DBpedia to Wikidata:
Exploring the Linked Jazz Name Directory

Mollie Echeverria
LIS-664 - Programming for Cultural Heritage
Fall 2016
Prof Matt Miller

DBpedia

- Founded by researchers from two German universities in 2007.
- Aims to extract content from Wikidata and publish as structured content.
- Users can semantically query this data, revealing relationships and properties connected to Wikipedia resources.

Wikidata

- Founded in 2012 by the Wikimedia Foundation.
- Aggregates content from all the Wikimedia sites as key-value pairs.
- May be supplanting DBpedia as a source for Wikipedia-based linked data.

Linked Jazz

- Research project at Pratt focused on exploring linked open data in the context of cultural institutions
- Focused specifically on exploring relationships between jazz musicians based on data in oral history transcripts, as well as other sources like archival documents.

The Jazz Names Directory

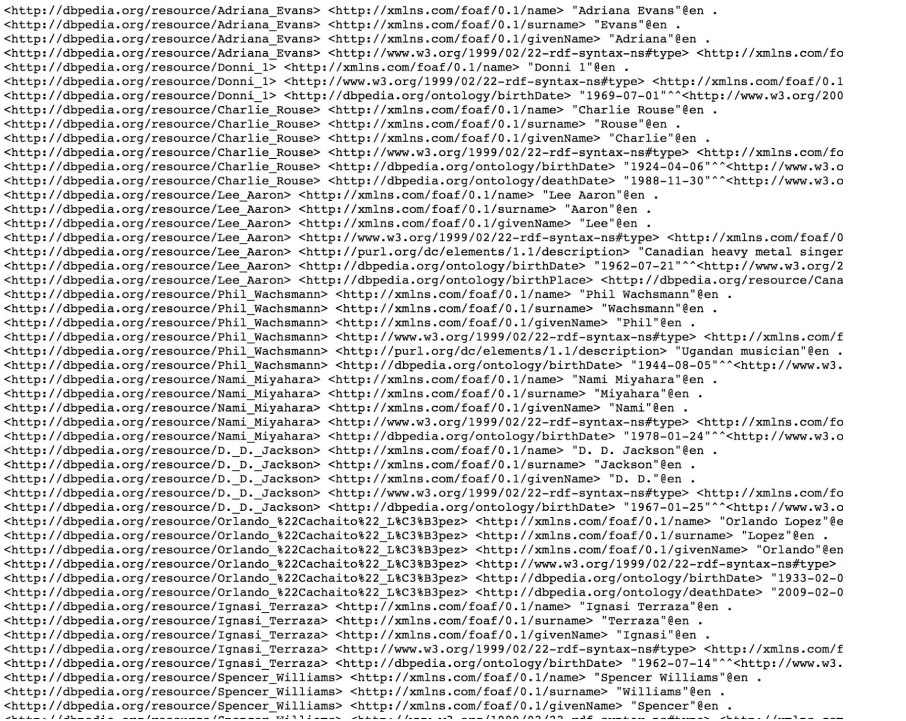
- At the start of the Linked Jazz Project in 2011, DBpedia was queried for names of jazz musicians. This yielded around 9,000 names.
- This list was later narrowed down to individuals mentioned in jazz oral history transcripts.
- 8,725 of the original names are still hosted on the Linked Jazz website as N-Triples (a textual format used to store linked data).

The New Orleans Jazz & Heritage Foundation

- Linked Jazz recently received a grant from the New Orleans Jazz and Heritage Foundation to create a linked data dataset of Louisiana-based jazz musicians.
- In support of this project, Linked Jazz wanted to investigate how many of the musicians already in its 8,725 name directory were New Orleans-based.
- The team also wanted to explore whether Wikidata could offer richer data than DBpedia (such as familial relationships).

Extracting Data From the Name Directory

- I started by downloading the Jazz Names Directory as an N-Triple file.
- To make this data, I converted the N-Triple into a JSON dictionary using a Python script.



Getting JSON from DBpedia

- Each name in the Jazz Names Directory is connected to a DBpedia resource page.
- Resources in DBpedia contain links to related resources, including the corresponding page for the same resource on Wikipedia.
- To query DBpedia, I had to access these pages in the form of JSON data.
- To do this, I used a script to replace the word “/resource/” in each URI in the directory to “/data/”. This allowed me to access the JSON equivalent of each page

Querying DBpedia for Wikidata URIs

- DBpedia resources store the URI for the resource's corresponding Wikidata page in a property called "sameAs".
- To get Wikidata URIs for the names in the Jazz Directory, I used a script to loop through all of properties in each person's DBpedia page, writing their DBpedia and Wikidata URI to a new document.

