# Goal of the project:

Here we want to find the list of restaurants in each city of USA, how many review they have received in yelp and how many stars(1,2,2.5 stars etc) they have received from users. Now in a city(identified by zip code) there may be multiple branch of a restaurant. So we also count how many outlets of a restaurants are in a city.

Now, in the yelp business database we have found few restaurants which not in  restaurant database (provided by primer.json). We have set their count to 0.

Finally, we have made a simple linear ***SCORE*** for each of the restaurants with the following formula

**Final Score =( # of review ×# of stars in a single review) ÷number  of outlet of the restaurent in  the city**

***Note: Remember that, this rating is  just a sample. It is neither mathematically sound nor normalized. So, it is not possible to use this score to compare 2 restaurants. This score is just to give the users some idea about how many people have praised the restaurant chain. We also projected few other features about the restaurants (i.e. good for dinner, ambience, outdoor sitting, attire etc). These fields can be used to generate a complex ranking system.***

### Description of the Code:

Step 1: Get all restaurants from business collection and save as restaurant_business collection

```
db.business.aggregate(
              { $match: {categories:{$in:["Restaurants"]}}},
              {$out:"restaurant_business"}
 )
```

Step 2: Full outer  Join with “restaurants” collection and save as “restaurant_business_info”

```
db.restaurant_business.aggregate([{
   $lookup: {
      from: "restaurants",
      localField: "name",
      foreignField: "name",
      as: "restaurant_details"
   }
},{
$project:{id : 0}},
{
   $unwind:
```

```
 {
    path: "$restaurant_details",
    preserveNullAndEmptyArrays: false
  }
 },
{
  $out:"restaurant_business_info"
}
])
```

Step 3: Full outer  Join with “zipcodes” collection and save as “restaurant_business_zip_info”. This collection actually contains data from full outer join of 3 collections a) business b)restaurants 3) zipcodes

```
db.restaurant_business_info.aggregate([{
  $lookup: {
    from: "zipcodes",
    localField: "restaurant_details.address.zipcode",
    foreignField: "_id",
    as: "zip_info"
  }
},{
$project:{_id : 0}},
{
  $unwind:
  {
    path: "$zip_info",
    preserveNullAndEmptyArrays: false
  }
 },{
  $out:"restaurant_business_zip_info"
}
])
```

Step 4: Create a map reduce pipeline. This will compare the values of state, city, zipcode and restaurant name over the restaurant_business_zip_info collection. This step basically calculates inner join of those 3 collections. Moreover it also counts number of outlets of a restaurant in a city

--This pipeline also contains a finalize stage. This stage actually calculates the final score
-- finally We store the restaurants with their score in “Joined_Data” collection

```
var mapFunction = function () {
  var key = this.business_id;
  var value =
  {
    business_id : this.business_id,
    name : this.name,
    neighborhood : this.neighborhood,
    borough :  this.restaurant_details.borough,
    city : this.city,
    state : this.state,
    postal_code : this.postal_code,
    stars : this.stars,
    review_count : this.review_count,
    RestaurantsAttire : this.attributes.RestaurantsAttire,
    Alcohol : this.attributes.Alcohol,
    OutdoorSeating : this.attributes.OutdoorSeating,
    cuisine :  this.restaurant_details.cuisine,
    restaurant_id : this.restaurant_details.restaurant_id,
    pop : this.zip_info.pop,
    stars : this.stars,
    review_count : this.review_count,
    count : 0
  };
if(
    (this.name == this.restaurant_details.name) &&
    (this.postal_code == this.restaurant_details.zipcode) &&
    (this.postal_code == this.zip_info._id) &&
    (this.state == this.zip_info.state)
  ){
    value.count = 1;
  }
  emit(key, value);
};

var reduceFunction = function (key, values) {
  var reducedObject = {
      "business_id" : key,
      "name" : "",
      "neighborhood" : "",
      "borough" : "",
      "city" : "",
      "state" : "",
      "postal_code" : "",
      "stars" : 0,
      "review_count" : 0,
      "RestaurantsAttire" : "caxsual",
      "Alcohol" : "none",
      "OutdoorSeating" : false,
      "cuisine" : "",
      "restaurant_id" : "",
      "cityPop" : 0,
      "count": 0
  };
  values.forEach(function (value) {
    reducedObject.business_id = value.business_id;
    reducedObject.name = value.name;
    reducedObject.neighborhood = value.neighborhood;
    reducedObject.borough =  value.borough;
    reducedObject.city = value.city;
    reducedObject.state = value.state;
    reducedObject.postal_code = value.postal_code;
    reducedObject.stars += value.stars;
    reducedObject.review_count += value.review_count;
    reducedObject.RestaurantsAttire = value.RestaurantsAttire;
    reducedObject.Alcohol = value.Alcohol;
    reducedObject.OutdoorSeating = value.OutdoorSeating;
    reducedObject.cuisine =  value.cuisine;
    reducedObject.restaurant_id = value.restaurant_id;
    reducedObject.cityPop += value.pop;
    reducedObject.count += 1;
    }
  );
  return reducedObject;
};


var finalizeFunction = function (key, reducedValue) {

      if ((reducedValue.stars > 0) && (reducedValue.review_count > 0) && (reducedValue.count > 0)){
          reducedValue.final_score = (reducedValue.stars * reducedValue.review_count)/reducedValue.count;
      }
```

```
     else{
       reducedValue.final_score = 0;
     }
     return reducedValue;
 };

db.restaurant_business_zip_info.mapReduce(mapFunction,
 reduceFunction,
 {
     out: "joined_data",
     finalize: finalizeFunction
   }
)
```

## Example:

Assume, we want to find the restaurant in KENT that includes following features

a) Full bar
b) Final score in out ranking > 50

To find that we can do the following query:

```
db.joined_data.find(
{'value.postal_code':"44240",'value.Alcohol':"full_bar","value.final_score":{$g
t : 50}}
)
```

## Result:

```
"name" : "Pizza Hut",
"neighborhood" : "",
"borough" : "Bronx",
"city" : "Kent",
"state" : "OH",
"postal_code" : "44240",
"stars" : 44,
"review_count" : 44,
"RestaurantsAttire" : "casual",
"Alcohol" : "full_bar",
"OutdoorSeating" : false,
"cuisine" : "Pizza",
"restaurant_id" : "41703517",
"cityPop" : 458597,
"count" : 11,
"final_score" : 176
```