PRACTICAL-4

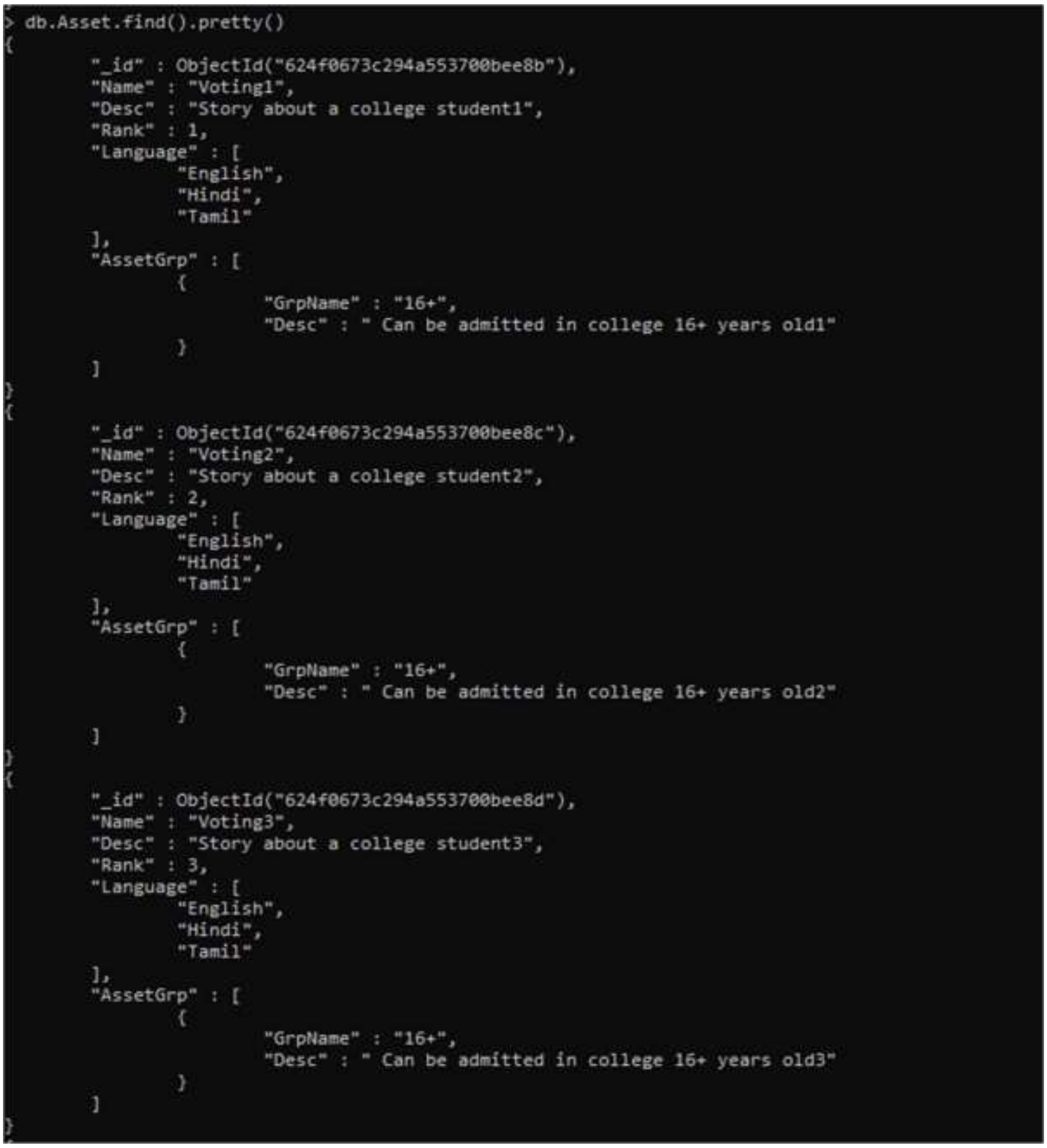
AIM: Indexing ,Aggregation and Map Reduce in NoSQL-DB.

I. PRACTICE QUESTIONS:

1. Indexing :- Query :- for(var

```
iCounter=1;iCounter<=

1000000;iCounter++)

{

db.Asset.ins

ert(

{

"Name":"Voting"+iCounter,

"Desc":"Story about a college student"+iCounter,

"Rank":iCounter,

"Language":["English","Hindi","Tamil"],

"AssetGrp":[ { "GrpName":"16+", "Desc":" Can be
admitted in college

16+ years old"+iCounter } ]

})

}
```

Output :-



```
> db.Asset.find().pretty()
{
        "_id" : ObjectId("624f0673c294a553700bee8b"),
        "Name" : "Voting1",
        "Desc" : "Story about a college student1",
        "Rank" : 1,
        "Language" : [
                "English",
                "Hindi",
                "Tamil"
        ],
        "AssetGrp" : [
                {
                        "GrpName" : "16+",
                        "Desc" : " Can be admitted in college 16+ years old1"
                }
        ]
}
{
        "_id" : ObjectId("624f0673c294a553700bee8c"),
        "Name" : "Voting2",
        "Desc" : "Story about a college student2",
        "Rank" : 2,
        "Language" : [
                "English",
                "Hindi",
                "Tamil"
        ],
        "AssetGrp" : [
                {
                        "GrpName" : "16+",
                        "Desc" : " Can be admitted in college 16+ years old2"
                }
        ]
}
{
        "_id" : ObjectId("624f0673c294a553700bee8d"),
        "Name" : "Voting3",
        "Desc" : "Story about a college student3",
        "Rank" : 3,
        "Language" : [
                "English",
                "Hindi",
                "Tamil"
        ],
        "AssetGrp" : [
                {
                        "GrpName" : "16+",
                        "Desc" : " Can be admitted in college 16+ years old3"
                }
        ]
}
```

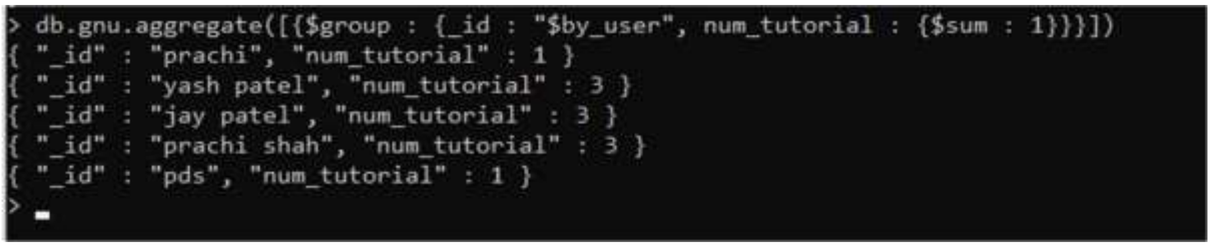
2. Aggregation :-

- **Ex-1 : Create collection name as "gnu"**

  Add 10 relevant documents in same collection.

Query :- db.gnu.aggregate([{$group : {_id : "$by_user",

num_tutorial : {$sum : 1}}}]) Output :-

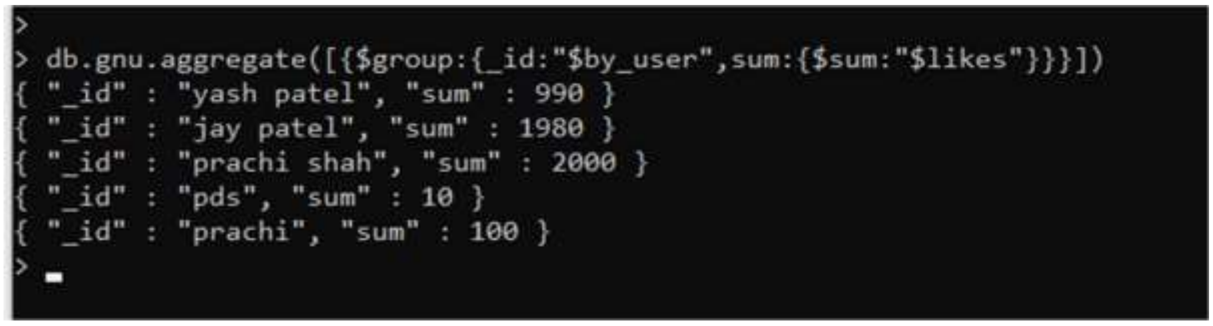


```
> db.gnu.aggregate([{$group : {_id : "$by_user", num_tutorial : {$sum : 1}}}])
{ "_id" : "prachi", "num_tutorial" : 1 }
{ "_id" : "yash patel", "num_tutorial" : 3 }
{ "_id" : "jay patel", "num_tutorial" : 3 }
{ "_id" : "prachi shah", "num_tutorial" : 3 }
{ "_id" : "pds", "num_tutorial" : 1 }
> _
```

