsican} ⊔ {French} ⊔

{'WesternEuropeanOrigins(exceptFrenchorigins)'} ⊔ {Austrian} ⊔ {Bavarian} ⊔ {Belgian} ⊔ {Dutch} ⊔ {Flemish} ⊔ {Frisian} ⊔ {German} ⊔ {Luxembourger} ⊔ {Swiss} ⊔

{WesternEuropeanorigins;n.i.e.} ⊔{'NorthernEuropeanOrigins(exceptBritishIslesorigins)'} ⊔

{Danish} ⊔ {Finnish} ⊔ {Icelandic} ⊔ {Norwegian} ⊔ {Swedish} ⊔ {NorthernEuropeanorigins;n.i.e.} ⊔ {EasternEuropeanOrigins} ⊔ {Bulgarian} ⊔ {CaribbeanOrigins} ⊔ {Byelorussian} ⊔ {Czech} ⊔ {Czechoslovakian;n.o.s.} ⊔ {Estonian} ⊔

{Hungarian} ⊔ {Latvian} ⊔ {Lithuanian} ⊔ {Moldovan} ⊔ {Polish} ⊔ {Romanian} ⊔ {Russian} ⊔

{Slovak} ⊔ {Ukrainian} ⊔ {EasternEuropeanorigins;n.i.e.} ⊔ {SouthernEuropeanOrigins} ⊔

{Albanian} ⊔{Bosnian} ⊔ {Catalan} ⊔ {Croatian} ⊔ {Cypriot} ⊔ {Greek} ⊔ {Italian} ⊔

{Kosovar} ⊔ {Macedonian} ⊔ {Maltese} ⊔ {Montenegrin} ⊔ {Serbian} ⊔ {Sicilian} ⊔ {Slovenian} ⊔ {Spanish} ⊔ {Yugoslavian;n.o.s.} ⊔ {SouthernEuropeanorigins;n.i.e.} ⊔ {Antiguan} ⊔

{OtherEuropeanOrigins} ⊔ {Basque} ⊔ {Jewish} ⊔ {'Roma(Gypsy)'} ⊔ {Slavic;n.o.s.} ⊔ {OtherEuropeanorigins;n.i.e.} ⊔ {Bahamian} ⊔ {Barbadian} ⊔ {Bermudan} ⊔ {Carib} ⊔ {Cuban} ⊔ {Dominican} ⊔ {Grenadian} ⊔ {Haitian} ⊔ {Jamaican} ⊔ {Kittitian/Nevisian} ⊔ {Martinican} ⊔

{Montserratan} ⊔ {PuertoRican} ⊔ {St.Lucian} or {Arawak} or {Trinidadian/Tobagonian} ⊔

{Vincentian/Grenadinian} ⊔ {WestIndian;n.o.s.} ⊔ {Caribbeanorigins;n.i.e.} ⊔

{Latin;CentralAndSouthAmericanOrigins} ⊔ {'AboriginalFromCentral/SouthAmerica(exceptArawakAndMaya)'} ⊔ {Argentinian} ⊔

{Belizean} ⊔ {Bolivian} ⊔ {Brazilian} ⊔ {Chilean} ⊔ {Colombian} ⊔ {CostaRican} ⊔ {Ecuadorian} ⊔ {Guatemalan} ⊔ {Guyanese} ⊔ {Hispanic} ⊔ {Honduran} ⊔ {Maya} ⊔ {Beninese} ⊔ {Mexican} ⊔

{Nicaraguan} ⊔ {Panamanian} ⊔ {Paraguayan} ⊔ {Peruvian} ⊔ {Salvadorean} ⊔ {Uruguayan} ⊔

{Venezuelan} ⊔ {Latin;CentralAndSouthAmericanorigins;n.i.e.} ⊔ {AfricanOrigins} ⊔ {CentralAndWestAfricanOrigins} ⊔ {Akan} ⊔ {Angolan} ⊔ {Ashanti} ⊔ {Burkinabe} ⊔

{Cameroonian} ⊔ {Chadian} ⊔ {Congolese} ⊔ {Edo} ⊔ {Ewe} ⊔ {Gabonese} ⊔ {Gambian} ⊔

{Ghanaian} ⊔ {Guinean} ⊔ {Ibo} ⊔ {Ivorian} ⊔ {Liberian} ⊔ {Malian} ⊔ {Malink} ⊔

{Nigerian} ⊔{Peulh} ⊔ {Senegalese} ⊔ {SierraLeonean} ⊔ {Togolese} ⊔ {Wolof} ⊔

{Yoruba} ⊔ {CentralAndWestAfricanorigins;n.i.e.} ⊔ {Maure} ⊔ {NorthAfricanOrigins} ⊔

{Algerian} ⊔ {Berber} ⊔ {Coptic} ⊔ {Dinka} ⊔ {Egyptian} ⊔ {Libyan} ⊔ {Moroccan} ⊔

{Sudanese} ⊔ {Tunisian} ⊔ {NorthAfricanorigins;n.i.e.} ⊔ {SouthernAndEastAfricanOrigins} ⊔

{Afrikaner} ⊔ {Amhara} ⊔ {Bantu;n.o.s.} ⊔ {Burundian} ⊔ {Djiboutian} ⊔ {Eritrean} ⊔

{Ethiopian} ⊔ {Harari} ⊔ {Kenyan} ⊔ {Malagasy} ⊔ {Mauritian} ⊔ {Oromo} ⊔ {Rwandan} ⊔

{Seychellois} ⊔ {Somali} ⊔ {SouthAfrican} ⊔ {Tanzanian} ⊔ {Tigrian} ⊔ {Ugandan} ⊔ {Zambian} ⊔ {Zimbabwean} ⊔ {Zulu} ⊔ {SouthernAndEastAfricanorigins;n.i.e.} ⊔ {OtherAfricanOrigins} ⊔

{Black;n.o.s.} ⊔ {OtherAfricanorigins;n.i.e.} ⊔ {AsianOrigins} ⊔ {WestCentralAsianAndMiddleEasternOrigins} ⊔ {Afghan} ⊔ {Arab;n.o.s.} ⊔ {Armenian} ⊔

{Assyrian} ⊔ {Azerbaijani} ⊔ {Georgian} ⊔ {Hazara} ⊔ {Iranian} ⊔ {Hmong} ⊔ {Iraqi} ⊔

{Israeli} ⊔ {Jordanian} ⊔ {Kazakh} ⊔ {Kurd} ⊔ {Kuwaiti} ⊔ {Kyrgyz} ⊔ {Maori} ⊔ {Lebanese} ⊔

{Palestinian} ⊔ {Pashtun} ⊔ {SaudiArabian} ⊔ {Syrian} ⊔ {Tajik} ⊔ {Tatar} ⊔ {Turk} ⊔ {Turkmen} ⊔ {Uighur} ⊔ {Uzbek} ⊔ {Yemeni} ⊔ {WestCentralAsianAndMiddleEasternorigins;n.i.e.} ⊔

{SouthAsianOrigins} ⊔ {Bangladeshi} ⊔ {Bengali} ⊔ {Bhutanese} ⊔ {EastIndian} ⊔ {Goan} ⊔

{Gujarati} ⊔ {Kashmiri} ⊔ {Nepali} ⊔ {Pakistani} ⊔ {Punjabi} ⊔ {Sinhalese} ⊔ {SriLankan} ⊔

{Indonesian} ⊔ {Tamil} ⊔ {SouthAsianorigins;n.i.e.} ⊔ {EastAndSoutheastAsianOrigins} ⊔

{Burmese} ⊔ {'Cambodian(Khmer)'} ⊔ {Chinese} ⊔ {Filipino} ⊔ {Japanese} ⊔ {Karen} ⊔

{Korean} ⊔ {Laotian} ⊔ {Malaysian} ⊔ {Mongolian} ⊔ {Singaporean} ⊔ {Taiwanese} ⊔

{Thai} ⊔ {Tibetan} ⊔ {Vietnamese} ⊔ {EastAndSoutheastAsianorigins;n.i.e.} ⊔

{Hawaiian} ⊔ {OtherAsianOrigins} ⊔ {OtherAsianorigins;n.i.e.} ⊔ {OceaniaOrigins} ⊔

{Australian} ⊔ {NewZealander} ⊔ {PacificIslandsOrigins} ⊔ {Fijian} ⊔ {Polynesian;n.o.s.} ⊔

{Samoan} ⊔ {PacificIslandsorigins;n.i.e.}

HouseholdCharactersitics is a subset of Demographics and has exactly one characteristics and each distinct characteristic has exactly one population value.

HouseholdCharacteristics ⊑ Demographics

⊓ =1 hasCharacteristics.HouseholdType

⊓ =1 population.xsd:positiveInteger

HouseholdType ≡ {2\_persons} ⊔ {3\_persons} ⊔ {4\_persons} ⊔ {5\_ or\_more\_persons}

MainModeOfCommute is a subset of Demographics and has exactly one characteristics and each distinct characteristic has exactly one population value.

MainModeOfCommute ⊑ Demographics

⊓ =1 hasCharacteristics.CommuteType

⊓ =1 population.xsd:positiveInteger

CommuteType ≡ {Bicycle} ⊔ {Public\_transit} ⊔ {'Car,\_truck,\_van\_-\_as\_a\_driver'} ⊔ {Other\_Method} ⊔ {'Car,\_truck,\_van\_-\_as\_a\_passenger'} ⊔ {Walked}

AgeCharacteristics is a subset of Demographics and has exactly one characteristics and each distinct characteristic has exactly one population value.

