

# 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.

**LINKED JAZZ**



# 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.

```
<http://dbpedia.org/resource/Adriana_Evans> <http://xmlns.com/foaf/0.1/name> "Adriana Evans"@en .
<http://dbpedia.org/resource/Adriana_Evans> <http://xmlns.com/foaf/0.1/surname> "Evans"@en .
<http://dbpedia.org/resource/Adriana_Evans> <http://xmlns.com/foaf/0.1/givenName> "Adriana"@en .
<http://dbpedia.org/resource/Adriana_Evans> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://xmlns.com/fo
<http://dbpedia.org/resource/Donni_1> <http://xmlns.com/foaf/0.1/name> "Donni 1"@en .
<http://dbpedia.org/resource/Donni_1> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://xmlns.com/foaf/0.1
<http://dbpedia.org/resource/Donni_1> <http://dbpedia.org/ontology/birthDate> "1969-07-01"^^<http://www.w3.org/200
<http://dbpedia.org/resource/Charlie_Rouse> <http://xmlns.com/foaf/0.1/name> "Charlie Rouse"@en .
<http://dbpedia.org/resource/Charlie_Rouse> <http://xmlns.com/foaf/0.1/surname> "Rouse"@en .
<http://dbpedia.org/resource/Charlie_Rouse> <http://xmlns.com/foaf/0.1/givenName> "Charlie"@en .
<http://dbpedia.org/resource/Charlie_Rouse> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://xmlns.com/fo
<http://dbpedia.org/resource/Charlie_Rouse> <http://dbpedia.org/ontology/birthDate> "1924-04-06"^^<http://www.w3.o
<http://dbpedia.org/resource/Charlie_Rouse> <http://dbpedia.org/ontology/deathDate> "1988-11-30"^^<http://www.w3.o
<http://dbpedia.org/resource/Lee_Aaron> <http://xmlns.com/foaf/0.1/name> "Lee Aaron"@en .
<http://dbpedia.org/resource/Lee_Aaron> <http://xmlns.com/foaf/0.1/surname> "Aaron"@en .
<http://dbpedia.org/resource/Lee_Aaron> <http://xmlns.com/foaf/0.1/givenName> "Lee"@en .
<http://dbpedia.org/resource/Lee_Aaron> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://xmlns.com/foaf/0
<http://dbpedia.org/resource/Lee_Aaron> <http://purl.org/dc/elements/1.1/description> "Canadian heavy metal singer
<http://dbpedia.org/resource/Lee_Aaron> <http://dbpedia.org/ontology/birthDate> "1962-07-21"^^<http://www.w3.org/2
<http://dbpedia.org/resource/Lee_Aaron> <http://dbpedia.org/ontology/birthPlace> <http://dbpedia.org/resource/Cana
<http://dbpedia.org/resource/Phil_Wachsmann> <http://xmlns.com/foaf/0.1/name> "Phil Wachsmann"@en .
<http://dbpedia.org/resource/Phil_Wachsmann> <http://xmlns.com/foaf/0.1/surname> "Wachsmann"@en .
<http://dbpedia.org/resource/Phil_Wachsmann> <http://xmlns.com/foaf/0.1/givenName> "Phil"@en .
<http://dbpedia.org/resource/Phil_Wachsmann> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://xmlns.com/fo
<http://dbpedia.org/resource/Phil_Wachsmann> <http://purl.org/dc/elements/1.1/description> "Ugandan musician"@en .