Query:-

db.gnu.aggregate([{$group:{_id:"$by_user",sum:{$sum:"$

likes"}}}]) Output :-

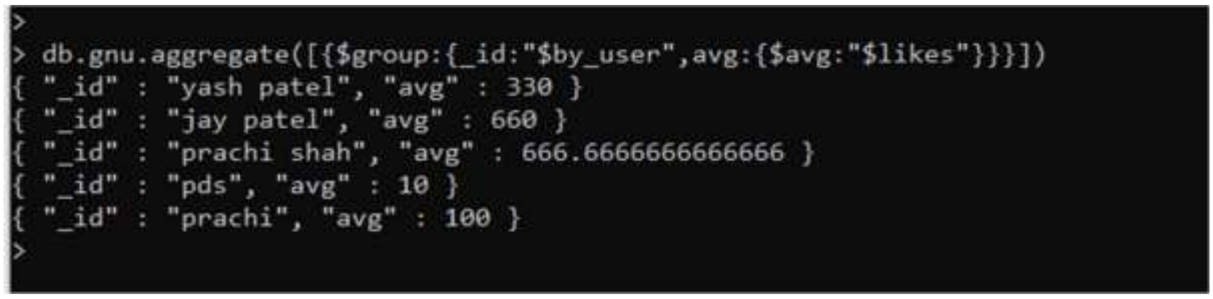


```
>
> db.gnu.aggregate([{$group:{_id:"$by_user",sum:{$sum:"$likes"}}}])
{ "_id" : "yash patel", "sum" : 990 }
{ "_id" : "jay patel", "sum" : 1980 }
{ "_id" : "prachi shah", "sum" : 2000 }
{ "_id" : "pds", "sum" : 10 }
{ "_id" : "prachi", "sum" : 100 }
> _
```

Query :-

db.gnu.aggregate([{$group:{_id:"$by_user",avg:{$avg:"$li

kes"}}}]) Output :-



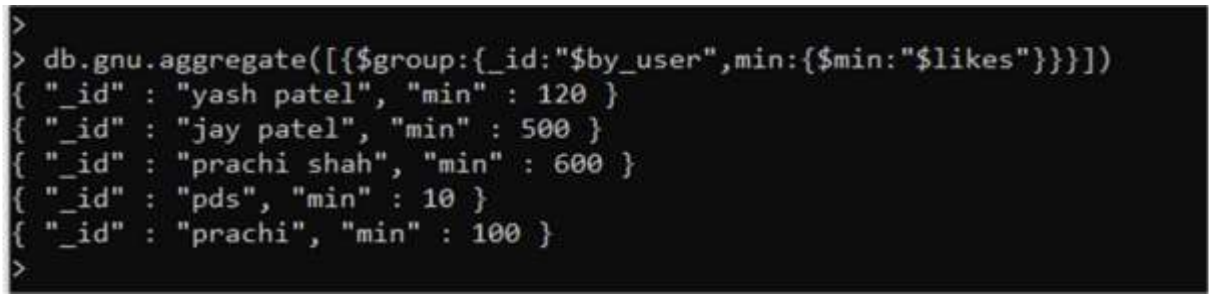
```
>
> db.gnu.aggregate([{$group:{_id:"$by_user",avg:{$avg:"$likes"}}}])
{ "_id" : "yash patel", "avg" : 330 }
{ "_id" : "jay patel", "avg" : 660 }
{ "_id" : "prachi shah", "avg" : 666.6666666666666 }
{ "_id" : "pds", "avg" : 10 }
{ "_id" : "prachi", "avg" : 100 }
>
```

Query :-

db.gnu.aggregate([{$group:{_id:"$by_user",min:{$min:"$li

kes"}}}])

Output :-

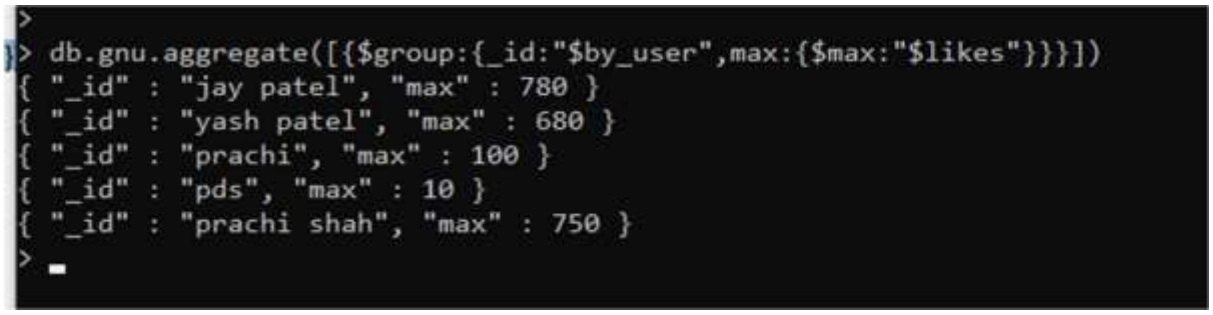


```
>
> db.gnu.aggregate([{$group:{_id:"$by_user",min:{$min:"$likes"}}}])
{ "_id" : "yash patel", "min" : 120 }
{ "_id" : "jay patel", "min" : 500 }
{ "_id" : "prachi shah", "min" : 600 }
{ "_id" : "pds", "min" : 10 }
{ "_id" : "prachi", "min" : 100 }
>
```

Query :-

db.gnu.aggregate([{$group:{_id:"$by_user",max:{$max:"$

likes"}}}]) Output :-

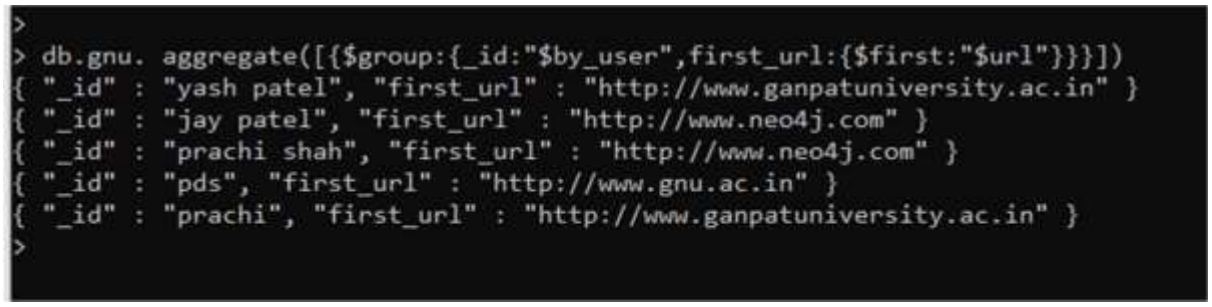


```
>
> db.gnu.aggregate([{$group:{_id:"$by_user",max:{$max:"$likes"}}}])
{ "_id" : "jay patel", "max" : 780 }
{ "_id" : "yash patel", "max" : 680 }
{ "_id" : "prachi", "max" : 100 }
{ "_id" : "pds", "max" : 10 }
{ "_id" : "prachi shah", "max" : 750 }
> _
```

Query :- db.gnu.