AgeCharacteristics ⊑ Demographics

⊓ =1 hasCharacteristics.AgeGroupType

⊓ =1 population.xsd:positiveInteger

AgeGroupType ≡ {Male:\_75\_to\_79\_years} ⊔ {Male:\_50\_to\_54\_years} ⊔ {Male:\_95\_to\_99\_years} ⊔ {Male:\_70\_to\_74\_years} ⊔ {Male:\_35\_to\_39\_years} ⊔ {Female:\_15\_to\_19\_years} ⊔ {Male:\_90\_to\_94\_years} ⊔ {Female:\_80\_to\_84\_years} ⊔ {Male:\_55\_to\_59\_years} ⊔ {'Seniors\_(65+\_years)'} ⊔ {Female:\_60\_to\_64\_years} ⊔ {Female:\_100\_years\_and\_over} ⊔ {Female:\_0\_to\_04\_years} ⊔ {Female:\_30\_to\_34\_years} ⊔ {'Children\_(0-14\_years)'} ⊔ {Female:\_55\_to\_59\_years} ⊔ {Male:\_15\_to\_19\_years} ⊔ {'Youth\_(15-24\_years)'} ⊔ {Female:\_10\_to\_14\_years} ⊔ {Female:\_35\_to\_39\_years} ⊔ {Female:\_95\_to\_99\_years} ⊔ {'Pre-retirement\_(55-64\_years)'} ⊔ {Male:\_20\_to\_24\_years} ⊔ {Female:\_75\_to\_79\_years} ⊔ {Male:\_40\_to\_44\_years} ⊔ {'Working\_Age\_(25-54\_years)'} ⊔ {Male:\_65\_to\_69\_years} ⊔ {Male:\_60\_to\_64\_years} ⊔ {Male:\_100\_years\_and\_over} ⊔ {Female:\_90\_to\_94\_years} ⊔ {'Older\_Seniors\_(85+\_years)'} ⊔ {Female:\_05\_to\_09\_years} ⊔ {Male:\_85\_to\_89\_years} ⊔ {Male:\_80\_to\_84\_years} ⊔ {Female:\_70\_to\_74\_years} ⊔ {Female:\_50\_to\_54\_years} ⊔ {Male:\_25\_to\_29\_years} ⊔{Male:\_0\_to\_04\_years} ⊔ {Male:\_45\_to\_49\_years} ⊔ {Male:\_05\_to\_09\_years} ⊔ {Female:\_40\_to\_44\_years} ⊔ {Female:\_45\_to\_49\_years} ⊔ {Female:\_20\_to\_24\_years} ⊔ {Female:\_25\_to\_29\_years} ⊔ {Male:\_10\_to\_14\_years} ⊔ {Male:\_30\_to\_34\_years} ⊔ {Female:\_65\_to\_69\_years}

EmploymentStatus is a subset of Demographics and has exactly one characteristics and each distinct characteristic has exactly one population value.

