

Use OpenAddresses Data To Validate Adresses

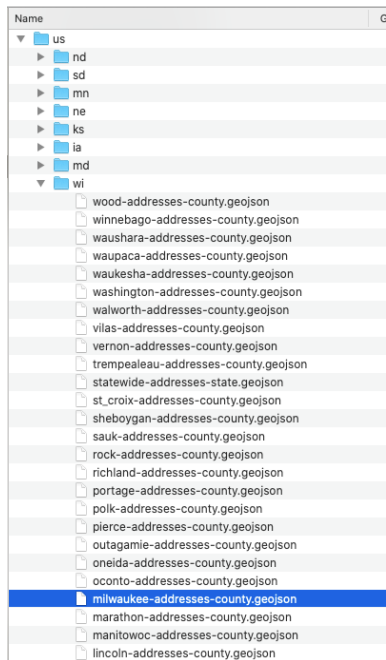
Practical Exam



Goal

OpenAddresses.io provides regular exports of worldwide addresses (we will focus on US south/west/midwest/northeast for now):

- <https://batch.openaddresses.io/data>



```
{
  "type": "Feature",
  "properties": {
    "hash": "394d6a8e3e6cecbf",
    "number": "7705",
    "street": "W LINCOLN AVE",
    "unit": "1",
    "city": "West Allis",
    "district": "",
    "region": "",
    "postcode": "53219",
    "id": ""
  },
  "geometry": {
    "type": "Point",
    "coordinates": [-88.0088621, 43.0025845]
  }
},
{
  "type": "Feature",
  "properties": {
    "hash": "6101cecb71c7bbe",
    "number": "7705",
    "street": "W LINCOLN AVE",
    "unit": "2",
    "city": "West Allis",
    "district": "",
    "region": "",
    "postcode": "53219",
    "id": ""
  },
  "geometry": {
    "type": "Point",
    "coordinates": [-88.0088621, 43.0025845]
  }
},
{
  "type": "Feature",
  "properties": {
    "hash": "81e3634e904916db",
    "number": "1060",
    "street": "N 115TH ST",
    "unit": "106",
    "city": "Wauwatosa",
    "district": "",
    "region": "",
    "postcode": "53226",
    "id": ""
  },
  "geometry": {
    "type": "Point",
    "coordinates": [-88.0551894, 43.0441061]
  }
},
{
  "type": "Feature",
  "properties": {
    "hash": "fbf0248cdd1623ad",
    "number": "12137",
    "street": "W BURLEIGH ST",
    "unit": "2",
    "city": "Wauwatosa",
    "district": "",
    "region": "",
    "postcode": "53222",
    "id": ""
  },
  "geometry": {
    "type": "Point",
    "coordinates": [-88.0649444, 43.0741544]
  }
},
{
  "type": "Feature",
  "properties": {
    "hash": "6c5867d0d98b7e9a",
    "number": "11515",
    "street": "W CLEVELAND AVE",
    "unit": "B231",
    "city": "West Allis",
    "district": "",
    "region": "",
    "postcode": "53227",
    "id": ""
  },
  "geometry": {
    "type": "Point",
    "coordinates": [-88.0560418, 42.9946529]
  }
}
[...]
```

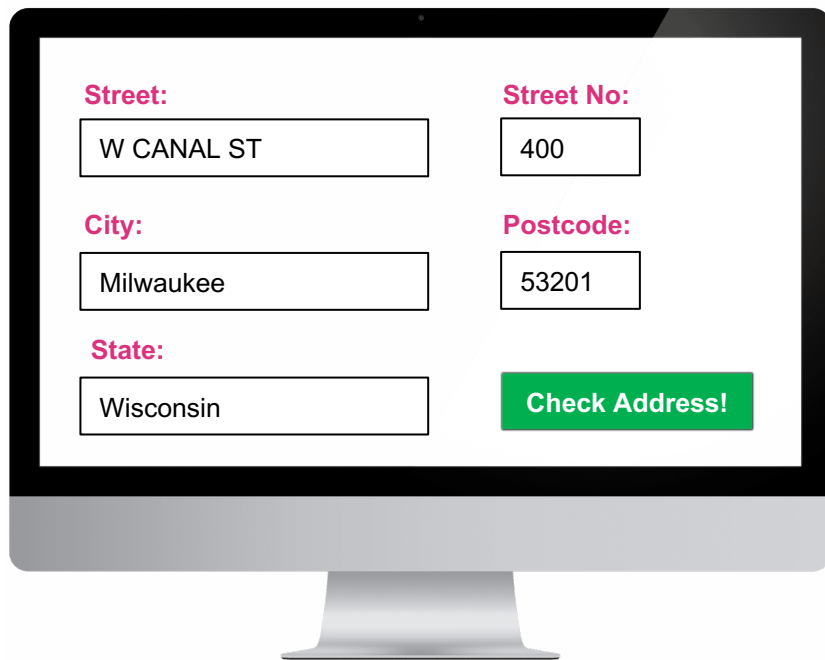


Goal

We want to make use of this data to validate addresses entered on a website, to check whether they are real or not.

