Assignment No. 12

Assignment Title:Create the following schema
Order (id, amount ,status)

Cus id Amount Status
A1 400 P
B1 300 D
A1 200 F
C1 200 F
B1 700 P
B1 800 P
Status: P=”Pending”, D= “Delivered”, F= “Failed”

Implement the following using Map Reduce function
1. Find the sum of amount of each customer whose status is P
2. Find the average amount of each customer
3. Find the min amount of each customer
4. Find the max amount of each customer whose status is F

Queries:

To implement the specified operations using the MapReduce function in MongoDB, you can follow these steps. We'll assume that you have a collection named "Order" with the given schema:

```
db.createCollection("Order")
db.Order.insertMany([
        { Cus_id: "A1", Amount: 400, Status: "P" },
        { Cus_id: "B1", Amount: 300, Status: "D" },
        { Cus_id: "A1", Amount: 200, Status: "F" },
        { Cus_id: "C1", Amount: 200, Status: "F" },
        { Cus_id: "B1", Amount: 700, Status: "P" },
        { Cus_id: "B1", Amount: 800, Status: "P" }
])
```

Step 1: Find the sum of the amount of each customer whose status is P

```
var mapFunction1 = function() {
        if (this.Status === "P") {
        emit(this.Cus_id, this.Amount);
        }
};

var reduceFunction1 = function(customer, amounts) {
```

```
        return Array.sum(amounts);
};

db.Order.mapReduce(
        mapFunction1,
        reduceFunction1,
        { out: "sum_amounts_P" }
)

db.sum_amounts_P.find()
```

Step 2: Find the average amount of each customer

```
var mapFunction2 = function() {
        emit(this.Cus_id, { count: 1, total: this.Amount });
};

var reduceFunction2 = function(customer, values) {
        var reducedVal = { count: 0, total: 0 };

        for (var idx = 0; idx < values.length; idx++) {
        reducedVal.count += values[idx].count;
        reducedVal.total += values[idx].total;
        }

        return reducedVal;
};

var finalizeFunction2 = function(customer, reducedVal) {
        reducedVal.average = reducedVal.total / reducedVal.count;
        return reducedVal;
};

db.Order.mapReduce(
        mapFunction2,
        reduceFunction2,
        { out: "average_amounts", finalize: finalizeFunction2 }
)

db.average_amounts.find()
```

Step 3: Find the min amount of each customer

```
var mapFunction3 = function() {
        emit(this.Cus_id, this.Amount);
};
```

```
var reduceFunction3 = function(customer, amounts) {
        return Math.min.apply(null, amounts);
};

db.Order.mapReduce(
        mapFunction3,
        reduceFunction3,
        { out: "min_amounts" }
)

db.min_amounts.find()
```

Step 4: Find the max amount of each customer whose status is F

```
var mapFunction4 = function() {
        if (this.Status === "F") {
        emit(this.Cus_id, this.Amount);
        }
};

var reduceFunction4 = function(customer, amounts) {
        return Math.max.apply(null, amounts);
};

db.Order.mapReduce(
        mapFunction4,
        reduceFunction4,
        { out: "max_amounts_F" }
)

db.max_amounts_F.find()
```