EmploymentStatus ⊑ Demographics

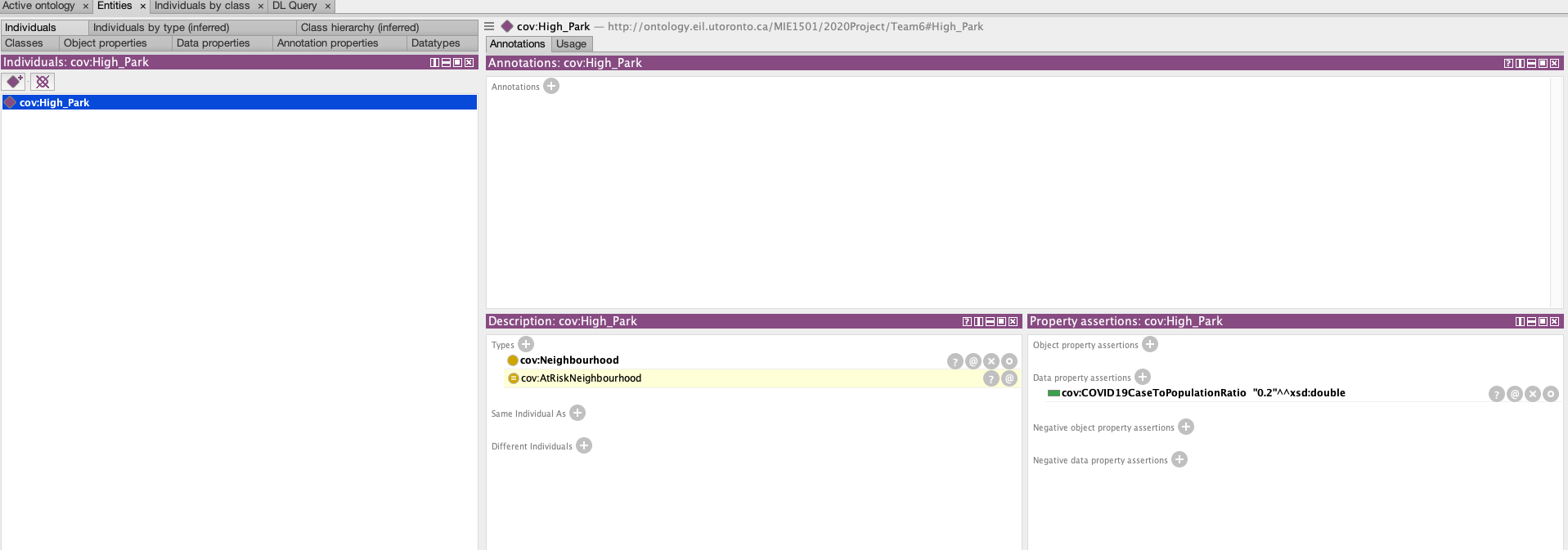
⊓ =1 hasCharacteristics.EmploymentType

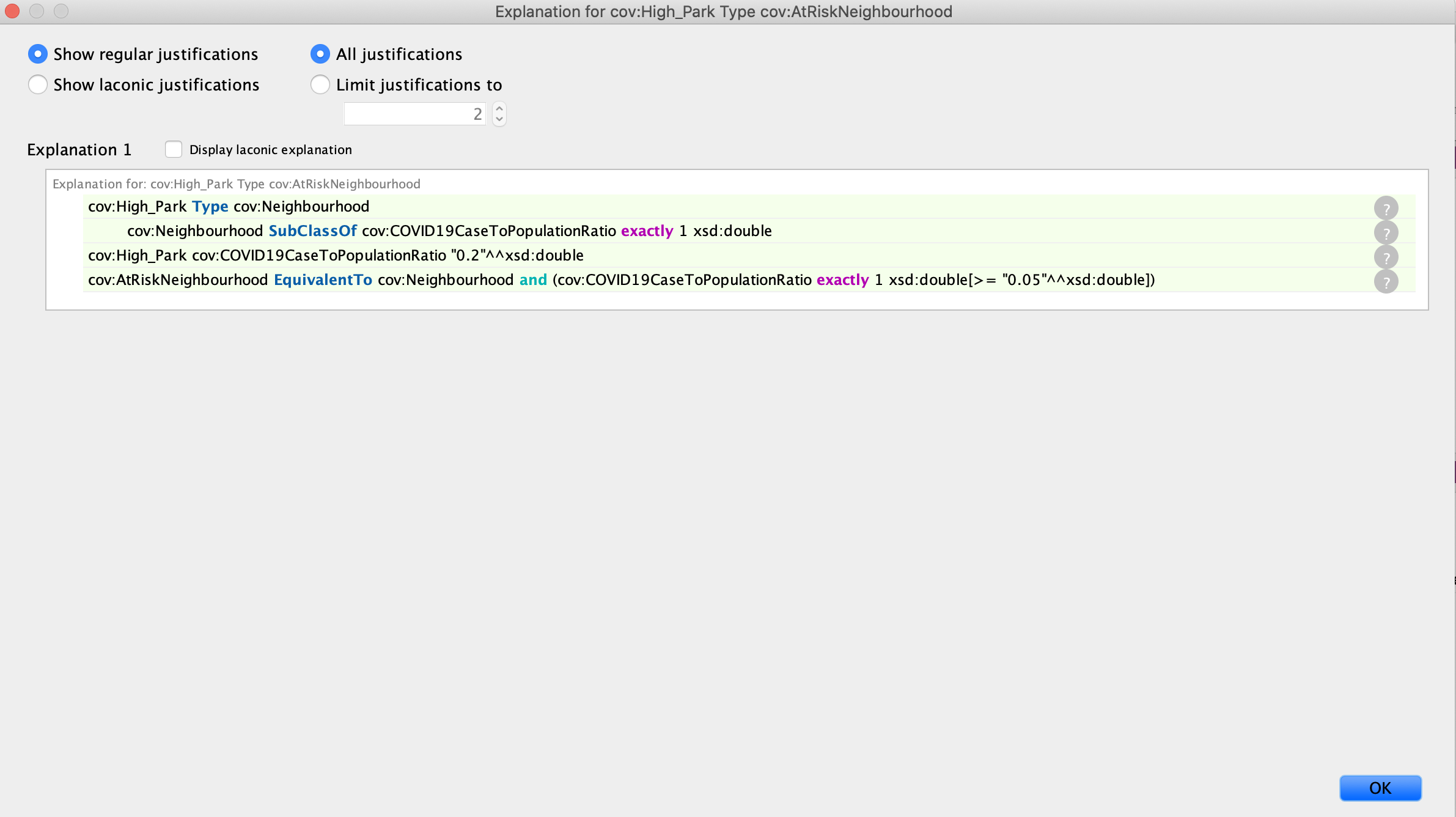
⊓ =1 population.xsd:positiveInteger

EmploymentType ≡ {Employed} ⊔ {Unemployed}

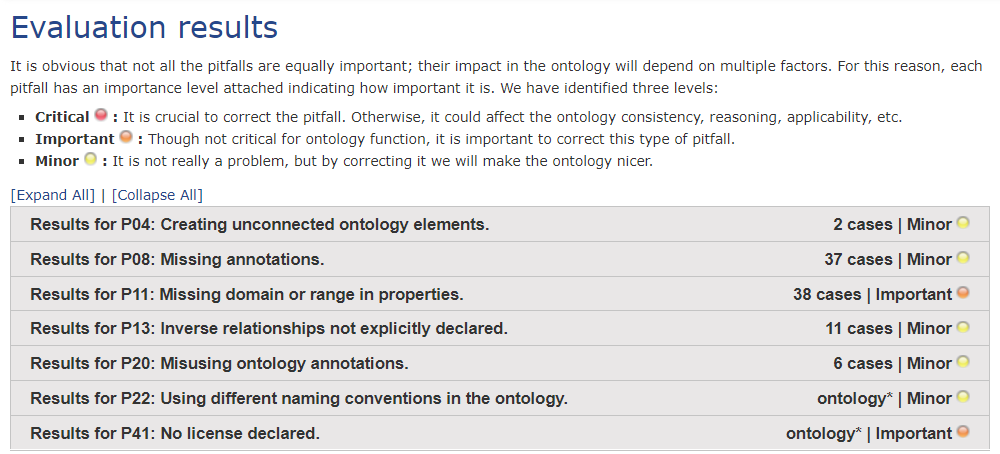
# 1.2 Protégé HermiT Reasoner Evaluation

A small sample data were loaded into Protégé to verify that the default HermiT reasoner was able to classify the neighbourhoods at risk.





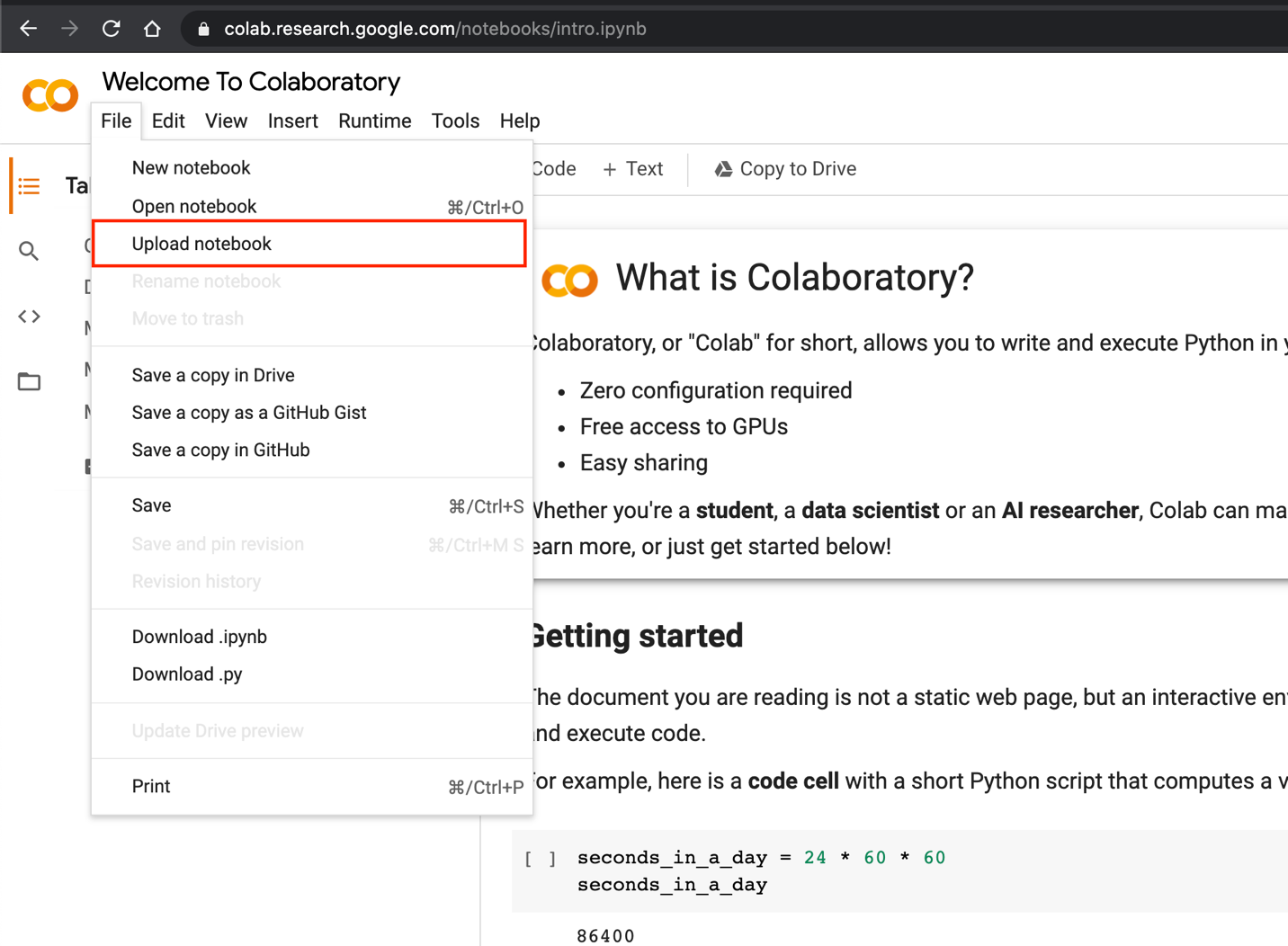
# Ontology Pitfalls Evaluation (OOPS)



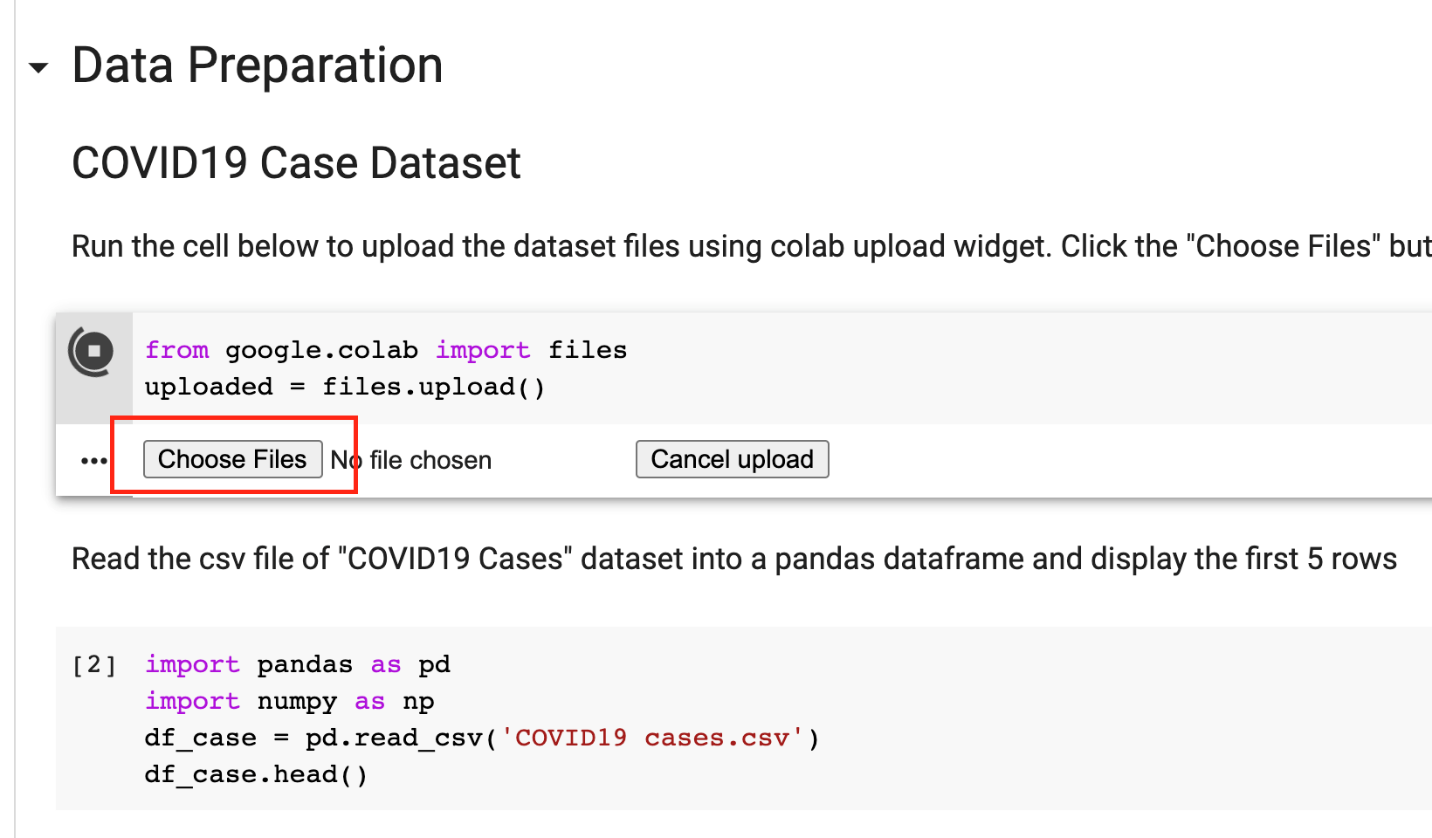
# Python Code for Turtle File

We produced the python code for processing and transforming the data in a Jupyter Notebook for a more interactive experience. Step by step explanation was added to the notebook, along with output display of the post-processing Pandas DataFrames.

