P2: OpenStreetMap Sample Project
Data Wrangling with MongoDB

Map Area: Kyoto (京都), Japan
`https://mapzen.com/metro-extracts/`

Section 1. Problems Encountered in the Map

After downloading the OSM data of Kyoto area and running a provisional `data.py` against it, the following problems are found.

- Garbled Japanese characters (Kanji and Kana) in the JSON file (e.g. '京都' is written as '`\xe4\xba\xac\xe9\x83\xbd`')
- Inconsistent usage of full-width and half-width numbers in the tag values (e.g. ４１５, 1-3-44, ⑪)
- Irrelevant or unnecessary expressions in the address values (e.g. 'yes', '(Kyoto)', '; 京都７号 京都宇治線')
- Inconsistent city names. "city" should be just like **市 (City) but it sometimes also contains **区 (Ward) and/or more detailed address information.
- Inconsistent postcodes (e.g. 123-4567, 1234567)
- Misplaced postcodes and phonenumbers. "street" and "housenumber" sometimes contain numbers that are apparently should be in "postcode" or "phone".

Garbled Japanese characters

Japanese characters could be written properly into JSON file by using UTF-8 encoding and adding an optional argument `ensure_ascii=False` in `process_map` function.

Inconsistent full-width/half-width numbers

I transformed all the full-width numbers into half-width numbers by using unicode normalization:

```
import unicodedata
node[atr] = unicodedata.normalize('NFKC', unicode(element.get(atr)))
```

where `atr` denotes an attribute of the element.

Irrelevant expressions in address values

From address values I removed 'yes', parenthesized words like '(Kyoto)', and all the expressions that are separated by ';'.

Inconsistent city values

If a "city" value contains prefecture(**府), city(**市), ward(**区), and other more detailed street names (e.g. 堀川通), only its city part was extracted and stored, while the remaining information after **区 was moved to "street" value.

Inconsistent postcodes

Postal codes were shaped and standardized to be 7-digit numbers by removing hyphens.

Misplaced postcodes and phonenumbers

If "housenumber" or "street" values matched the regular expressions for either "postcode" or "phone", those values were moved to the right places.

Section 2. Data Overview

This section provides a statistical overview about the dataset and the MongoDB queries to find them.

- File sizes

```
kyoto_japan.osm:  193M
kyoto_japan.osm.json:  216M
```

- Number of documents, nodes, and ways

```
> db.kyoto.find().count()
1012667
> db.kyoto.find({''type'':  ''node''}).count()
880936
> db.kyoto.find({''type'':  ''way''}).count()
131731
```

- Top 1 contributing user

```
> db.kyoto.aggregate([{''$group'':  {''_id'':''$created.user'',
''count'':{''$sum'':1}}}, {''$sort'':  {''count'':  -1}}, {''$limit'':  1}])
{u'_id':  u'kyoka', u'count':  444501}
```

- Top 3 appearing cities

```
> db.kyoto.aggregate([{''$match'':  {''address.city'':  {''$exists'':1}}},
{''$match'':  {''address.city'':  {''$ne'':  ''''}}}, {''$group'':
{''_id'':''$address.city'', ''count'':{''$sum'':1}}}, {''$sort'':
{''count'':  -1}}, {''$limit'':  3}])
[{u'_id':  u'京都市', u'count':  39}, {u'_id':  u'宇治市', u'count':  6},
{u'_id':  u'大津市', u'count':  2}]
```

- Top 5 appearing amenities

```
> db.kyoto.aggregate([{‘‘$match’’: {‘‘amenity’’:{‘‘$exists’’:1}}},
{‘‘$group’’: {‘‘_id’’:’’$amenity’’, ‘‘count’’:{‘‘$sum’’:1}}}, {‘‘$sort’’:
{‘‘count’’: -1}}, {‘‘$limit’’: 5}])
[{u‘_id’: u‘parking’, u‘count’: 2618}, {u‘_id’: u‘vending_machine’,
u‘count’: 728}, {u‘_id’: u‘restaurant’, u‘count’: 679}, {u‘_id’:
u‘school’, u‘count’: 534}, {u‘_id’: u‘social_facility’, u‘count’: 503}]
```

Section 3. Additional Ideas

Restrictions on the format of address values

Out of 685 documents that contain an “address” dictionary, only 79 (11.5%) have a “city” value and 73 (10.6%) have a “postcode” value after data cleaning process (See below). I guess this can be largely due to the arbitrary and inconsistent inputs to those values. Setting restrictions on the characters and numbers that are accepted for those values or introducing selection forms so that only standardized address information can be entered will improve the data significantly.

```
> db.kyoto.find({‘‘address’’ : {‘‘$exists’’:1}}).count()
685
> db.kyoto.find({‘‘address.city’’ : {‘‘$exists’’:1}}).count()
79
> db.kyoto.find({‘‘address.postcode’’ : {‘‘$exists’’:1}}).count()
73
```

Making English names mandatory

While there are 18663 documents that contain a “name” value, only a small fraction of them (3212) have an English name (“name:en”). Some of “address” values, on the other hand, are only written in English. These inconsistencies make it difficult to run MongoDB queries to get complete statistics about the dataset since the same names or addresses are recognized as different. One solution would be to make it mandatory to provide an English version of “name” and “address” or to somehow prepare a mapping function (at least a rough one) from Japanese names to English names.

```
> db.kyoto.find({‘‘name’’ : {‘‘$exists’’:1}}).count()
18663
> db.kyoto.find({‘‘name:en’’ : {‘‘$exists’’:1}}).count()
3212
```