<http://dbpedia.org/resource/Phil_Wachsmann> <http://dbpedia.org/ontology/birthDate> "1944-08-05"^^<http://www.w3.
<http://dbpedia.org/resource/Nami_Miyahara> <http://xmlns.com/foaf/0.1/name> "Nami Miyahara"@en .
<http://dbpedia.org/resource/Nami_Miyahara> <http://xmlns.com/foaf/0.1/surname> "Miyahara"@en .
<http://dbpedia.org/resource/Nami_Miyahara> <http://xmlns.com/foaf/0.1/givenName> "Nami"@en .
<http://dbpedia.org/resource/Nami_Miyahara> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://xmlns.com/fo
<http://dbpedia.org/resource/Nami_Miyahara> <http://dbpedia.org/ontology/birthDate> "1978-01-24"^^<http://www.w3.o
<http://dbpedia.org/resource/D._D._Jackson> <http://xmlns.com/foaf/0.1/name> "D. D. Jackson"@en .
<http://dbpedia.org/resource/D._D._Jackson> <http://xmlns.com/foaf/0.1/surname> "Jackson"@en .
<http://dbpedia.org/resource/D._D._Jackson> <http://xmlns.com/foaf/0.1/givenName> "D. D."@en .
<http://dbpedia.org/resource/D._D._Jackson> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://xmlns.com/fo
<http://dbpedia.org/resource/D._D._Jackson> <http://dbpedia.org/ontology/birthDate> "1967-01-25"^^<http://www.w3.o
<http://dbpedia.org/resource/Orlando_22Cachaito22_L4C38B3pez> <http://xmlns.com/foaf/0.1/name> "Orlando Lopez"@e
<http://dbpedia.org/resource/Orlando_22Cachaito22_L4C38B3pez> <http://xmlns.com/foaf/0.1/surname> "Lopez"@en .
<http://dbpedia.org/resource/Orlando_22Cachaito22_L4C38B3pez> <http://xmlns.com/foaf/0.1/givenName> "Orlando"@en
<http://dbpedia.org/resource/Orlando_22Cachaito22_L4C38B3pez> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
<http://dbpedia.org/resource/Orlando_22Cachaito22_L4C38B3pez> <http://dbpedia.org/ontology/birthDate> "1933-02-0
<http://dbpedia.org/resource/Orlando_22Cachaito22_L4C38B3pez> <http://dbpedia.org/ontology/deathDate> "2009-02-0
<http://dbpedia.org/resource/Ignasi_Terraza> <http://xmlns.com/foaf/0.1/name> "Ignasi Terraza"@en .
<http://dbpedia.org/resource/Orlando_22Cachaito22_L4C38B3pez> <http://xmlns.com/foaf/0.1/surname> "Terraza"@en .
<http://dbpedia.org/resource/Orlando_22Cachaito22_L4C38B3pez> <http://xmlns.com/foaf/0.1/givenName> "Ignasi"@en .
<http://dbpedia.org/resource/Ignasi_Terraza> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://xmlns.com/fo
<http://dbpedia.org/resource/Ignasi_Terraza> <http://dbpedia.org/ontology/birthDate> "1962-07-14"^^<http://www.w3.
<http://dbpedia.org/resource/Spencer_Williams> <http://xmlns.com/foaf/0.1/name> "Spencer Williams"@en .
<http://dbpedia.org/resource/Spencer_Williams> <http://xmlns.com/foaf/0.1/surname> "Williams"@en .
<http://dbpedia.org/resource/Spencer_Williams> <http://xmlns.com/foaf/0.1/givenName> "Spencer"@en .
<http://dbpedia.org/resource/Spencer_Williams> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://xmlns.com/fo
```