It is highly recommended to view our .ipynb file in Google Colabotory <https://colab.research.google.com/notebooks/intro.ipynb>, as we use some of their file widgets in our code. To get started, please go to the link above and click File -> Upload notebook to upload the .ipynb file provided by us. You will be able to drag and drop the file into the upload pop-up window.



In order to completely run the code, it will be necessary to upload the two datasets from your local. Please follow the instructions provided in the notebook - run the cell and choose the csv files to upload.



After successfully running all the cells, you will be able to download the turtle file.



# SPARQL in Apache Janna Fuseki

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

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

PREFIX ns1: <http://ontology.eil.utoronto.ca/MIE1501/2020Project/Team6#>

PREFIX time: <http://www.w3.org/2006/time#>

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

PREFIX co: <http://purl.org/ontology/co/core#>

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

**What is the neighbourhood with the highest number of Covid-19 events during time period [t1, t2]?**

SELECT ?nbh ?count

WHERE {

SELECT ?nbh (Count(?id) AS ?count)

WHERE {

?case rdf:type ns1:COVID19Case ;

ns1:assignedID ?id ;

ns1:inNeighbourhood ?nbh ;

ns1:episodeDate ?date .

?date time:inXSDDate ?d .

FILTER (?d > "2020-01-01"^^xsd:date && ?d < "2020-12-31"^^xsd:date)

}

GROUP BY ?nbh

}

ORDER BY DESC(xsd:integer(?count)) LIMIT 1



**What is the neighbourhood with the highest per capita Covid-19 events during time period [t1, t2]?**

SELECT \*

WHERE {

?nbh rdf:type ns1:Neighbourhood ;

ns1:COVID19CaseToPopulationRatio ?r .

}

ORDER BY DESC(?r) LIMIT 1



**What neighbourhood had the largest number of “outbreak associated” cases during the time period [t1, t2]?**

SELECT ?nbh ?count

WHERE {

SELECT ?nbh (Count(?id) AS ?count)

WHERE {

?case rdf:type ns1:COVID19Case ;

ns1:assignedID ?id ;

ns1:outbreakAssociated "Outbreak Associated"^^xsd:string ;

ns1:inNeighbourhood ?nbh ;

ns1:episodeDate ?date .

?date time:inXSDDate ?d .

FILTER (?d > "2020-01-01"^^xsd:date && ?d < "2020-12-31"^^xsd:date)

}

GROUP BY ?nbh

}

ORDER BY DESC(xsd:integer(?count)) LIMIT 1



**What neighbourhood had the largest number of cases that were admitted to the intensive care unit (ICU) related to their COVID-19 infection during time period [t1, t2]?**

SELECT ?nbh ?count

WHERE {

SELECT ?nbh (Count(?id) AS ?count)

WHERE {

?case rdf:type ns1:COVID19Case ;

ns1:assignedID ?id ;

ns1:everInICU "true"^^xsd:boolean ;

ns1:inNeighbourhood ?nbh ;

ns1:episodeDate ?date .

?date time:inXSDDate ?d .

FILTER (?d > "2020-01-01"^^xsd:date && ?d < "2020-12-31"^^xsd:date)

}

GROUP BY ?nbh

}

ORDER BY DESC(xsd:integer(?count)) LIMIT 1

****

**What are the number of cases of all three outcome types (Fatal, Resolved, Active) for each neighbourhood during time period [t1, t2]?**

SELECT ?nbh ?outcome (Count(?id) AS ?count)

WHERE {

?case rdf:type ns1:COVID19Case ;

ns1:assignedID ?id ;

ns1:outcome ?outcome ;

ns1:inNeighbourhood ?nbh ;

ns1:episodeDate ?date .

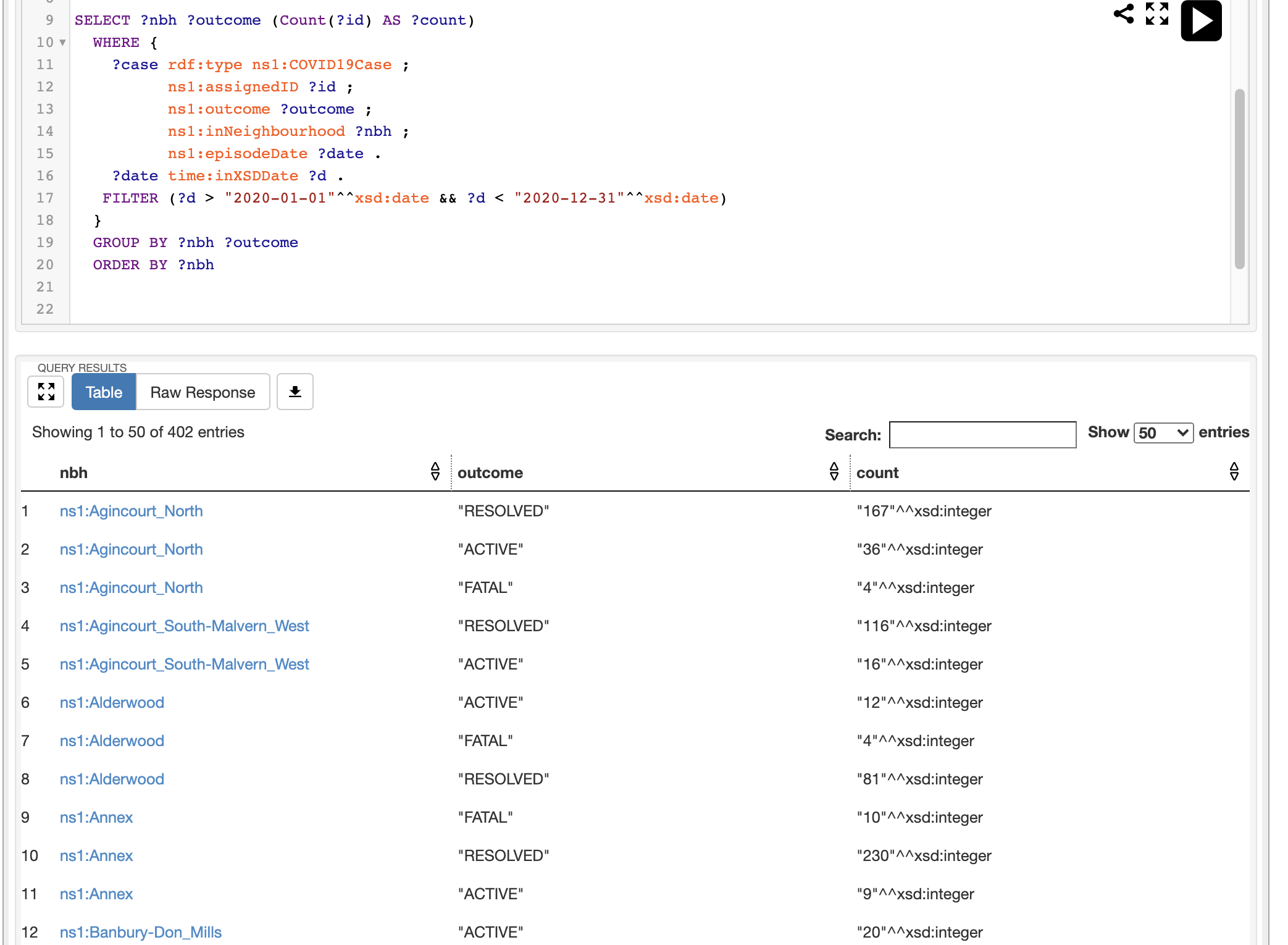
?date time:inXSDDate ?d .

FILTER (?d > "2020-01-01"^^xsd:date && ?d < "2020-12-31"^^xsd:date)

}

GROUP BY ?nbh ?outcome

ORDER BY ?nbh

****

**What are the demographics of the neighbourhood with the largest number of cases due to a given “Source of Infection” s, during the time period [t1, t2]?**

SELECT ?nbh ?characteristics ?population

WHERE {

?nbh ?pred ?object .

?object ns1:hasCharacteristics ?characteristics ;

ns1:population ?population .

{

SELECT ?nbh ?count

WHERE {

SELECT ?nbh (Count(?id) AS ?count)

WHERE {

?case rdf:type ns1:COVID19Case ;

ns1:assignedID ?id ;

ns1:sourceOfInfection "Close contact"^^rdfs:Literal ;

ns1:inNeighbourhood ?nbh ;

ns1:episodeDate ?date .

?date time:inXSDDate ?d .

