```
#!/usr/bin/env python
# -*- coding: utf-8 -*-
from pprint import pprint
from pymongo import MongoClient
def get_db(db_name):
    client = MongoClient('localhost:27017')
    db = client[db_name]
    return db

class AggregationPipeline(object):
    def __init__(self):
        pass

    def top_user(self):
        """ returns pipeline for user with most contributions """
        return [{
            '$group': {
                '_id': '$created.user',
                'count': {
                    '$sum': 1
                }
            }
        }, {
            '$sort': {
                'count': -1
            }
        }, {
            '$limit': 1
        }]

    def single_post_users(self):
        """ returns pipeline for number of users contributing only once """
        return [{
            '$group': {
                '_id': '$created.user',
                'count': {
                    '$sum': 1
                }
            }
        }, {
            '$group': {
                '_id': '$count',
                'num_users': {
                    '$sum': 1
                }
            }
        }, {
            '$sort': {
                '_id': 1
            }
        }, {
            '$limit': 1
        }]

    def zipcodes(self):
        """ returns pipeline for collecting zipcodes """
        return [{
            '$match': {
                'zipcodes': {
                    '$exists': 1
                }
            }
        }, {
            '$unwind': '$zipcodes'
        }, {
            '$group': {
                '_id': '$zipcodes'
            }
        }, {
            '$group': {
                '_id': 'Zip Codes in Austin',
                'count': {
                    '$sum': 1
                },
                'zipcodes': {
                    '$push': '$_id'
                },
            }
        }]

    def most_common_buildings(self):
        """ returns pipeline for top 10 building types """
        return [{
            '$match': {
                'building': {
                    '$exists': 1
                }
            }
        }, {
            '$group': {
                '_id': '$building',
                'count': {
                    '$sum': 1
                }
            }
        }, {
```

```
        '$sort': {
            'count': -1
        }
    }, {
        '$limit': 10
    }]

def most_common_address(self):
    """ returns pipeline for most common address """
    return [{
        '$match': {
            'address.street': {
                '$exists': 1
            }
        }
    }, {
        '$group': {
            '_id': '$address.street',
            'count': {
                '$sum': 1
            }
        }
    }, {
        '$sort': {
            'count': -1
        }
    }, {
        '$limit': 1
    }]

def nodes_without_addresses(self):
    """ returns pipeline for nodes without addresses """
    return [{
        '$match': {
            'type': 'node',
            'address': {
                '$exists': 0
            }
        }
    }, {
        '$group': {
            '_id': 'Nodes without addresses',
            'count': {
                '$sum': 1
            }
        }
    }]
    def nodes_with_tiger_data(self):
        """ returns pipeline for nodes with TIGER data """
        return [{
            '$match': {
                'tiger': {
                    '$exists': 1
                }
            }
        }, {
            '$group': {
                '_id': 'Nodes with TIGER data',
                'count': {
                    '$sum': 1
                }
            }
        }]


if __name__ == '__main__':
    db = get_db('test')
    pipeline = AggregationPipeline()
    print "top contributing user"
    pprint(db.austin.aggregate(pipeline.top_user()))
    raw_input("Press enter to continue...\n")

    print "single post users"
    pprint(db.austin.aggregate(pipeline.single_post_users()))
    raw_input("Press enter to continue...\n")

    print "zipcodes"
    pprint(db.austin.aggregate(pipeline.zipcodes()))
    raw_input("Press enter to continue...\n")
```