CSE 7320 Final Report

Student Name: Chenming Cui

Student ID: 47781790

# Part A

* **Codes**

.n3 file code:

@prefix CC: <http://www.smu.edu/~47781790/#>.

@prefix schema: <http://schema.org>.

CC:Jimmy a schema:Person.

CC:Jimmy schema:givenName "Jimmy".

CC:Jimmy schema:affiliation CC:Google.

CC:Jimmy schema:knows CC:Marla.

CC:Marla a schema:Person.

CC:Marla schema:givenName "Marla".

CC:Google a schema:Organization.

CC:Google schema:name "Google".

CC:Google schema:location CC:California.

CC:California a schema:Place.

SPARQL1.txt

prefix CC: <http://www.smu.edu/~47781790/#>

prefix schema: <http://schema.org>

SELECT ?x

WHERE{

?x a schema:Person.

}

SPARQL2.txt

prefix CC: <http://www.smu.edu/~47781790/#>

prefix schema: <http://schema.org>

SELECT ?name ?location

WHERE{

?name a schema:Organization.

?name schema:location ?location.

}

SPARQL3.txt

prefix CC: <http://www.smu.edu/~47781790/#>

prefix schema: <http://schema.org>

SELECT ?x

WHERE{

CC:Jimmy schema:affiliation ?x

}

SPARQL4.txt

prefix CC: <http://www.smu.edu/~47781790/#>

prefix schema: <http://schema.org>

SELECT ?knowsMarla

WHERE{

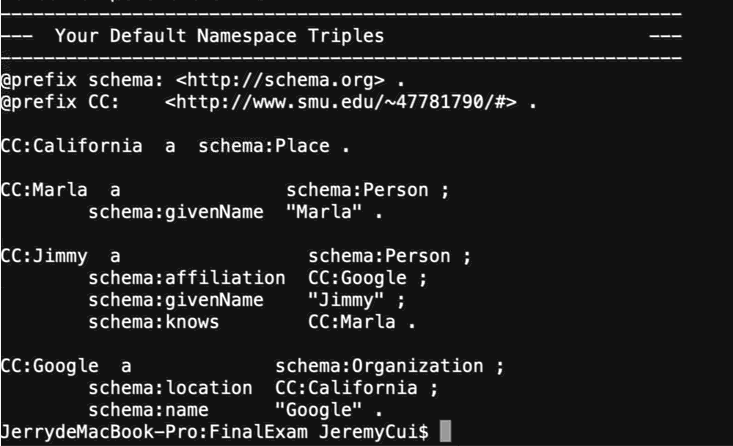
CC:Jimmy schema:knows ?x.

BIND (exists{?x schema:givenName "Marla"} AS ?existsMarla)

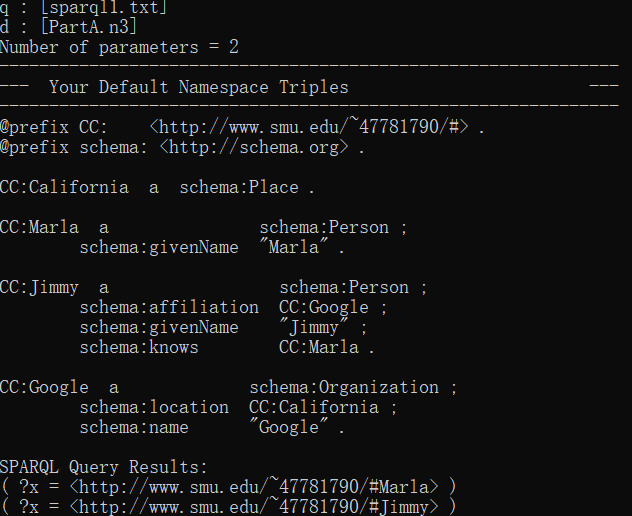
BIND (IF(?existsMarla, "yes", "no") AS ?knowsMarla)