aggregate([{$group:{_id:"$by_user",first_url:{$first:"$

url"}}}]) Output :-



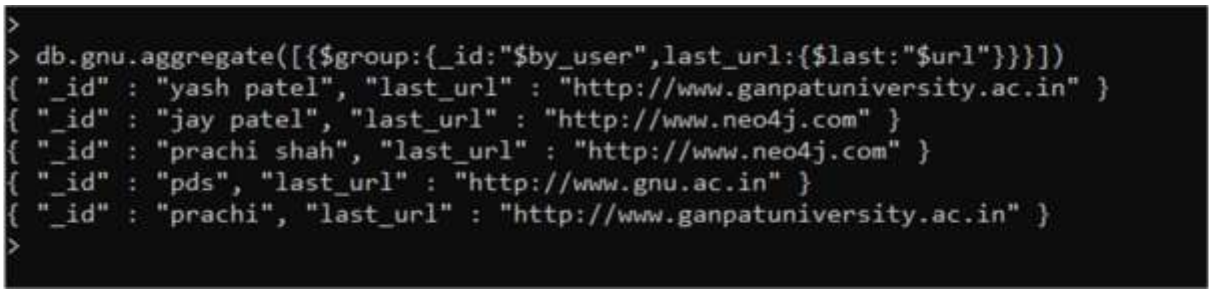
```
>
> db.gnu. aggregate([{$group:{_id:"$by_user",first_url:{$first:"$url"}}}])
{ "_id" : "yash patel", "first_url" : "http://www.ganpatuniversity.ac.in" }
{ "_id" : "jay patel", "first_url" : "http://www.neo4j.com" }
{ "_id" : "prachi shah", "first_url" : "http://www.neo4j.com" }
{ "_id" : "pds", "first_url" : "http://www.gnu.ac.in" }
{ "_id" : "prachi", "first_url" : "http://www.ganpatuniversity.ac.in" }
>
```

Query :-

db.gnu.aggregate([{$group:{_id:"$by_user",last_url:{$last

:"$url"}}}])

Output :-



```
>
> db.gnu.aggregate([{$group:{_id:"$by_user",last_url:{$last:"$url"}}}])
{ "_id" : "yash patel", "last_url" : "http://www.ganpatuniversity.ac.in" }
{ "_id" : "jay patel", "last_url" : "http://www.neo4j.com" }
{ "_id" : "prachi shah", "last_url" : "http://www.neo4j.com" }
{ "_id" : "pds", "last_url" : "http://www.gnu.ac.in" }
{ "_id" : "prachi", "last_url" : "http://www.ganpatuniversity.ac.in" }
>
```

 EX-2 :

Create a collection called purchase_orders having fildes product (toothbrush , guitar , milk , pizza) , price , customer_name insert 10 records into collections.

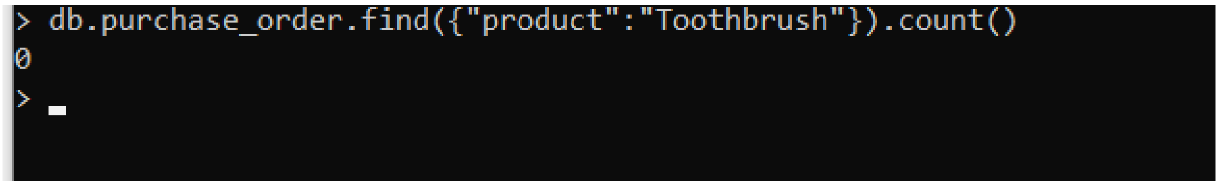
 Query: -

1. **find out how many toothbrushes were sold.**

Query: -

db.purchase_order.find({"product":"Toothbrush"}).c

ount() Output :-

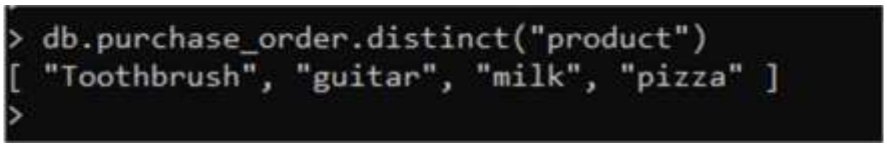


```
> db.purchase_order.find({"product":"Toothbrush"}).count()
0
> 
```

2. **find the list of all products sold. Query: -**

db.purchase_order.distinct("product")

Output :-

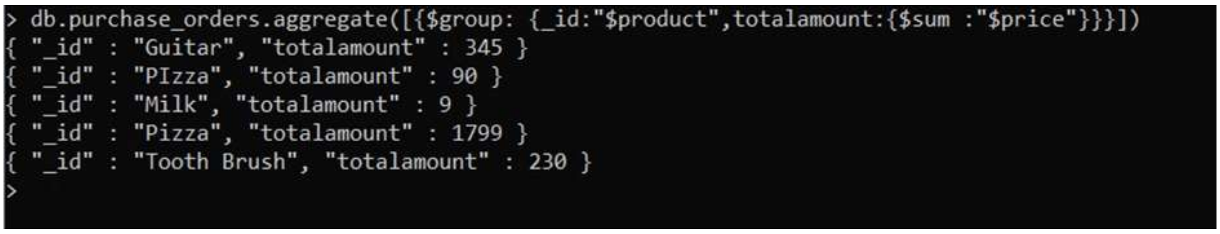


```
> db.purchase_order.distinct("product")
[ "Toothbrush", "guitar", "milk", "pizza" ]
>
```

3. **find the total amount of money spent by each customer. Query: -**

db.purchase_order.aggregate([{$group :{_id:"$customername",total:{$sum:"$price"}}}])

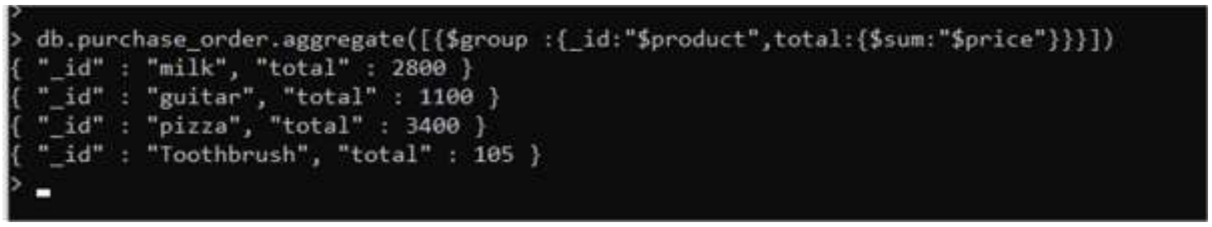
Output :-



```
> db.purchase_orders.aggregate([{$group: {_id:"$product",totalamount:{$sum :"$price"}}}])
{ "_id" : "Guitar", "totalamount" : 345 }
{ "_id" : "PIzza", "totalamount" : 90 }
{ "_id" : "Milk", "totalamount" : 9 }
{ "_id" : "Pizza", "totalamount" : 1799 }
{ "_id" : "Tooth Brush", "totalamount" : 230 }
>
```

4. **find the total amount of money spent on each product. Query: -**

db.purchase_order.aggregate([{$group :{_id:"$product",total:{$sum:"$price"}}}]) Output :-

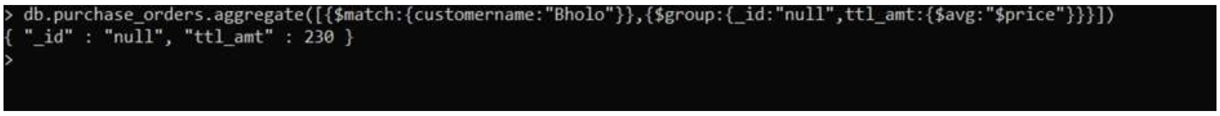


```
>
> db.purchase_order.aggregate([{$group :{_id:"$product",total:{$sum:"$price"}}}])
{ "_id" : "milk", "total" : 2800 }
{ "_id" : "guitar", "total" : 1100 }
{ "_id" : "pizza", "total" : 3400 }
{ "_id" : "Toothbrush", "total" : 105 }
> _
```

5. **find how much money each customer has spent on toothbrushes or pizza. Query: -**

db.purchase_order.aggregate([{$match:{$or:[{"product":"pizza"},{"pr oduct":"Toothbrush"}] }},{$group:{_id:"$customername",spent:{$sum:"$price"}}}])