FILTER (?d > "2020-01-01"^^xsd:date && ?d < "2020-12-31"^^xsd:date)

}

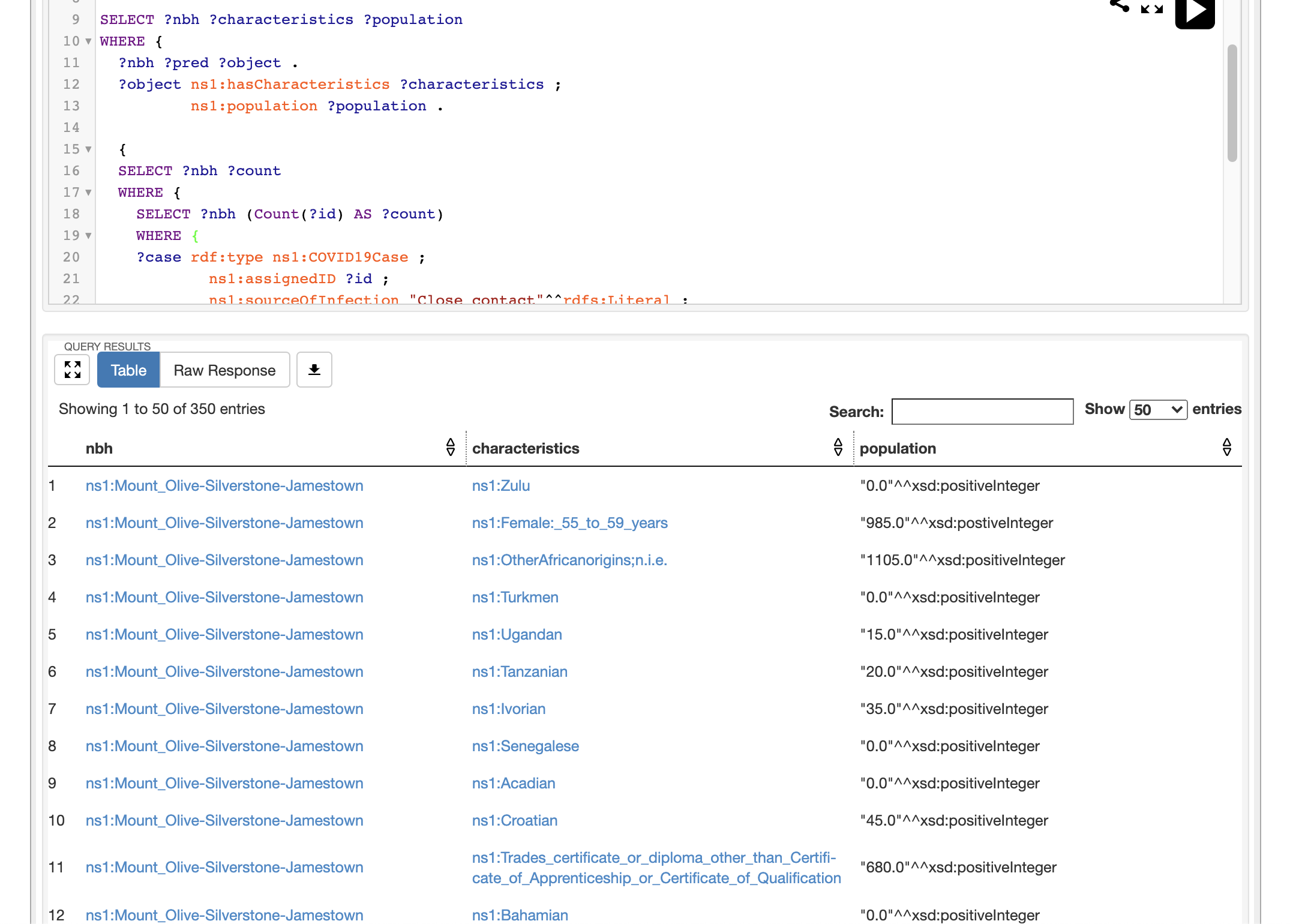
GROUP BY ?nbh

}

ORDER BY DESC(xsd:integer(?count)) LIMIT 1

}

}



**For any neighbourhood, how many cases due to a given “Source of Infection” s, for a given age range [a1, a1], during the time period [t1, t2]?**

SELECT ?nbh (Count(?id) AS ?count)

WHERE {

?case rdf:type ns1:COVID19Case ;

ns1:assignedID ?id ;

ns1:sourceOfInfection "Close contact"^^rdfs:Literal ;

ns1:ageGroup "20 to 29 Years"^^rdfs:Literal ;

ns1:inNeighbourhood ?nbh ;

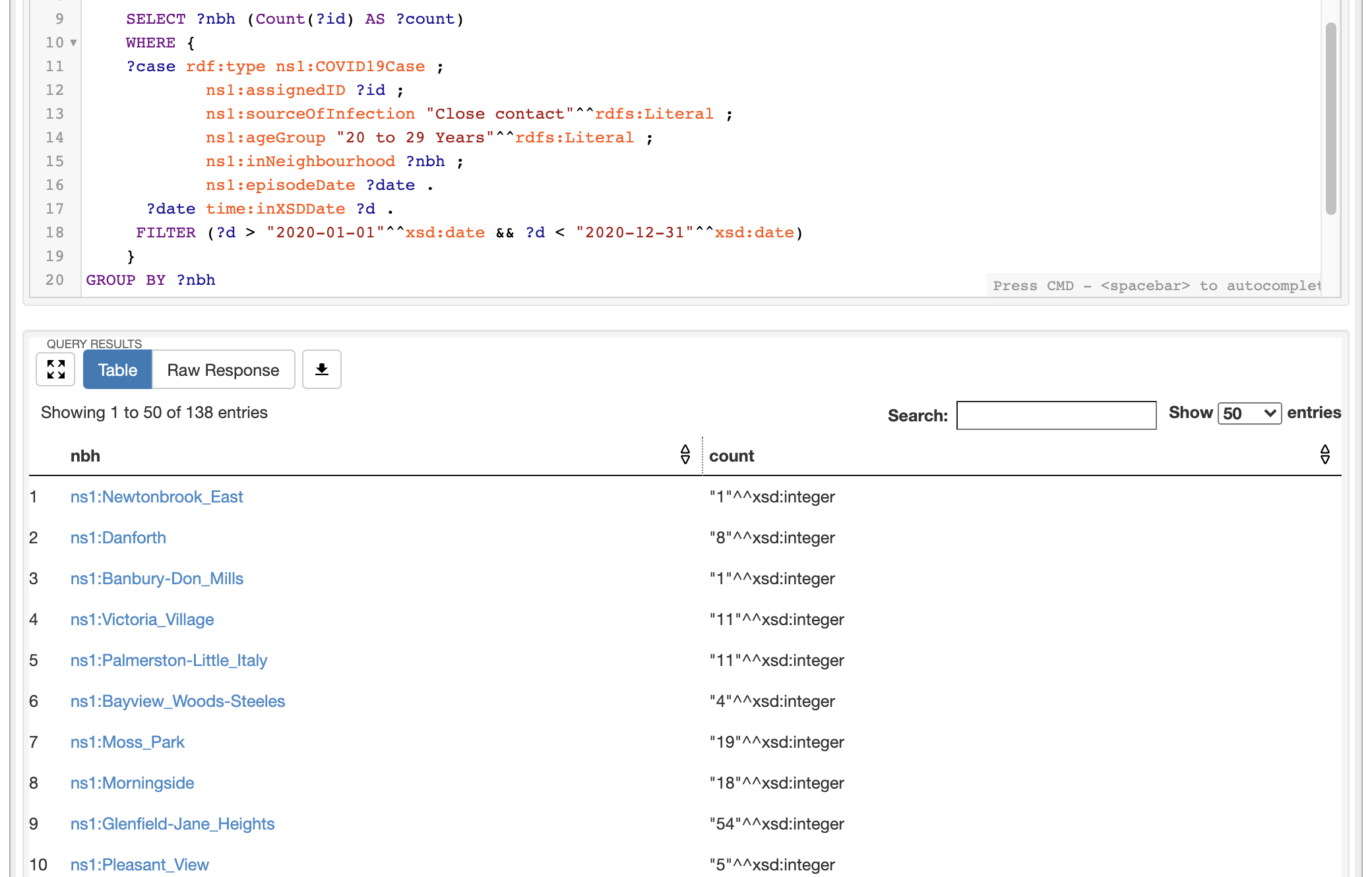
ns1:episodeDate ?date .

?date time:inXSDDate ?d .

FILTER (?d > "2020-01-01"^^xsd:date && ?d < "2020-12-31"^^xsd:date)

}

GROUP BY ?nbh

****

**What is the ratio of number of currently intubated COVID-19 cases to number of seniors for a given neighbourhood for a given time period?**

SELECT ?nbh ?count ?seniorPopulation ((xsd:float(?count)/xsd:float(?seniorPopulation)) AS ?ratio)

WHERE{

?nbh rdf:type ns1:Neighbourhood;

ns1:hasName "Yonge-Eglinton"^^xsd:string ;

ns1:hasAgeCharacteristics ?age .

?age ns1:hasCharacteristics <http://ontology.eil.utoronto.ca/MIE1501/2020Project/Team6#Seniors\_(65+\_years)> ;

ns1:population ?seniorPopulation .

{

SELECT ?nbh (Count(?id) AS ?count)

WHERE {

?case rdf:type ns1:COVID19Case ;

ns1:assignedID ?id ;

ns1:currentlyIntubated "true"^^xsd:boolean ;

ns1:inNeighbourhood ?nbh;

ns1:episodeDate ?date .

?date time:inXSDDate ?d .

FILTER (?d > "2020-01-01"^^xsd:date && ?d < "2020-12-31"^^xsd:date)

}

GROUP BY ?nbh

}

}



**What is the employment rate of the neighbourhood with the largest number of cases, during the time period [t1, t2]? What about the neighbourhood with the smallest number of cases?**

SELECT ?nbh ?employmentRate

WHERE {

?nbh ns1:hasEmploymentStatus ?emp .