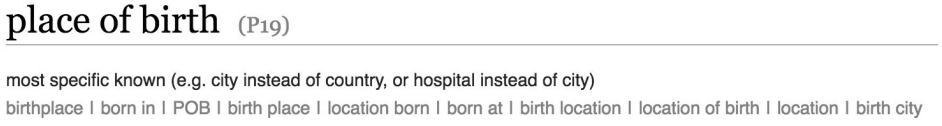
DBpedia Querying Issues

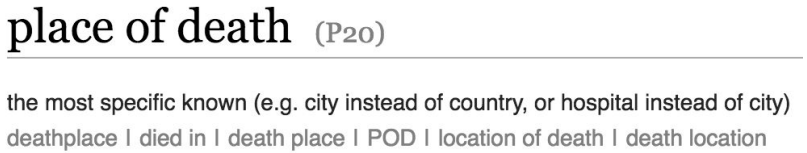
- Not all DBpedia pages had a corresponding Wikidata page, causing me to get a KeyError when I tried to run the script.
- I eventually ended up adding a Try/Except statement, allowing the script to pass over DBpedia resources missing Wikidata URIs.
- Eventually, I ended up with another JSON directory containing corresponding DBpedia and Wikidata URIs for those names in the Jazz Names Directory that had resources on both sites

```
try:
    for prop in dbpedia_data[dictionary["URI"]]:

            if prop =="http://www.w3.org/2002/07/owl#sameAs":

                for obj in dbpedia_data[dictionary["URI"]][prop]:

                    if "wikidata.org/entity/" in obj['value']:

                            print (obj['value'])

                            a_uri = {}

                            a_uri["db_uri"] = dictionary["URI"]

                            a_uri["wiki_uri"] = obj['value']

                            all_uris.append(a_uri)

except KeyError:

    pass
```

Getting Places of Birth and Death from Wikidata

- Now that I had Wikidata URIs for names in the Jazz Names Directory, my next step was figuring out which of these musicians were from New Orleans.
- To find New Orleans-based musicians, I had to query two of Wikidata's resource properties: Place of Birth and Place of Death.
- Using Wikidata's API, I attempted to extract these two properties for each resource in the name directory.





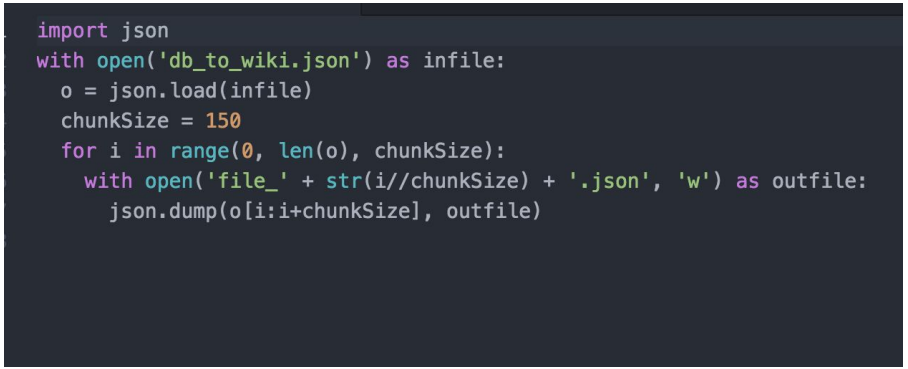
Hitting a Wall: Wikipedia's Server

- When I attempted to query Wikidata's API, I soon encountered a major obstacle in the form of ConnectionResetError 54.
- A few hundred names into my query, I would get disconnected from Wikidata's server on Wikidata's end. This was possibly due to the volume of data I was querying.
- I modified the "requests" method in my script from "requests.get" to "requests.post", and set my requests to not time out, but continued to be kicked off of Wikidata's server after a certain point.

```
Library/Frameworks/Python.framework/Versions/3.5/lib/python3.5/site-packages/requests/packages/urllib3/connectionpool.py
equest
_validate_conn(conn)
Library/Frameworks/Python.framework/Versions/3.5/lib/python3.5/site-packages/requests/packages/urllib3/connectionpool.py
te_conn
connect()
Library/Frameworks/Python.framework/Versions/3.5/lib/python3.5/site-packages/requests/packages/urllib3/connection.py", l

ersion=resolved_ssl_version)
Library/Frameworks/Python.framework/Versions/3.5/lib/python3.5/site-packages/requests/packages/urllib3/util/ssl_.py", li
ket
n context.wrap_socket(sock, server_hostname=server_hostname)
Library/Frameworks/Python.framework/Versions/3.5/lib/python3.5/ssl.py", line 377, in wrap_socket
ext=self)
Library/Frameworks/Python.framework/Versions/3.5/lib/python3.5/ssl.py", line 752, in __init__
do_handshake()
Library/Frameworks/Python.framework/Versions/3.5/lib/python3.5/ssl.py", line 988, in do_handshake
_sslobj.do_handshake()
Library/Frameworks/Python.framework/Versions/3.5/lib/python3.5/ssl.py", line 633, in do_handshake
_sslobj.do_handshake()
packages.urllib3.exceptions.ProtocolError: ('Connection aborted.', ConnectionResetError(54, 'Connection reset by peer'))

ndling of the above exception, another exception occurred:

 (most recent call last):
et_bd_data.py", line 45, in <module>
request = requests.post(url, timeout=None)
Library/Frameworks/Python.framework/Versions/3.5/lib/python3.5/site-packages/requests/api.py", line 110, in post
n request('post', url, data=data, json=json, **kwargs)
Library/Frameworks/Python.framework/Versions/3.5/lib/python3.5/site-packages/requests/api.py", line 56, in request
n session.request(method=method, url=url, **kwargs)
Library/Frameworks/Python.framework/Versions/3.5/lib/python3.5/site-packages/requests/sessions.py", line 475, in request
= self.send(prep, **send_kwargs)
```