}

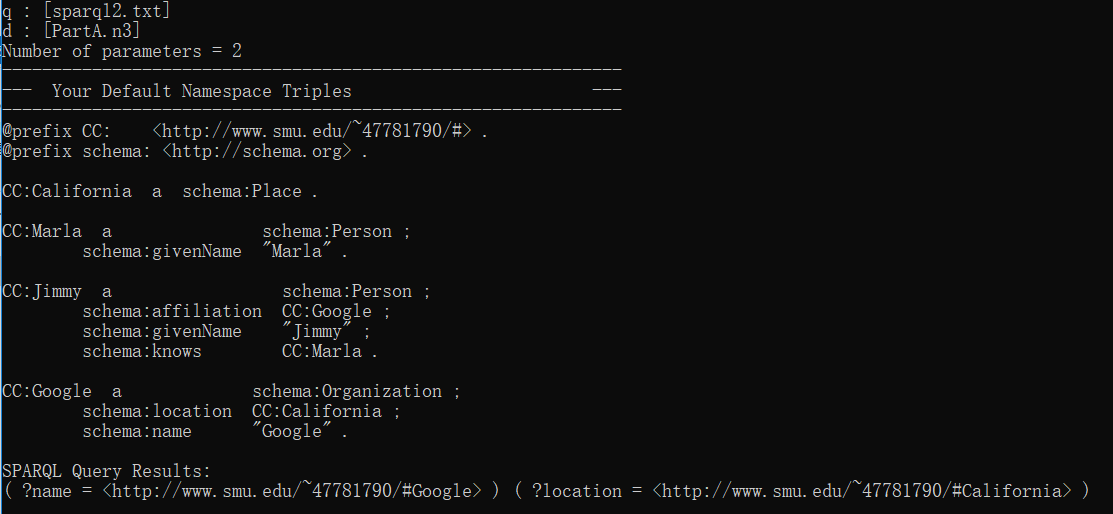
* **Results**

****

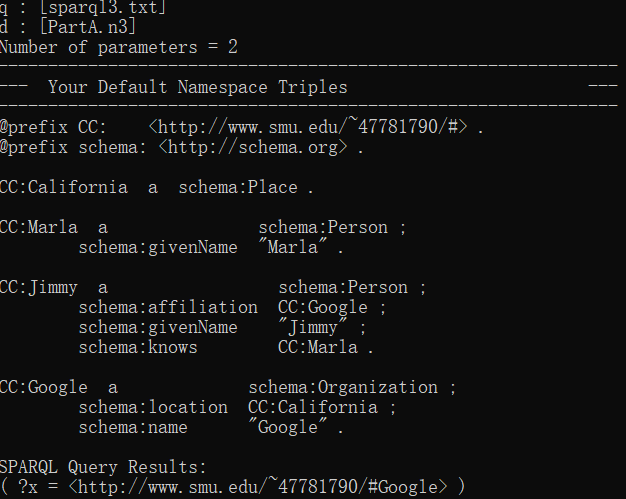
.n3 file capture



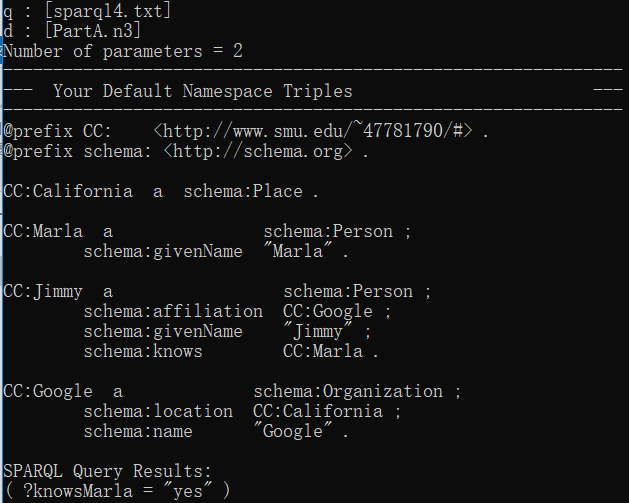
SPARQL1.txt result



SPARQL2.txt result



SPARQL3.txt result



SPARQL4.txt result

# Part B

* **Codes**

import nltk

nltk.download('punkt')

nltk.download('averaged\_perceptron\_tagger')

nltk.download('maxent\_ne\_chunker')

nltk.download('words')

#Create the first sentence

sentence1 = "Jimmy works at Google in California. He was seen whispering to Marla."

tokens1 = nltk.word\_tokenize(sentence1)

tags1 = nltk.pos\_tag(tokens1)

tree1 = nltk.ne\_chunk(tags1)

print(tree1)

#Create a word list including “works”.

#And define a grammar to choose VP.

work\_word\_list = ["works", "hired", "contract", "paid", "employed"]

output = "Jimmy"

grammar = "VP: {<VB.\*><TO|IN>?<NN.\*>}"

cp = nltk.RegexpParser(grammar)

result = cp.parse(tags1)

print(result)

for tuples in result:

if len(tuples) > 2: #select VP tuples

if tuples[0][0] in work\_word\_list: #select target VP

for couples in tuples: #add the words into the output

output += " "

output += couples[0]

print(output)

#Create another sentence to finish the third step

sentence2 = "Jimmy was seen whispering to Marla."

tokens2 = nltk.word\_tokenize(sentence2)

tags2 = nltk.pos\_tag(tokens2)

tree2 = nltk.ne\_chunk(tags2)

print(tree2)

#Create a word list contains “whispering”

#And create a grammar to find VP

know\_word\_list = ["knows", "whispering", "contact", "calls", "talks"]

output2 = "Jimmy"

knows = False

grammar2 = "VP: {<VB.\*>+<TO|IN>?<NN.\*>}"

cp2 = nltk.RegexpParser(grammar2)

result2 = cp2.parse(tags2)

print(result2)

for tuples in result2:

if len(tuples) > 2: #select VP tuples

for i in range(len(tuples)):

if tuples[i][0] in know\_word\_list: #select target VP

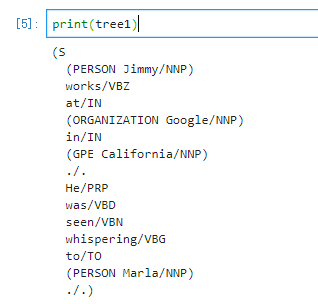
output2 += " knows " #imply that A knows B

if i == (len(tuples)-1):

output2 += tuples[i][0] #find out who is B

print(output2)

* **Results**



Result of tree1

In this result, it shows that Jimmy and Marla belong to PERSON Google belong to ORGANIZATION and California is a GPE, which means that California is a Place.

Thus, I can generate the following N3:

CC:Jimmy a schema:Person.

CC:Jimmy schema:givenName "Jimmy".

CC:Google a schema:Organization.

CC:Google schema:name "Google".

CC:California a schema:Place.

CC:Google schema:location CC:California.

CC:Marla a schema:Person.

CC:Marla schema:givenName "Marla".



Step 2 Result

From this result, I generated the following N3:

CC:Jimmy schema:affiliation CC:Google.



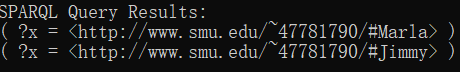
Step 3 Result

From this result, I generated the following N3:

CC:Jimmy schema:knows CC:Marla.

# Part C

* **Results**



SPARQL1.txt result



SPARQL2.txt result



SPARQL3.txt result



SPARQL4.txt result