# 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

```
"last name": "Mayes",
"URI": "http://dbpedia.org/resource/Pete\_Mayes",
"first name": "Pete",
"full name": "Pete Mayes"
},
{
  "last name": "Szatmary",
  "URI": "http://dbpedia.org/resource/David\_Szatmary",
  "first name": "David",
  "full name": "David Szatmary"
},
{
  "last name": "Wertico",
  "URI": "http://dbpedia.org/resource/Paul\_Wertico",
  "first name": "Paul",
  "full name": "Paul Wertico"
},
{
  "last name": "Payton",
  "URI": "http://dbpedia.org/resource/Walter\_Payton\_\(musician\)",
  "first name": "Walter",
  "full name": "Walter Payton"
},
{
  "last name": "Suen"
```



# 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.

```
{
  "type": "literal", "value": "Paul Wertico (ur. 5 stycznia 1953 w Chicago
u0119p\u00F3w w Pat Metheny Group w latach 1983-2001.Znany r\u00F3wnie\u017C
biutanckiej p\u0142ycie Apostolisa Anthimosa \u201EDays We Can't Forget\u201D
" } ] ],
http://dbpedia.org/property/wordnet_type": [ { "type": "uri", "value": "ht
http://dbpedia.org/ontology/background": [ { "type": "literal", "value": "
http://dbpedia.org/property/background": [ { "type": "literal", "value": "
http://www.w3.org/2002/07/owl#sameAs" [ { "type": "uri", "value": "http:/
{ "type": "uri", "value": "http://it.dbpedia.org/resource/Paul_Wertico" }
{ "type": "uri", "value": "http://dbpedia.org/resource/Paul_Wertico" },
{ "type": "uri", "value": "http://de.dbpedia.org/resource/Paul_Wertico" },
{ "type": "uri", "value": "http://www.wikidata.org/entity/Q2037163" },
{ "type": "uri", "value": "http://viaf.org/viaf/1987882" },
{ "type": "uri", "value": "http://pl.dbpedia.org/resource/Paul_Wertico" }
{ "type": "uri", "value": "http://yago-knowledge.org/resource/Paul_Wertico
{ "type": "uri", "value": "http://wikidata.dbpedia.org/resource/Q2037163"
{ "type": "uri", "value": "http://zitgist.com/music/artist/3107647b-7bc5-4
http://www.w3.org/ns/prov#wasDerivedFrom": [ { "type": "uri", "value": "ht
http://xmlns.com/foaf/0.1/name": [ { "type": "literal", "value": "Paul Wer
http://xmlns.com/foaf/0.1/homepage": [ { "type": "uri", "value": "http://w
http://xmlns.com/foaf/0.1/isPrimaryTopicOf": [ { "type": "uri", "value": "
http://purl.org/dc/terms/subject": [ { "type": "uri", "value": "http://dbp
{ "type": "uri", "value": "http://dbpedia.org/resource/Category:Pat_Methen
{ "type": "uri", "value": "http://dbpedia.org/resource/Category:Living_peo
{ "type": "uri", "value": "http://dbpedia.org/resource/Category:Musicians_
{ "type": "uri", "value": "http://dbpedia.org/resource/Category:American_j
http://dbpedia.org/property/genre": [ { "type": "uri", "value": "http://db
{ "type": "uri", "value": "http://dbpedia.org/resource/Jazz" },
{ "type": "uri", "value": "http://dbpedia.org/resource/Jazz_fusion" } ] ],
http://dbpedia.org/ontology/abstract": [ { "type": "literal", "value": "Pa
```

# 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 URLs.
- Eventually, I ended up with another JSON directory containing corresponding DBpedia and Wikidata URLs 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.

## place of birth (P19)

---

most specific known (e.g. city instead of country, or hospital instead of city)

birthplace | born in | POB | birth place | location born | born at | birth location | location of birth | location | birth city

## place of death (P20)

---

the most specific known (e.g. city instead of country, or hospital instead of city)

deathplace | died in | death place | POD | location of death | death location

# 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", 1
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(prepare, **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	
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 <i>&lt;father&gt;</i> George VI	-
mother	P25	Item	mother: female parent	Elizabeth II <i>&lt;mother&gt;</i> Queen Elizabeth The Queen Mother	-
brother	P7	Item	brother: subject has the object as their brother (male sibling)	Andy Murray <i>&lt;brother&gt;</i> Jamie Murray	-
sister	P9	Item	sister and female: subject has the object as their sister (female sibling)	Elizabeth II <i>&lt;sister&gt;</i> 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 <i>&lt;spouse&gt;</i> 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 <i>&lt;partner&gt;</i> 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 <i>&lt;child&gt;</i> 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 <i>&lt;stepfather&gt;</i> Roger Clinton, Sr.	-
stepmother	P44	Item	stepmother: wife of the subject's parent, who is not the subject's biological mother	Wilhelm Friedemann Bach <i>&lt;stepmother&gt;</i> 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 <i>&lt;relative&gt;</i> ancestor	-
godparent	P1290	Item	godparent: person who is the godparent of a given person	Queen Victoria <i>&lt;godparent&gt;</i> Alexander I of Russia	-

Questions?

---