Querying in Chunks

- After dozens of unsuccessful querying attempts, I decided to split my 8,000+ name directory into smaller JSON files.
- Using a script, I split the large directory into 54 smaller JSON files, each with around 150 names. 150 names seemed to be a cutoff where I could reliably query Wikidata without being kicked off their server.
- I then manually updated and ran my birthplace/deathplace script 54 times, once for each JSON file.



```
import json
with open('db_to_wiki.json') as infile:
  o = json.load(infile)
  chunkSize = 150
  for i in range(0, len(o), chunkSize):
    with open('file_' + str(i//chunkSize) + '.json', 'w') as outfile:
      json.dump(o[i:i+chunkSize], outfile)
```

Results

- 8,201 of the 8,725 names in the DBpedia-based Jazz Name Directory have corresponding Wikidata pages.
- 168 people were born in New Orleans.
- 71 people died in New Orleans.

Wiki Place Of Birth Literal	
Natchitoches	2
Natick	1
Navasota	2
Neasden	1
Nebraska	2
Nelspruit	1
Nembro	1
New Albany	1
New Brunswick	4
New Castle	1
New Hartford	1
New Haven	16
New Jersey	7
New Kensington	2
New London	1
New Orleans	168
New Rochelle	4
New Wilmington	1
New York	10
New York City	334
New Zealand	1
Newark	30
Newbridge, Caerphilly	1
Newcastle Emlyn	1
Newcastle upon Tyne	3
Newellton	1
Newland	1
Newnan	1
Newport	1
Newport Beach	2
Newport News	5
Newton	1
Newton Grove	1
Niagara Falls	2
Nice	3
Nicosia	1
Niimegen	1

Wiki Place Of Death Literal	
Montreal	5
Morristown	1
Moscow	5
Mount Vernon	3
Munich	3
Münster	1
Nacogdoches	1
Naples	1
Nashville	7
Navasota	1
Netherlands	1
Neuilly-sur-Seine	2
New Braunfels	1
New Britain	1
New Brunswick	2
New Canaan	1
New Iberia	1
New Jersey	7
New London	2
New Milford	1
New Orleans	71
New Rochelle	2
New York	7
New York City	352
Newark	8
Newburg, Maryland	1
Newport Beach	5
Newport Pagnell	1
Niterói	1
North Adelaide	1
North Carolina	1
North Hollywood	1
North Miami	2
Northampton County	1
Northridge	4
Norwalk	2

Possible Future Directions for the Project

- Refine list by profession (not all names are actually musicians).
- Query Wikidata's 11 familial relationship properties. Many New Orleans-based jazz musicians are part of musical dynasties.
- Examine names from Tulane University list.
- Use links in Wikidata entity pages to query music database sites like MusicBrainz and Discogs (Wikidata has links to corresponding pages on these sites).

Relationship [edit]

Title	ID	Data type	Description	Examples	Inverse
father	P22	Item	father: male parent	Elizabeth II *<father>* George VI	-
mother	P25	Item	mother: female parent	Elizabeth II *<mother>* Queen Elizabeth The Queen Mother	-
brother	P7	Item	brother: subject has the object as their brother (male sibling)	Andy Murray *<brother>* Jamie Murray	-
sister	P9	Item	sister and female: subject has the object as their sister (female sibling)	Elizabeth II *<sister>* Princess Margaret, Countess of Snowdon	-
spouse	P26	Item	spouse: the subject has the object as their spouse (husband, wife, partner, etc.). Use "cohabitant" (P451) for non-married companions	Elizabeth II *<spouse>* Prince Philip, Duke of Edinburgh	-
partner	P451	Item	someone with whom the person is in a relationship without being married. Use "spouse" for married couples.	Simone de Beauvoir *<partner>* Jean-Paul Sartre	-
child	P40	Item	offspring: subject has the object in their family as their offspring son or daughter (independently of their age)	Elizabeth II *<child>* Charles, Prince of Wales	father and mother
stepfather	P43	Item	stepfather: husband of the subject's parent, who is not the subject's biological father	Bill Clinton *<stepfather>* Roger Clinton, Sr.	-
stepmother	P44	Item	stepmother: wife of the subject's parent, who is not the subject's biological mother	Wilhelm Friedemann Bach *<stepmother>* Anna Magdalena Bach	-
relative	P1038	Item	kinship: family member (qualify with "type of kinship", P1039; for direct family member please use specific property)	Moulay Ali Cherif *<relative>* ancestor	-
godparent	P1290	Item	godparent: person who is the godparent of a given person	Queen Victoria *<godparent>* Alexander I of Russia	-

Questions?