?emp ns1:hasCharacteristics ns1:Employed ;

ns1:hasRate ?employmentRate .

{

SELECT ?nbh ?count

WHERE {

SELECT ?nbh (Count(?id) AS ?count)

WHERE {

?case rdf:type ns1:COVID19Case ;

ns1:assignedID ?id ;

ns1:inNeighbourhood ?nbh ;

ns1:episodeDate ?date .

?date time:inXSDDate ?d .

FILTER (?d > "2020-01-01"^^xsd:date && ?d < "2020-12-31"^^xsd:date)

}

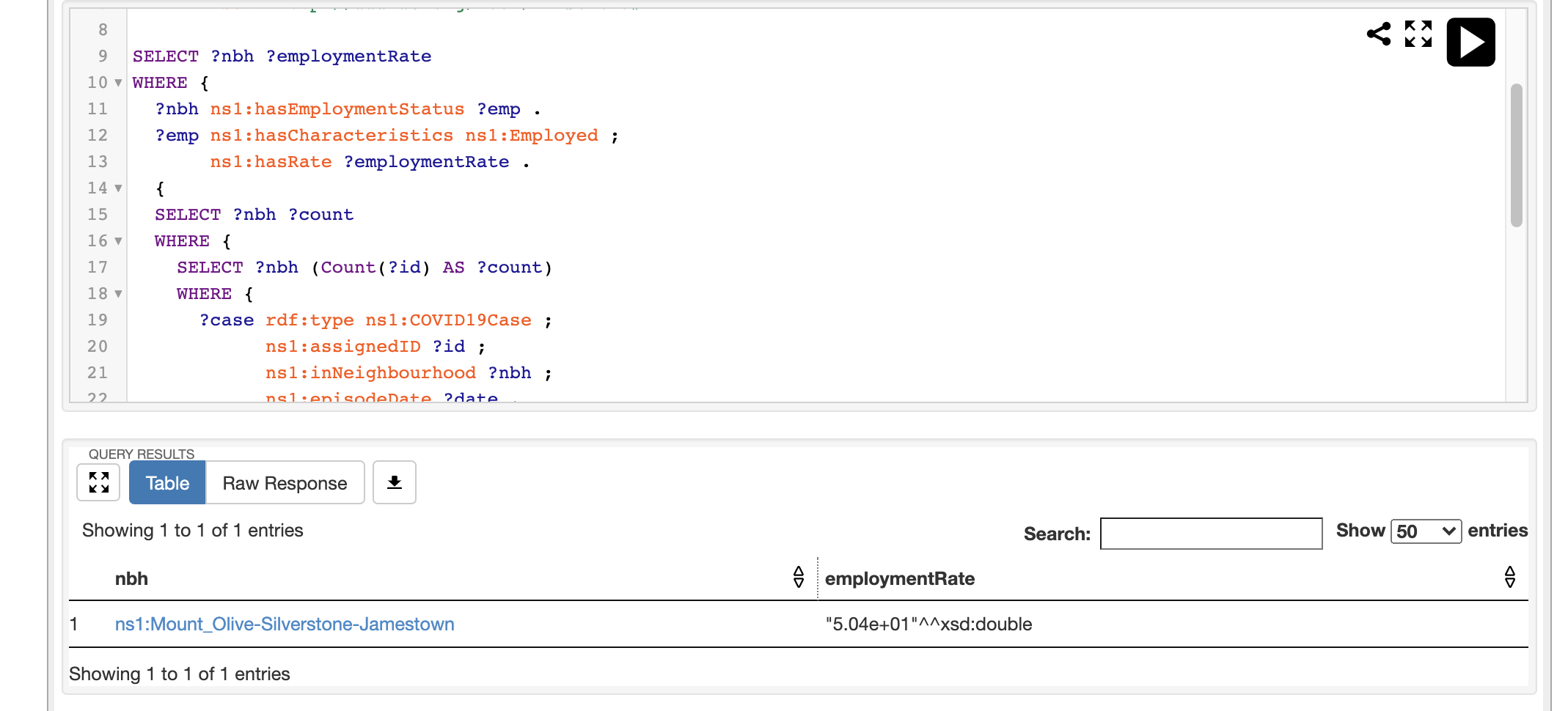
GROUP BY ?nbh

}

ORDER BY DESC(xsd:integer(?count)) LIMIT 1

}

}

**What is the ethnic origin distribution of the neighbourhood with the largest number of cases, during the time period [t1, t2]? What about the neighbourhood with the smallest number of cases?**

SELECT ?nbh ?ethnicOriginType ?population

WHERE {

?nbh ns1:hasEthnicOrigin ?eo .

?eo ns1:hasCharacteristics ?ethnicOriginType ;

ns1:population ?population .

{

SELECT ?nbh ?count

WHERE {

SELECT ?nbh (Count(?id) AS ?count)

WHERE {

?case rdf:type ns1:COVID19Case ;

ns1:assignedID ?id ;

ns1:inNeighbourhood ?nbh ;

ns1:episodeDate ?date .

?date time:inXSDDate ?d .

FILTER (?d > "2020-01-01"^^xsd:date && ?d < "2020-12-31"^^xsd:date)