Workflow:

- **Gather data** from OpenAddresses.io
- **Save raw data** (*JSON files*) to HDFS (partitioned by state shortcut, e.g. *wi, nd, sd...*)
- **Optimize, reduce** and **clean raw data** and save it to **final** directory on HDFS
- **Export** address data to **end-user database** (e.g. MySQL, MongoDB...)
- Provide a simple **HTML Frontend** which is able to:
 - read from end-user database
 - process user input (Street, City, Postcode...)
 - validate user input against OpenAddress data in end-user database
 - Display result (real or non real address)
- The whole data workflow **must be implemented** within an ETL **workflow tool** (e.g. **Pentaho Data Integration** or **Airflow**) and **run automatically**



The image shows a computer monitor with a web form on the screen. The form has four input fields arranged in a 2x2 grid. The top-left field is labeled 'Street:' and contains the text 'W CANAL ST'. The top-right field is labeled 'Street No:' and contains the text '400'. The bottom-left field is labeled 'City:' and contains the text 'Milwaukee'. The bottom-right field is labeled 'Postcode:' and contains the text '53201'. Below the 'City' field, there is a field labeled 'State:' containing the text 'Wisconsin'. To the right of the 'State' field is a green button with the text 'Check Address!'.

Dataflow: 1. Get Address Data

Get <https://batch.openaddresses.io/data>

1



<http://results.openaddresses.io/>

2

Name	
us	
nd	
sd	
mn	
ne	
ks	
ia	
md	
wi	
wood-addresses-county.geojson	
winnebago-addresses-county.geojson	
waushara-addresses-county.geojson	
waupaca-addresses-county.geojson	
waukesha-addresses-county.geojson	
washington-addresses-county.geojson	
walworth-addresses-county.geojson	
milwaukee-addresses-county.geojson	
vernon-addresses-county.geojson	
trempealeau-addresses-county.geojson	
statewide-addresses-state.geojson	
st_croix-addresses-county.geojson	
sheboygan-addresses-county.geojson	
sauk-addresses-county.geojson	
rock-addresses-county.geojson	
richland-addresses-county.geojson	
portage-addresses-county.geojson	
polk-addresses-county.geojson	
pierce-addresses-county.geojson	
outagamie-addresses-county.geojson	
oneida-addresses-county.geojson	
oconto-addresses-county.geojson	
milwaukee-addresses-county.geojson	
marathon-addresses-county.geojson	
manitowoc-addresses-county.geojson	



`/user/hadoop/openaddresses/raw/us/wi/*.json`
`/user/hadoop/openaddresses/raw/us/nd/*.json`
`/user/hadoop/openaddresses/raw/us/sd/*.json`
...



Dataflow: 2. Raw To Final Transfer



/user/hadoop/openaddresses/**raw**/us/wi/*.json
/user/hadoop/openaddresses/**raw**/us/nd/*.json
/user/hadoop/openaddresses/**raw**/us/sd/*.json
...



1

- move data from **raw** to **final** directory
- Convert/Explode data structure
- **optimize and reduce data structure** for later query purposes if necessary
- remove duplicates if necessary
- ...



/user/hadoop/openaddresses/**final**/us/wi/*.parquet
/user/hadoop/openaddresses/**final**/us/nd/*.parquet
/user/hadoop/openaddresses/**final**/us/sd/*.parquet
...

Dataflow: 3. Enhance Data And Save Results



/user/hadoop/openaddresses/final/us/wi/*.parquet
/user/hadoop/openaddresses/final/us/nd/*.parquet
/user/hadoop/openaddresses/final/us/sd/*.parquet
...



1

- enhance data (e.g. add missing entries of street no's)
- use *Hive*, *Spark* or *PySpark*
- save everything to a end-user database (e.g. *MySQL*, *MongoDB*)



Dataflow: 4. Provide Simple Web Interface

A computer monitor displaying a web form for address validation. The form has four input fields: 'Street' (containing 'W CANAL ST'), 'Street No' (containing '400'), 'City' (containing 'Milwaukee'), and 'State' (containing 'Wisconsin'). There is also a 'Postcode' field (containing '53201') and a green 'Check Address!' button. The labels for the fields are in pink.

- Provide a simple **HTML Frontend** which is able to:
 - read from end-user database
 - process user input (Street, City, Postcode...)
 - validate user input against OpenAddress data in end-user database
 - Display result (real or non real address)