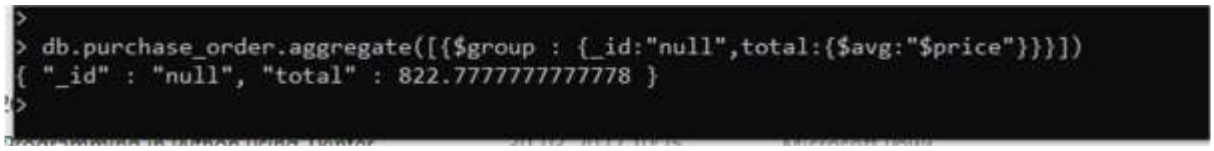
Output :-



```
> db.purchase_orders.aggregate([{$match:{customername:"Bholo"}},{$group:{_id:"null",ttl_amt:{$avg:"$price"}}}])
{ "_id" : "null", "ttl_amt" : 230 }
>
```

6. **calc. the avg purchase price of ABC. Query: -**

db.purchase_order.aggregate([{$group : {_id:"null",total:{$avg:"$price"}}}]) Output :-



```
>
> db.purchase_order.aggregate([{$group : {_id:"null",total:{$avg:"$price"}}}])
{ "_id" : "null", "total" : 822.7777777777778 }
>
```

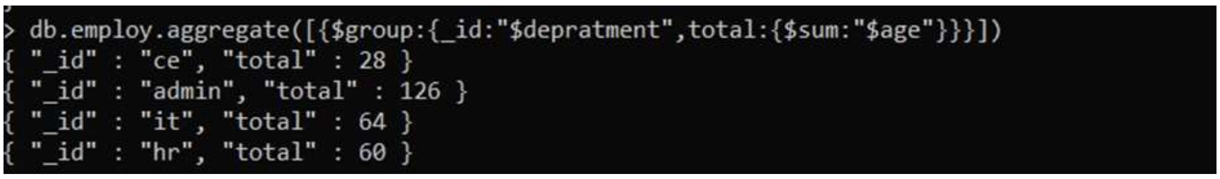
 EX-3:

create a collection called employee having fields name, department(Admin,ce,it,hr) , age , total_exp, languages(diff languages) insert 8 records into collections

 Queries:-

1) **find the total age of employees for each department.**

Query:- db.emp.aggregate([{$group : {_id:"$department",age_sum:{$sum:"$age"}}}]) Output :-

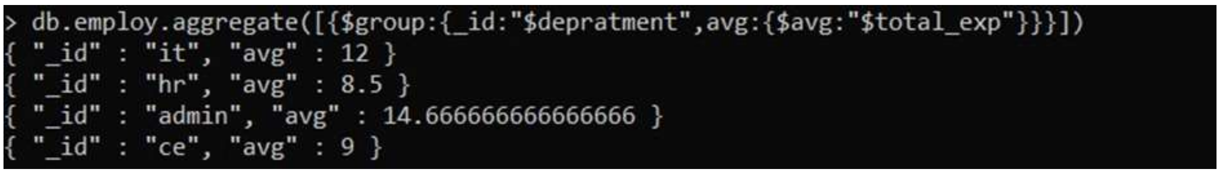


```
> db.employ.aggregate([{$group:{_id:"$depratment",total:{$sum:"$age"}}}])
{ "_id" : "ce", "total" : 28 }
{ "_id" : "admin", "total" : 126 }
{ "_id" : "it", "total" : 64 }
{ "_id" : "hr", "total" : 60 }
```

2) **calc. the avg experience of each department. Query:-**

db.emp.aggregate([{$group :

{_id:"$department",avgexp:{$avg:"$exp"}}}]) Output :-

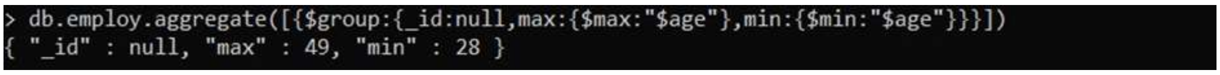


```
> db.employ.aggregate([{$group:{_id:"$depratment",avg:{$avg:"$total_exp"}}}])
{ "_id" : "it", "avg" : 12 }
{ "_id" : "hr", "avg" : 8.5 }
{ "_id" : "admin", "avg" : 14.666666666666666 }
{ "_id" : "ce", "avg" : 9 }
```

3) **find the youngest and oldest employee. Query:-**

db.emp.aggregate([{$group:{_id:null,youngest_emp:{$min:"$age"},oldest_emp:{$max:"$age"}}}])

Output :-



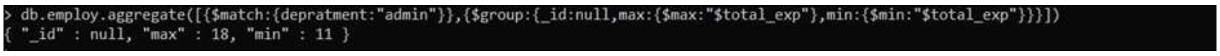
```
> db.employ.aggregate([{$group:{_id:null,max:{$max:"$age"},min:{$min:"$age"}}}])
{ "_id" : null, "max" : 49, "min" : 28 }
```

4) **find the minimum and maximum experienced employee from department admin.**

Query:-

db.emp.aggregate([{$match:{"department":"admin"}},{$group:{_id:null,min_total_exp:{$min:"$total_exp"},max_total_exp:{$max:"$total_exp"}}}])

Output :-



```
> db.employ.aggregate([{$match:{depratment:"admin"}},{$group:{_id:null,max:{$max:"$total_exp"},min:{$min:"$total_exp"}}}])
{ "_id" : null, "max" : 18, "min" : 11 }
```

3. Map-Reduce :-

- **Ex-1:**

create a collection called car having fields car_id, name, color, car_number, mfd_country, speed and price insert 8 records

create a map function that will get data of cars having

speeds greater than 70 create a reduce function that will

find the average speed code :- var map1=function(){

if(this.speed>70){ emit(this.car_id,this.speed);

} }; var

reduce=function(car_id,sp

eed){ var

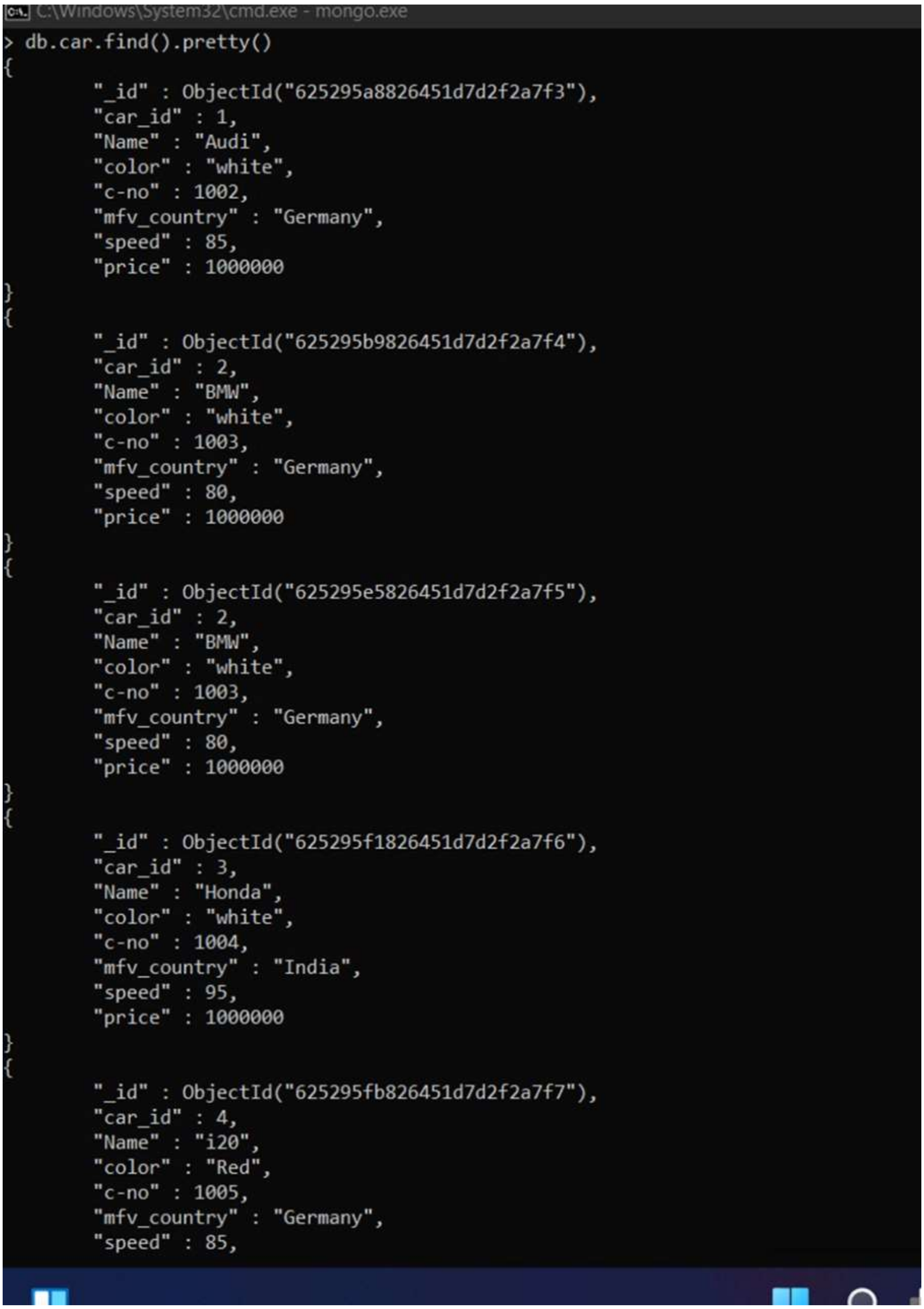
a=Array.avg(speed);

return a;

