Q1. Find the restaurant ID of “Caffe Dante”.

```
db.restaurants.find(
{name:'Caffe Dante'},
{_id:0,restaurant_id:1}
)
```

Q2. Find all restaurants whose name has “Steak” in it, return only the restaurant’s ids and names.

```
db.restaurants.find(
{name:/.*Steak.*/},
{name:1,_id:0,restaurant_id:1}
)
```

Q3. Find the names of all restaurants that serve either Italian or American cuisine and are located in the Brooklyn borough. (the result of cuisine is just "Italian" or "American")

```
db.restaurants.find(
{$and: [{cuisine:{$in:["American","Italian"]}},
  {borough:"Brooklyn"}]},
{name:1,_id:0}
)
```

Q4. Return the list of boroughs ranked by the number of American restaurants (cuisine with the word "American") in it. That is, for each borough, find how many restaurants serve American cuisine and print the borough and the number of such restaurants sorted by this number in descending order.

```
db.restaurants.aggregate(
[{$match:{cuisine:/.*American.*/}},
{$group: {_id:"$borough",count:{$sum:1}}},
{$sort:{count:-1}}]
)
```

Q5. Find the top 5 Chinese restaurants (cuisine with the word "Chinese") in Manhattan that have the highest total score. Return for each restaurant the restaurants’ name and the total score. Hint: You can use “$unwind”.

```
db.restaurants.aggregate(
[{$match: {$and: [{cuisine:/.*Chinese.*/},
  {borough:"Manhattan"}]}},
{$unwind:{path:"$grades",preserveNullAndEmptyArrays: true}},
{$group:{. _id:"$restaurant_id",
   name:{$first:"$name"},
   score:{$sum:"$grades.score"}}},
{$sort:{score:-1}},
{$project:{_id:0,name:1,score:1}},
{$limit:5}]
)
```

Q6. Consider a rectangle area on the location field, in which the vertices are [-74 , 40.5] , [-74 , 40.7] , [-73.5 , 40.5] and [-73.5 , 40.7]. Find the number of restaurants in this area that have received a grade score (at least one) more than 70.

```
db.restaurants.find(
{$and:[ {"address.coord": {$geoWithin: {$box:[[-74,40.5],[-73.5,40.7]]}}},
{"grades.score":{$gt:70}}]},
{_id:0,name:1,grades:1}).count()
```

Q7. Find the top 10 zipcodes with the largest population, return the zipcode, the city name and the state

```
db.zips.aggregate({$sort: {pop: -1}},{$limit:10},{$project:{"pop":0,"loc":0}});
```

Q8. Find the largest city in each state, return the city name and the state
(Some answers return a zipcode which comes from a not-updated question. Points are not deducted if the logic for city name and state is correct)

```
db.zips.aggregate(
[{$group: { _id: { state:"$state",city:"$city"}, pop: {$sum: "$pop"}}},
 {$sort:{pop:-1}},
 {$group:{_id:"$_id.state", city: {$first:"$_id.city"}}}
]);
```

Q9. Find the states where the average population of cities is larger than 10000, return the population and the state

```
db.zips.aggregate(
[{$group: { _id: { state:"$state",city:"$city"}, pop: {$sum: "$pop"}}},
 {$group:{_id:"$_id.state",avgPop: {$avg:"$pop"}}},
 {$match:{avgPop:{$gt: 10000}}}
]);
```

Q10. Find the top 5 cities nearest to [-70,40], return only the city name

```
db.zips.find({loc:{$near:[ -70, 40]}}, {"_id":0,city:1}).limit(5);
```