}

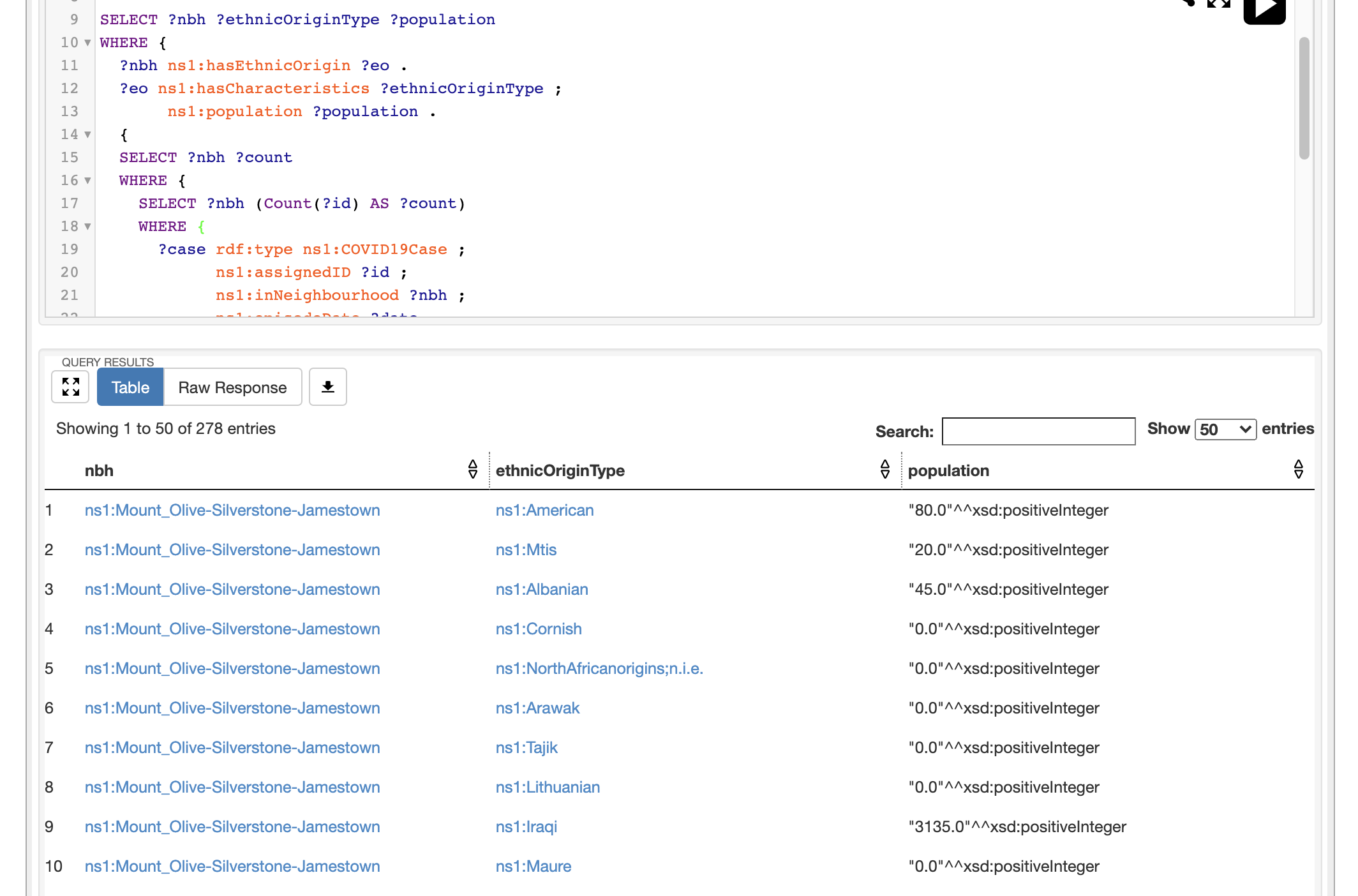
GROUP BY ?nbh

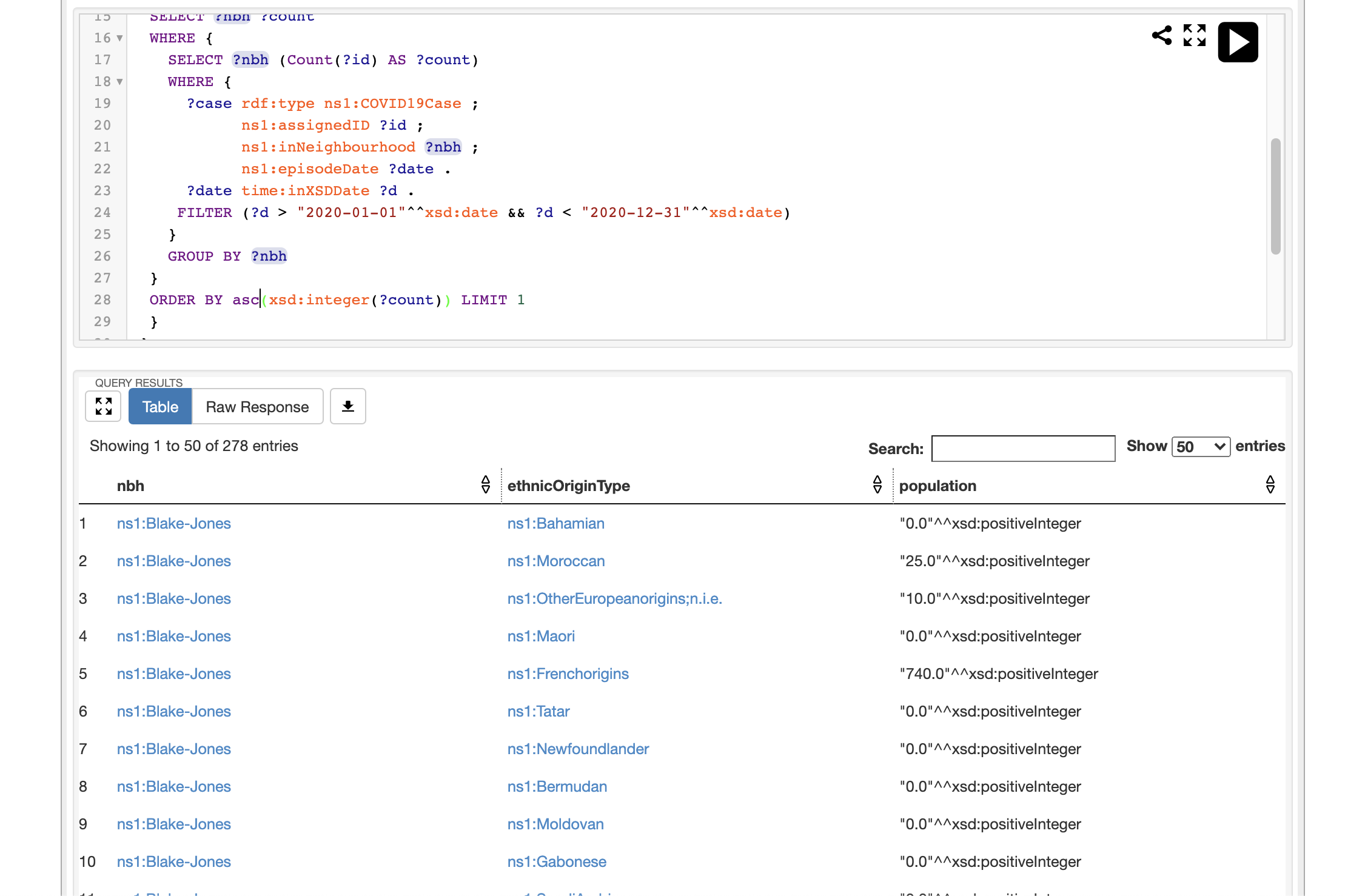
}

ORDER BY DESC(xsd:integer(?count)) LIMIT 1

}

}

****



**What is the household characteristics distribution (e.g. 2 persons, 3 persons etc) of the neighbourhood with the largest number of cases, during the time period [t1, t2]? What about the neighbourhood with the smallest number of cases?**

SELECT ?nbh ?characteristics ?population

WHERE {

?nbh ns1:hasHouseholdCharacteristics ?obj .

?obj ns1:hasCharacteristics ?characteristics;

ns1:population ?population .

{

SELECT ?nbh ?count

WHERE {

SELECT ?nbh (Count(?id) AS ?count)

WHERE {

?case rdf:type ns1:COVID19Case ;

ns1:assignedID ?id ;

ns1:inNeighbourhood ?nbh ;

ns1:episodeDate ?date .

?date time:inXSDDate ?d .

FILTER (?d > "2020-01-01"^^xsd:date && ?d < "2020-12-31"^^xsd:date)

}

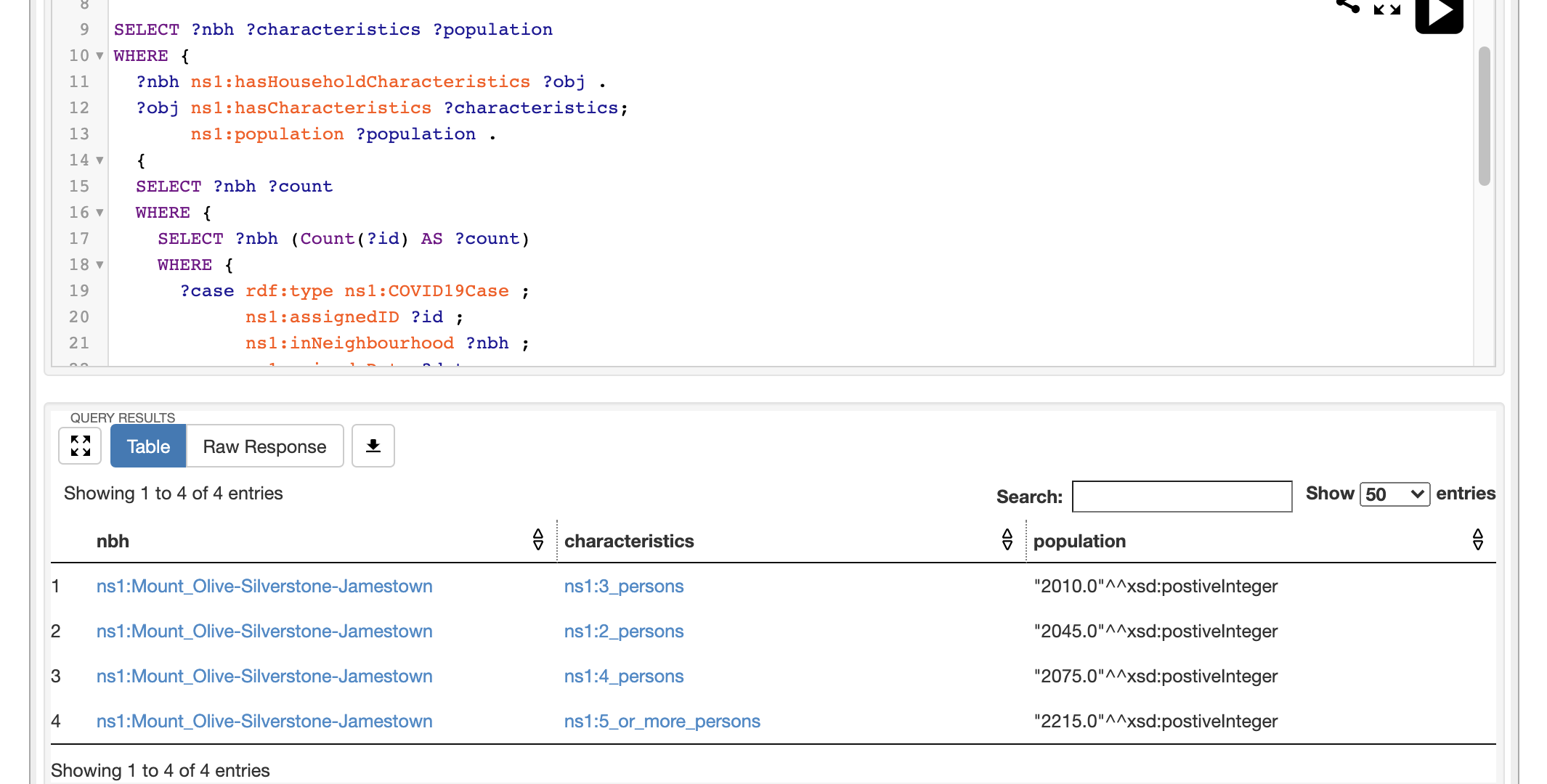
GROUP BY ?nbh

}

ORDER BY DESC(xsd:integer(?count)) LIMIT 1

}

}

**What is the average after tax household income of the neighbourhood with the largest number of cases, during the time period [t1, t2]? What about the neighbourhood with the smallest number of cases?**

SELECT ?nbh ?income

WHERE {

?nbh ns1:averageAfterTaxHouseholdIncome ?income .

{

SELECT ?nbh ?count

WHERE {

SELECT ?nbh (Count(?id) AS ?count)

WHERE {

?case rdf:type ns1:COVID19Case ;

ns1:assignedID ?id ;

ns1:inNeighbourhood ?nbh ;

ns1:episodeDate ?date .

?date time:inXSDDate ?d .

FILTER (?d > "2020-01-01"^^xsd:date && ?d < "2020-12-31"^^xsd:date)

}

GROUP BY ?nbh

}

ORDER BY DESC(xsd:integer(?count)) LIMIT 1

}

}

****

****

**What is the most common main mode of commute of the neighbourhood with the largest number of cases, during the time period [t1, t2]? What about the neighbourhood with the smallest number of cases?**

SELECT ?nbh ?characteristics ?population

WHERE {

?nbh ns1:hasMainModeofCommute ?obj .