};
Query :-

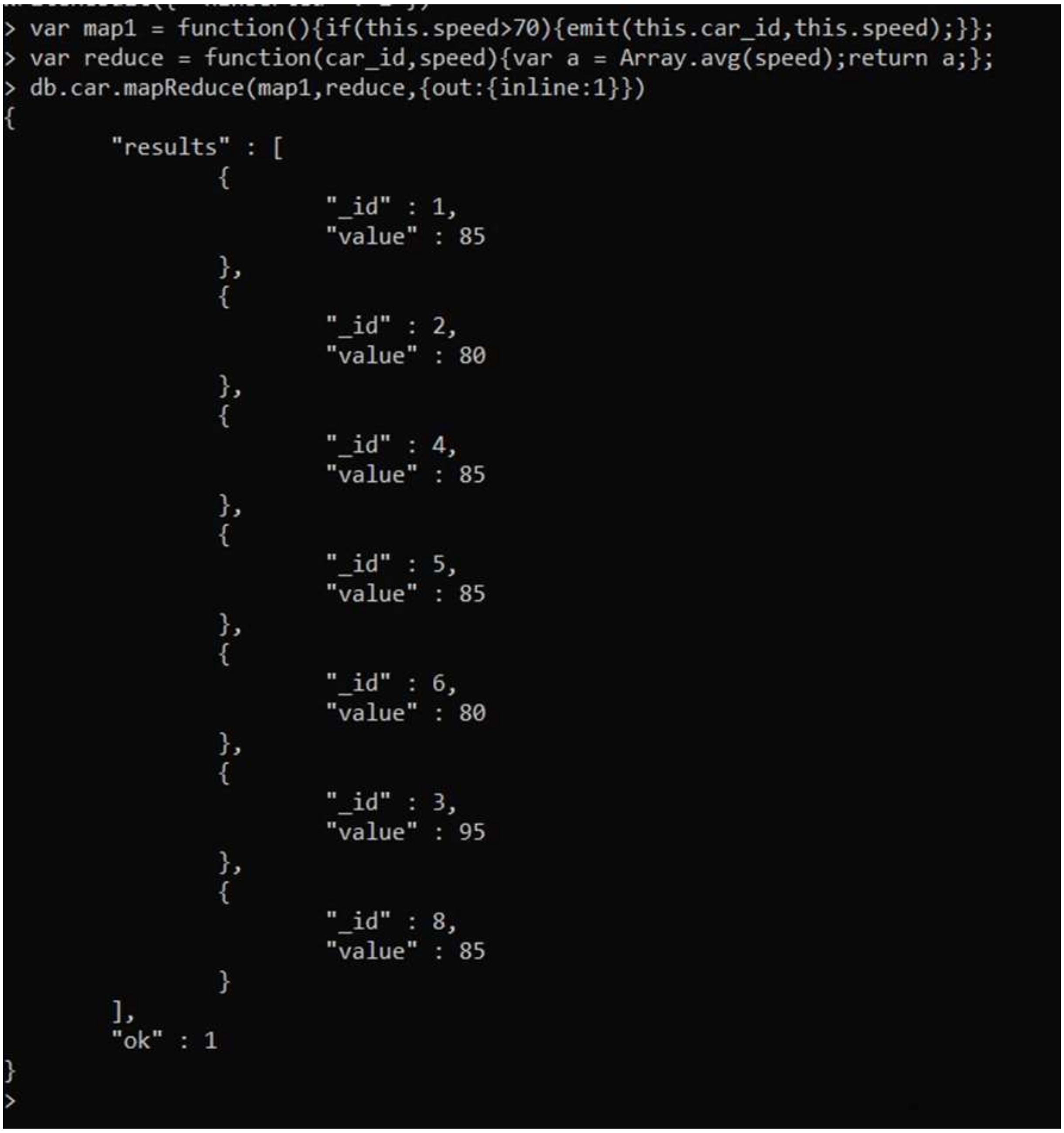
db.car.mapReduce(map,reduce1,{out:{inline

:1}}) Output :-



C:\Windows\System32\cmd.exe - mongo.exe

```
> db.car.find().pretty()
{
        "_id" : ObjectId("625295a8826451d7d2f2a7f3"),
        "car_id" : 1,
        "Name" : "Audi",
        "color" : "white",
        "c-no" : 1002,
        "mfv_country" : "Germany",
        "speed" : 85,
        "price" : 1000000
}
{
        "_id" : ObjectId("625295b9826451d7d2f2a7f4"),
        "car_id" : 2,
        "Name" : "BMW",
        "color" : "white",
        "c-no" : 1003,
        "mfv_country" : "Germany",
        "speed" : 80,
        "price" : 1000000
}
{
        "_id" : ObjectId("625295e5826451d7d2f2a7f5"),
        "car_id" : 2,
        "Name" : "BMW",
        "color" : "white",
        "c-no" : 1003,
        "mfv_country" : "Germany",
        "speed" : 80,
        "price" : 1000000
}
{
        "_id" : ObjectId("625295f1826451d7d2f2a7f6"),
        "car_id" : 3,
        "Name" : "Honda",
        "color" : "white",
        "c-no" : 1004,
        "mfv_country" : "India",
        "speed" : 95,
        "price" : 1000000
}
{
        "_id" : ObjectId("625295fb826451d7d2f2a7f7"),
        "car_id" : 4,
        "Name" : "i20",
        "color" : "Red",
        "c-no" : 1005,
        "mfv_country" : "Germany",
        "speed" : 85,
```



```
> var map1 = function(){if(this.speed>70){emit(this.car_id,this.speed);}};
> var reduce = function(car_id,speed){var a = Array.avg(speed);return a;};
> db.car.mapReduce(map1,reduce,{out:{inline:1}})
{
        "results" : [
                {
                        "_id" : 1,
                        "value" : 85
                },
                {
                        "_id" : 2,
                        "value" : 80
                },
                {
                        "_id" : 4,
                        "value" : 85
                },
                {
                        "_id" : 5,
                        "value" : 85
                },
                {
                        "_id" : 6,
                        "value" : 80
                },
                {
                        "_id" : 3,
                        "value" : 95
                },
                {
                        "_id" : 8,
                        "value" : 85
                }
        ],
        "ok" : 1
}
>
```

