Group C: Assignment No.4

Implement Map reduces operation with suitable example on above MongoDB database.

1) Return the Total Price Per Customer

```
> var reduceFunction1 = function(keyCustId, valuesPrices) {
...     return Array.sum(valuesPrices);
... };
> db.orders.mapReduce(mapFunction1,reduceFunction1,{ out: "map_reduce_example" })
{ "result" : "map_reduce_example", "ok" : 1 }
> db.map_reduce_example.find().sort( { _id: 1 } )
{ "_id" : "Ant O. Knee", "value" : 95 }
{ "_id" : "Busby Bee", "value" : 125 }
{ "_id" : "Cam Elot", "value" : 60 }
{ "_id" : "Don Quis", "value" : 155 }
> db.orders.aggregate([
...     { $group: { _id: "$cust_id", value: { $sum: "$price" } } },
...     { $out: "agg_alternative_1" }
... ])
> db.agg_alternative_1.find().sort( { _id: 1 } )
{ "_id" : "Ant O. Knee", "value" : 95 }
{ "_id" : "Busby Bee", "value" : 125 }
{ "_id" : "Cam Elot", "value" : 60 }
{ "_id" : "Don Quis", "value" : 155 }
>
```

2) Calculate Order and Total Quantity with Average Quantity Per Item

```
> var mapFunction2 = function() {
...     for (var idx = 0; idx < this.items.length; idx++) {
...        var key = this.items[idx].sku;
...        var value = { count: 1, qty: this.items[idx].qty };
...
...        emit(key, value);
...     }
... };
> var reduceFunction2 = function(keySKU, countObjVals) {
...     reducedVal = { count: 0, qty: 0 };
...
...     for (var idx = 0; idx < countObjVals.length; idx++) {
...         reducedVal.count += countObjVals[idx].count;
...         reducedVal.qty += countObjVals[idx].qty;
...     }
...
...     return reducedVal;
... };
> var finalizeFunction2 = function (key, reducedVal) {
...    reducedVal.avg = reducedVal.qty/reducedVal.count;
...    return reducedVal;
... };
> db.orders.mapReduce(
...     mapFunction2,
...     reduceFunction2,
...     {
...       out: { merge: "map_reduce_example2" },
...       query: { ord_date: { $gte: new Date("2020-03-01") } },
...       finalize: finalizeFunction2
...     }
...   );
{ "result" : "map_reduce_example2", "ok" : 1 }
```

```
> db.map_reduce_example2.find().sort( { _id: 1 } )
{ "_id" : "apples", "value" : { "count" : 4, "qty" : 35, "avg" : 8.75 } }
{ "_id" : "carrots", "value" : { "count" : 2, "qty" : 15, "avg" : 7.5 } }
{ "_id" : "chocolates", "value" : { "count" : 3, "qty" : 15, "avg" : 5 } }
{ "_id" : "oranges", "value" : { "count" : 7, "qty" : 63, "avg" : 9 } }
{ "_id" : "pears", "value" : { "count" : 1, "qty" : 10, "avg" : 10 } }
```

```
> db.orders.aggregate( [
...     { $match: { ord_date: { $gte: new Date("2020-03-01") } } },
...     { $unwind: "$items" },
...     { $group: { _id: "$items.sku", qty: { $sum: "$items.qty" }, orders_ids: { $addToSet: "$_id" } }  },
...     { $project: { value: { count: { $size: "$orders_ids" }, qty: "$qty", avg: { $divide: [ "$qty", { $size: "$orders_ids" } ] } } } }
,
...     { $merge: { into: "agg_alternative_3", on: "_id", whenMatched: "replace",  whenNotMatched: "insert" } }
... ] )
> db.agg_alternative_3.find().sort( { _id: 1 } )
{ "_id" : "apples", "value" : { "count" : 4, "qty" : 35, "avg" : 8.75 } }
{ "_id" : "carrots", "value" : { "count" : 2, "qty" : 15, "avg" : 7.5 } }
{ "_id" : "chocolates", "value" : { "count" : 3, "qty" : 15, "avg" : 5 } }
{ "_id" : "oranges", "value" : { "count" : 7, "qty" : 63, "avg" : 9 } }
{ "_id" : "pears", "value" : { "count" : 1, "qty" : 10, "avg" : 10 } }
>
```

3) Total items for each type

```
> db.orders.mapReduce(function(){for(i=0;i<this.items.length;i++){var key=this.items[i].sku;value=this.items[i].qty;emit(key,value)}},fu
nction(key,values){return Array.sum(values);},{out:"units"})
{ "result" : "units", "ok" : 1 }
> db.units.find().pretty()
{ "_id" : "chocolates", "value" : 15 }
{ "_id" : "carrots", "value" : 15 }
{ "_id" : "pears", "value" : 10 }
{ "_id" : "oranges", "value" : 63 }
{ "_id" : "apples", "value" : 35 }
>
```