?obj ns1:hasCharacteristics ?characteristics ;

ns1:population ?population .

{

SELECT ?nbh ?count

WHERE {

SELECT ?nbh (Count(?id) AS ?count)

WHERE {

?case rdf:type ns1:COVID19Case ;

ns1:assignedID ?id ;

ns1:inNeighbourhood ?nbh ;

ns1:episodeDate ?date .

?date time:inXSDDate ?d .

FILTER (?d > "2020-01-01"^^xsd:date && ?d < "2020-12-31"^^xsd:date)

}

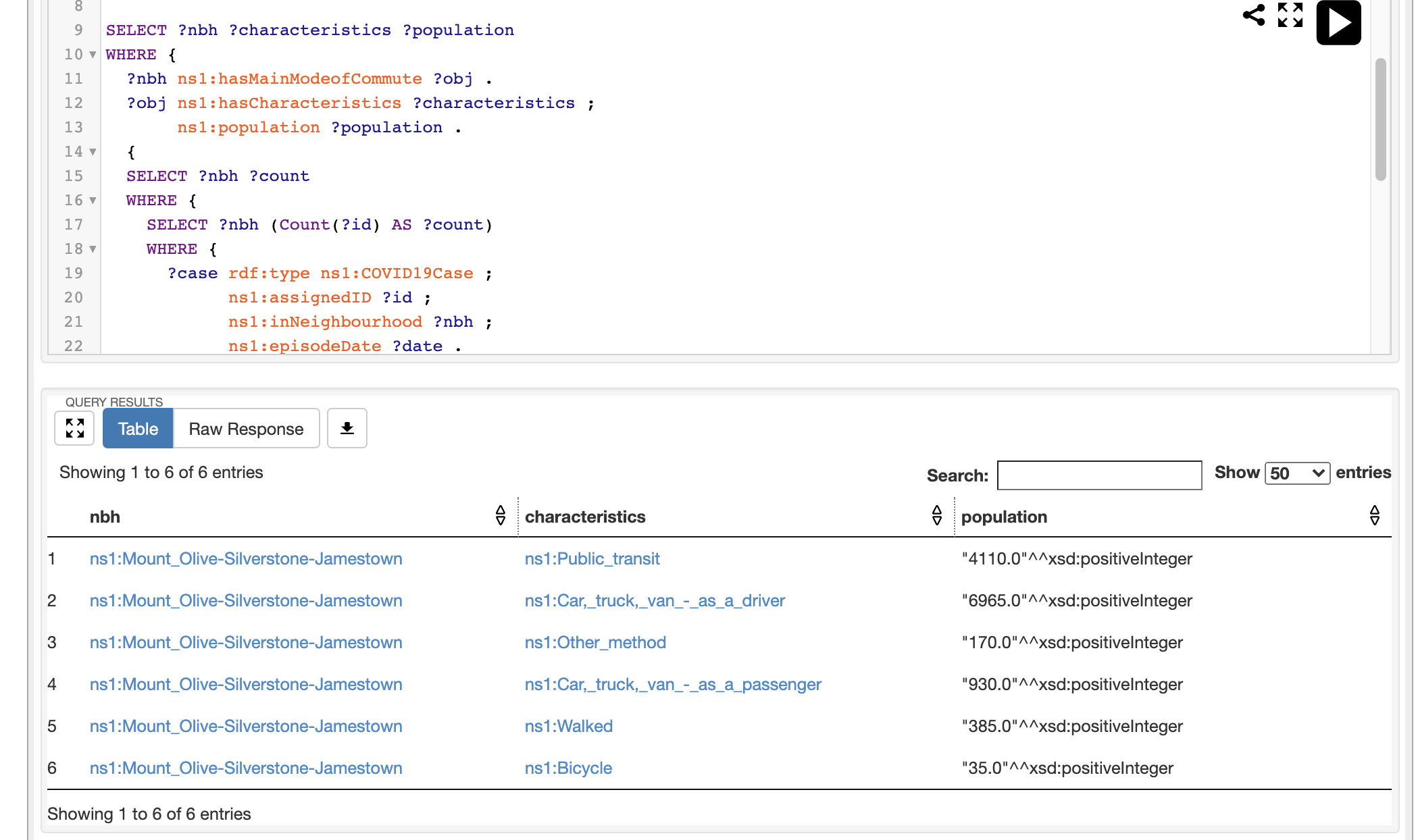
GROUP BY ?nbh

}

ORDER BY DESC(xsd:integer(?count)) LIMIT 1

}

}

****

****

**What is the percentage of the population with no certificate, diploma or degree of the neighbourhood with the largest number of cases, during the time period [t1, t2]? What about the neighbourhood with the smallest number of cases?**

SELECT ?nbh ?characteristics ?population ((xsd:float(?population)/xsd:float(?totalPopulation)) AS ?ratio)

WHERE {

?nbh ns1:hasEducation ?obj ;

ns1:hasTotalPopulation ?totalPopulation .

?obj ns1:hasCharacteristics ?characteristics;

ns1:population ?population .

{

SELECT ?nbh ?count

WHERE {

SELECT ?nbh (Count(?id) AS ?count)

WHERE {

?case rdf:type ns1:COVID19Case ;

ns1:assignedID ?id ;

ns1:inNeighbourhood ?nbh ;

ns1:episodeDate ?date .

?date time:inXSDDate ?d .

FILTER (?d > "2020-01-01"^^xsd:date && ?d < "2020-12-31"^^xsd:date)

}

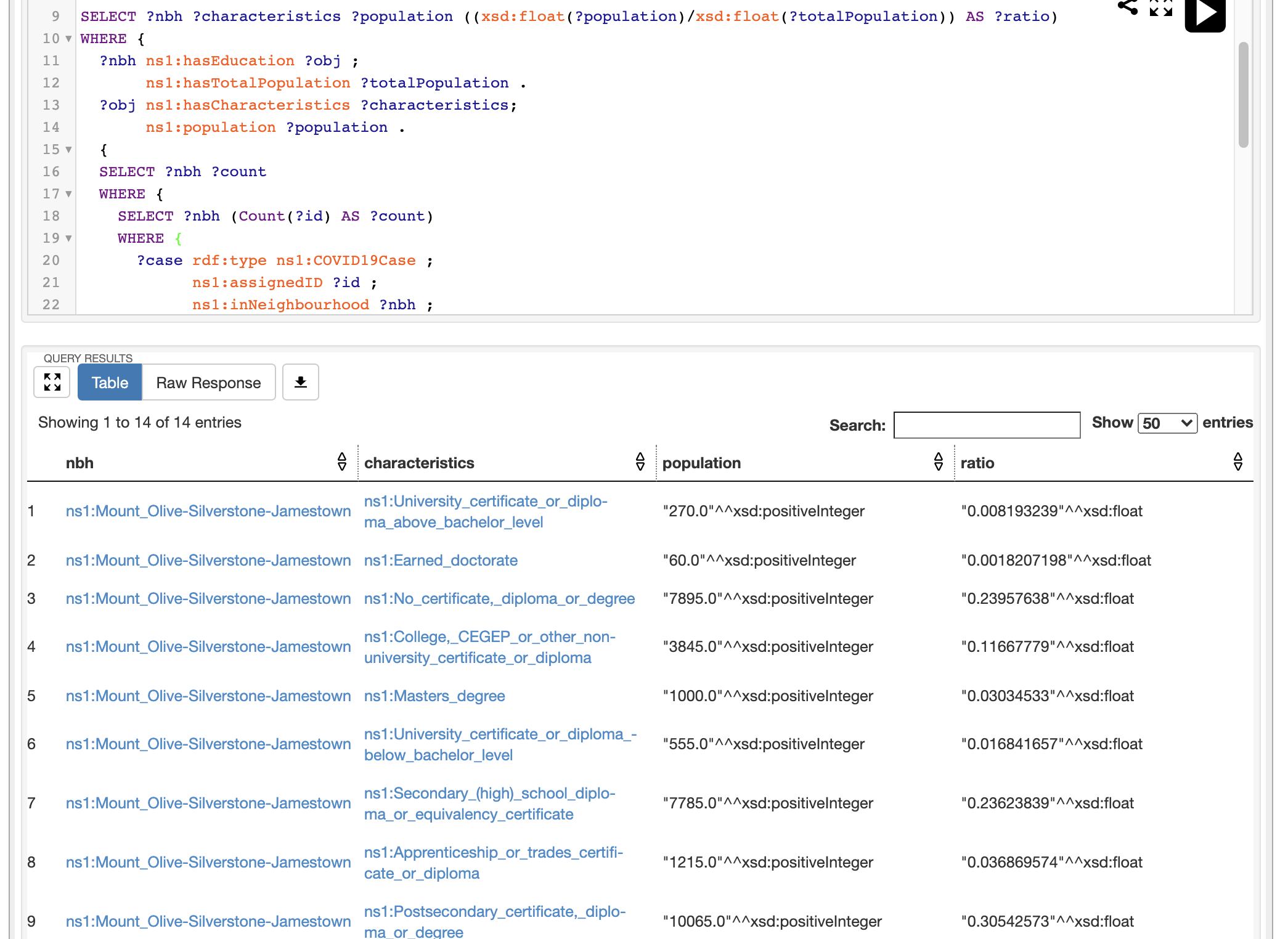
GROUP BY ?nbh

}

ORDER BY DESC(xsd:integer(?count)) LIMIT 1

}

}

****