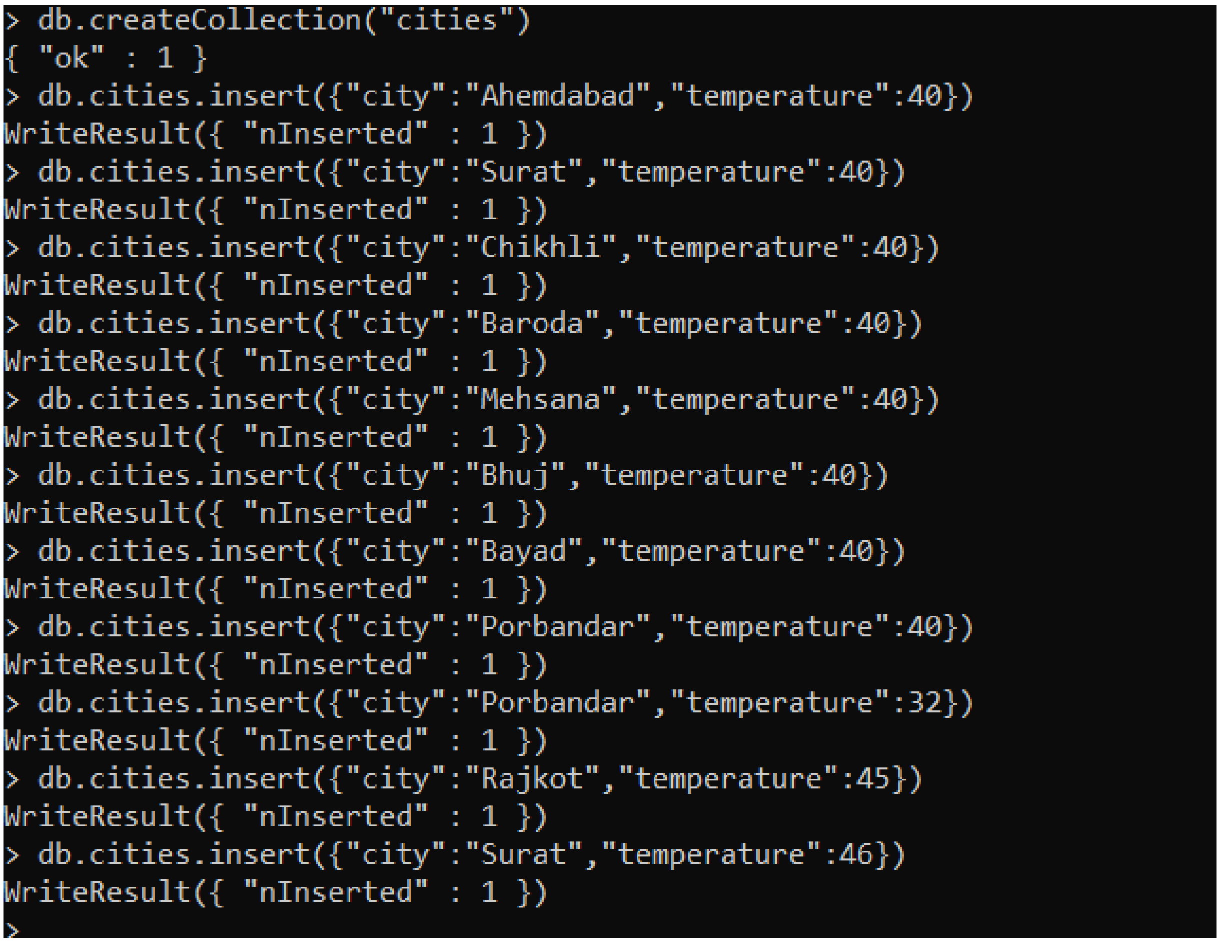
- **Ex-2:**

create a collection called city having two fields city(Ahemdabad, Mehsana,

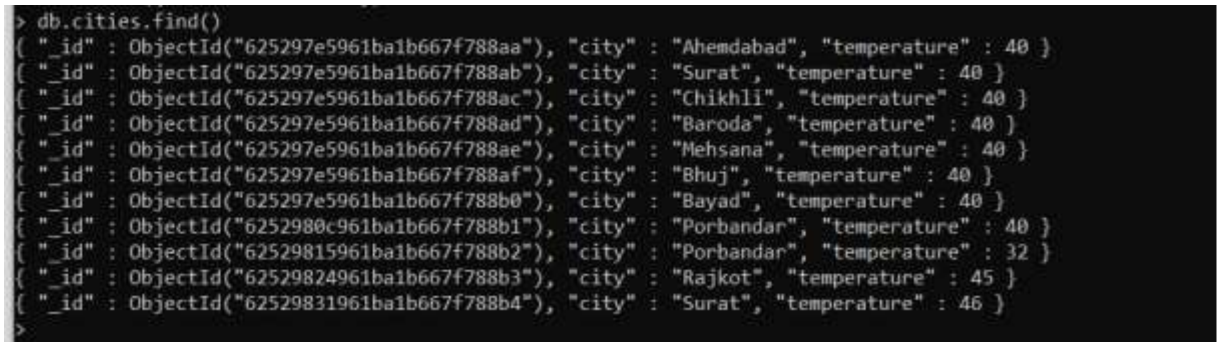
Baroda) and Temperature.

insert 8 records

Output :-



```
> db.createCollection("cities")
{ "ok" : 1 }
> db.cities.insert({"city":"Ahemdabad","temperature":40})
WriteResult({ "nInserted" : 1 })
> db.cities.insert({"city":"Surat","temperature":40})
WriteResult({ "nInserted" : 1 })
> db.cities.insert({"city":"Chikhli","temperature":40})
WriteResult({ "nInserted" : 1 })
> db.cities.insert({"city":"Baroda","temperature":40})
WriteResult({ "nInserted" : 1 })
> db.cities.insert({"city":"Mehsana","temperature":40})
WriteResult({ "nInserted" : 1 })
> db.cities.insert({"city":"Bhuj","temperature":40})
WriteResult({ "nInserted" : 1 })
> db.cities.insert({"city":"Bayad","temperature":40})
WriteResult({ "nInserted" : 1 })
> db.cities.insert({"city":"Porbandar","temperature":40})
WriteResult({ "nInserted" : 1 })
> db.cities.insert({"city":"Porbandar","temperature":32})
WriteResult({ "nInserted" : 1 })
> db.cities.insert({"city":"Rajkot","temperature":45})
WriteResult({ "nInserted" : 1 })
> db.cities.insert({"city":"Surat","temperature":46})
WriteResult({ "nInserted" : 1 })
>
```



```
> db.cities.find()
{ "_id" : ObjectId("625297e5961ba1b667f788aa"), "city" : "Ahemdabad", "temperature" : 40 }
{ "_id" : ObjectId("625297e5961ba1b667f788ab"), "city" : "Surat", "temperature" : 40 }
{ "_id" : ObjectId("625297e5961ba1b667f788ac"), "city" : "Chikhli", "temperature" : 40 }
{ "_id" : ObjectId("625297e5961ba1b667f788ad"), "city" : "Baroda", "temperature" : 40 }
{ "_id" : ObjectId("625297e5961ba1b667f788ae"), "city" : "Mehsana", "temperature" : 40 }
{ "_id" : ObjectId("625297e5961ba1b667f788af"), "city" : "Bhuj", "temperature" : 40 }
{ "_id" : ObjectId("625297e5961ba1b667f788b0"), "city" : "Bayad", "temperature" : 40 }
{ "_id" : ObjectId("6252980c961ba1b667f788b1"), "city" : "Porbandar", "temperature" : 40 }
{ "_id" : ObjectId("62529815961ba1b667f788b2"), "city" : "Porbandar", "temperature" : 32 }
{ "_id" : ObjectId("62529824961ba1b667f788b3"), "city" : "Rajkot", "temperature" : 45 }
{ "_id" : ObjectId("62529831961ba1b667f788b4"), "city" : "Surat", "temperature" : 46 }
>
```

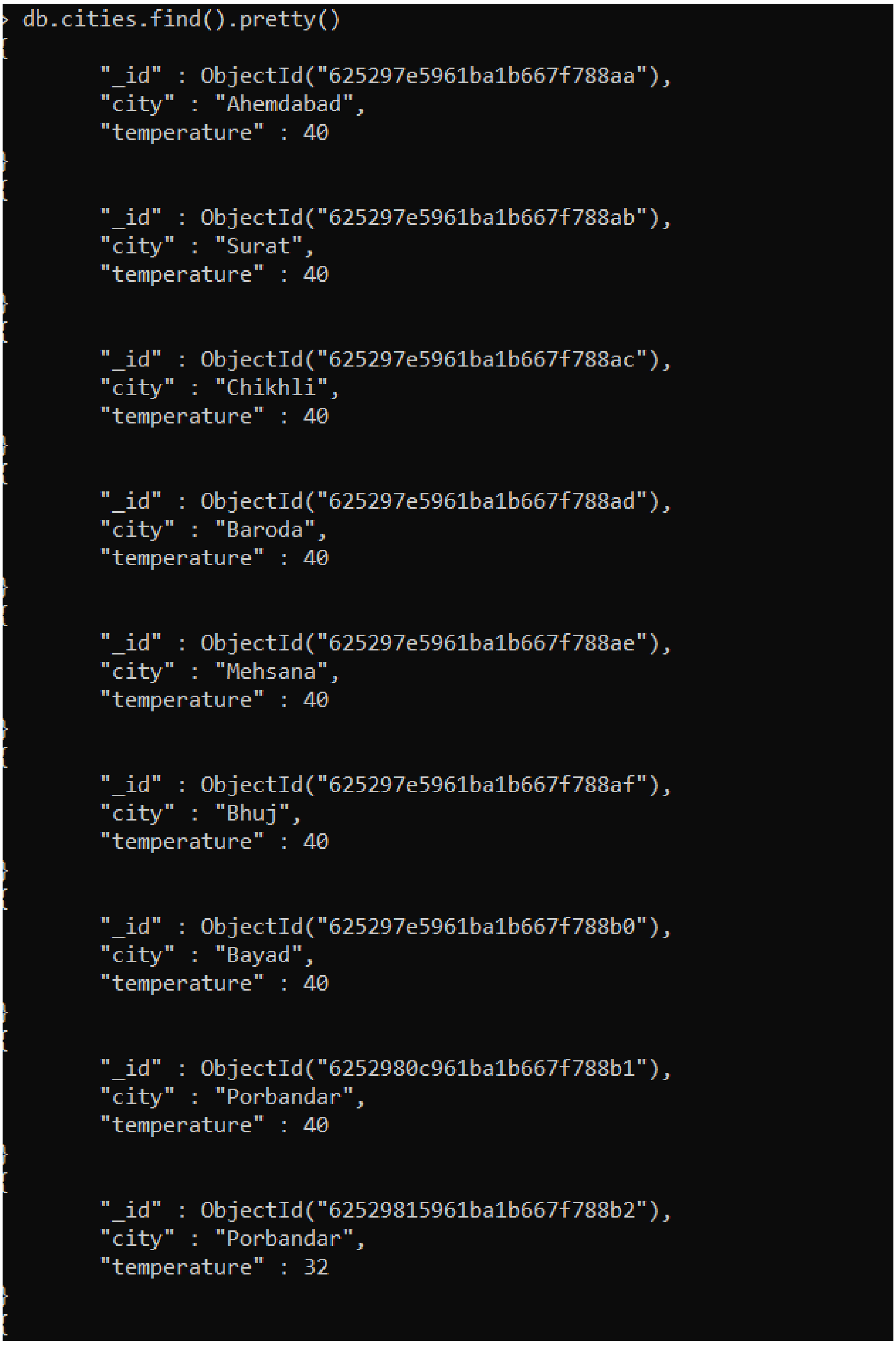
create a map and reduce function to find maximum temperature for each city.

```
var map=function(){

emit(this.city,this.tempereture)

}; var maxt=function(city,

tempereture){  var max=

tempereture [0];  for(var

i=0;i<

tempereture.length;i++){

if(tempereture [i]>max){

max= tempereture [i];

 }  return

max;

 } }; var mint=function(city,

tempereture){


 var min= tempereture [0];

for(var i=0;i<

tempereture.length;i++){

if(tempereture [i]<min){

min= tempereture [i];

 }  return

min;

 }

};
```

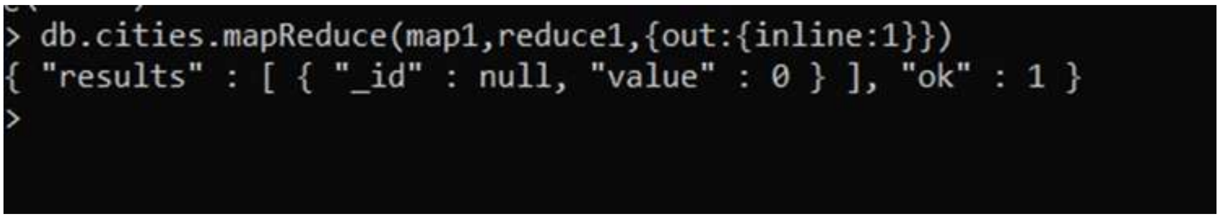
Query:

```
db.city.mapReduce(map,mint,{out:{inline:1}})
```

Output :-



```
> db.cities.find().pretty()
{
        "_id" : ObjectId("625297e5961ba1b667f788aa"),
        "city" : "Ahemdabad",
        "temperature" : 40
}
{
        "_id" : ObjectId("625297e5961ba1b667f788ab"),
        "city" : "Surat",
        "temperature" : 40
}
{
        "_id" : ObjectId("625297e5961ba1b667f788ac"),
        "city" : "Chikhli",
        "temperature" : 40
}
{
        "_id" : ObjectId("625297e5961ba1b667f788ad"),
        "city" : "Baroda",
        "temperature" : 40
}
{
        "_id" : ObjectId("625297e5961ba1b667f788ae"),
        "city" : "Mehsana",
        "temperature" : 40
}
{
        "_id" : ObjectId("625297e5961ba1b667f788af"),
        "city" : "Bhuj",
        "temperature" : 40
}
{
        "_id" : ObjectId("625297e5961ba1b667f788b0"),
        "city" : "Bayad",
        "temperature" : 40
}
{
        "_id" : ObjectId("6252980c961ba1b667f788b1"),
        "city" : "Porbandar",
        "temperature" : 40
}
{
        "_id" : ObjectId("62529815961ba1b667f788b2"),
        "city" : "Porbandar",
        "temperature" : 32
}
{
```

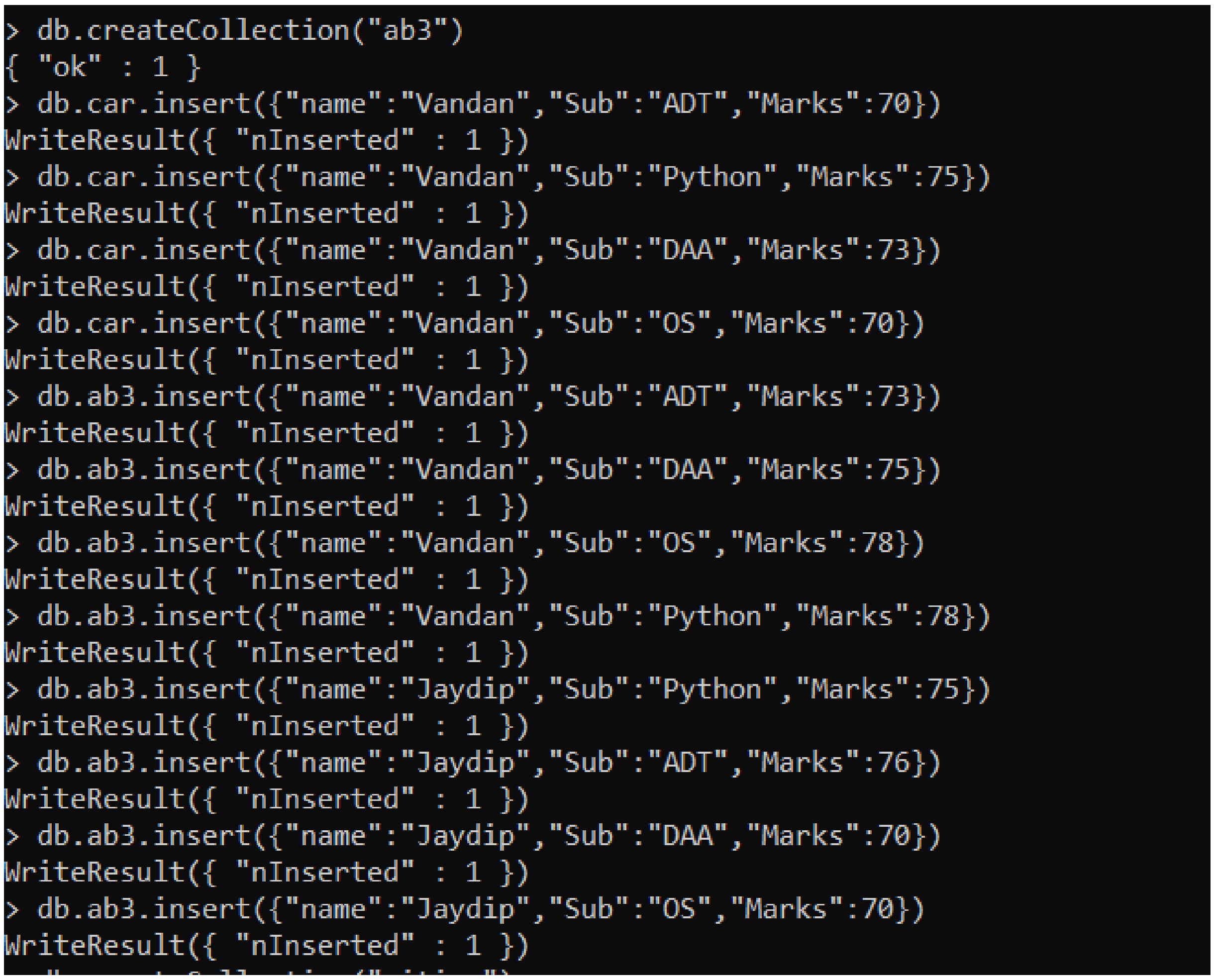


```
> db.cities.mapReduce(map1,reduce1,{out:{inline:1}})
{ "results" : [ { "_id" : null, "value" : 0 } ], "ok" : 1 }
>
```

 Ex-3 : create a collection called AB3 having fields

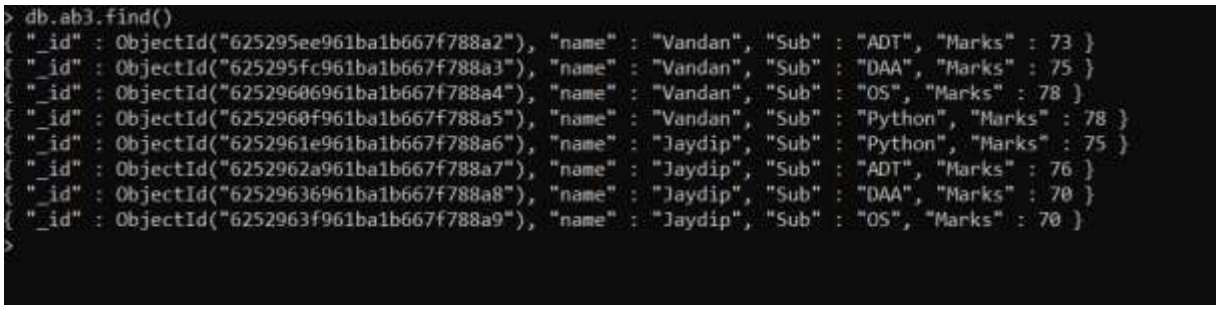
student_names,subject and marks.

insert 8 records



```
> db.createCollection("ab3")
{ "ok" : 1 }
> db.car.insert({"name":"Vandan","Sub":"ADT","Marks":70})
WriteResult({ "nInserted" : 1 })
> db.car.insert({"name":"Vandan","Sub":"Python","Marks":75})
WriteResult({ "nInserted" : 1 })
> db.car.insert({"name":"Vandan","Sub":"DAA","Marks":73})
WriteResult({ "nInserted" : 1 })
> db.car.insert({"name":"Vandan","Sub":"OS","Marks":70})
WriteResult({ "nInserted" : 1 })
> db.ab3.insert({"name":"Vandan","Sub":"ADT","Marks":73})
WriteResult({ "nInserted" : 1 })
> db.ab3.insert({"name":"Vandan","Sub":"DAA","Marks":75})
WriteResult({ "nInserted" : 1 })
> db.ab3.insert({"name":"Vandan","Sub":"OS","Marks":78})
WriteResult({ "nInserted" : 1 })
> db.ab3.insert({"name":"Vandan","Sub":"Python","Marks":78})
WriteResult({ "nInserted" : 1 })
> db.ab3.insert({"name":"Jaydip","Sub":"Python","Marks":75})
WriteResult({ "nInserted" : 1 })
> db.ab3.insert({"name":"Jaydip","Sub":"ADT","Marks":76})
WriteResult({ "nInserted" : 1 })
> db.ab3.insert({"name":"Jaydip","Sub":"DAA","Marks":70})
WriteResult({ "nInserted" : 1 })
> db.ab3.insert({"name":"Jaydip","Sub":"OS","Marks":70})
WriteResult({ "nInserted" : 1 })
```

Output :



```
> db.ab3.find()
{ "_id" : ObjectId("625295ee961ba1b667f788a2"), "name" : "Vandan", "Sub" : "ADT", "Marks" : 73 }
{ "_id" : ObjectId("625295fc961ba1b667f788a3"), "name" : "Vandan", "Sub" : "DAA", "Marks" : 75 }
{ "_id" : ObjectId("62529606961ba1b667f788a4"), "name" : "Vandan", "Sub" : "OS", "Marks" : 78 }
{ "_id" : ObjectId("6252960f961ba1b667f788a5"), "name" : "Vandan", "Sub" : "Python", "Marks" : 78 }
{ "_id" : ObjectId("6252961e961ba1b667f788a6"), "name" : "Jaydip", "Sub" : "Python", "Marks" : 75 }
{ "_id" : ObjectId("6252962a961ba1b667f788a7"), "name" : "Jaydip", "Sub" : "ADT", "Marks" : 76 }
{ "_id" : ObjectId("62529636961ba1b667f788a8"), "name" : "Jaydip", "Sub" : "DAA", "Marks" : 70 }
{ "_id" : ObjectId("6252963f961ba1b667f788a9"), "name" : "Jaydip", "Sub" : "OS", "Marks" : 70 }
>
```

create map and reduce function to get total marks for each

student and output should be written in collection name total.

db.ab3.find()

db.createCollection("H

ello")

var map=function(){

emit(this.name,this.mar

ks)

};

var red=function(name,marks){
var sum=0; for(var

i=0;i<marks.length;i++){

sum=sum+marks[i];

} return

sum;

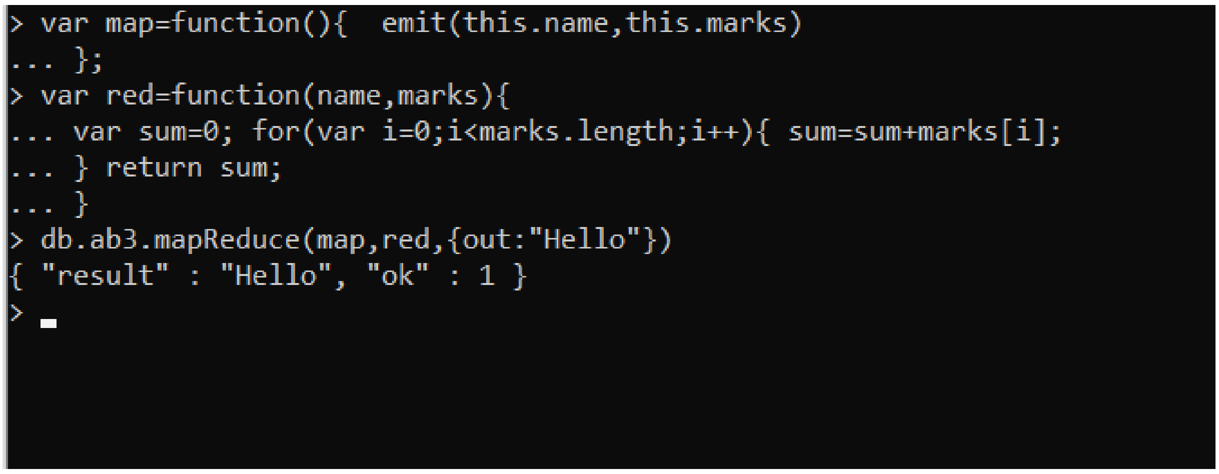
}

Query:

db.ab3.mapReduce(map,red,{out:"Hello"})

db.Hello.find()

Output :-



```
> var map=function(){  emit(this.name,this.marks)
... };
> var red=function(name,marks){
... var sum=0; for(var i=0;i<marks.length;i++){ sum=sum+marks[i];
... } return sum;
... }
> db.ab3.mapReduce(map,red,{out:"Hello"})
{ "result" : "Hello", "ok" : 1 }
> 
```