**Group-B**

**Assignment no – 04**

**PROBLEM STATEMENT**:-Implement map reduce operation with suitable example using monoDB.

db.createCollection("mr1")

{ "ok" : 1 }

> db.mr1.insert([{prod\_id:"a1",price:400,status:"a"},{prod\_id:"b1",price:300,status:"d"},{prod\_id:"a1",price:200,status:"c"},{prod\_id:"c1",price:200,status:"c"},{prod\_id:"b1",price:700,status:"a"},{prod\_id:"b1",price:800,status:"a"},{prod\_id:"c1",price:200,status:"c"}])

> db.mr1.find().pretty()

{

"\_id" : ObjectId("59cdbb0468708196535e81e7"),

"prod\_id" : "a1",

"price" : 400,

"status" : "a"

}

{

"\_id" : ObjectId("59cdbc4068708196535e81e8"),

"prod\_id" : "b1",

"price" : 300,

"status" : "d"

}

{

"\_id" : ObjectId("59cdbc4068708196535e81e9"),

"prod\_id" : "a1",

"price" : 200,

"status" : "c"

}

{

"\_id" : ObjectId("59cdbc4068708196535e81ea"),

"prod\_id" : "c1",

"price" : 200,

"status" : "c"

}

{

"\_id" : ObjectId("59cdbc4068708196535e81eb"),

"prod\_id" : "b1",

"price" : 700,

"status" : "a"

}

{

"\_id" : ObjectId("59cdbc4068708196535e81ec"),

"prod\_id" : "b1",

"price" : 800,

"status" : "a"

}

{

"\_id" : ObjectId("59cdbc4068708196535e81ed"),

"prod\_id" : "c1",

"price" : 200,

"status" : "c"

}

1. Find the sum of price of each product whose status is A.

> db.mr1.mapReduce(function(){emit(this.prod\_id,this.price)},function(key,values){return Array.sum(values)},{query:{status:"a"},out:"total\_price"}).find().pretty()

{ "\_id" : "a1", "value" : 400 }

{ "\_id" : "b1", "value" : 1500 }

2. Find the average price of each product.

> db.mr1.mapReduce(function(){emit(this.prod\_id,this.price)},function(key,values){return Array.avg(values)},{query:{status:"a"},out:"mr\_avg"}).find().pretty()

{ "\_id" : "a1", "value" : 400 }

{ "\_id" : "b1", "value" : 750 }

3. Find the min price of each product whose status is A.

> db.mr1.mapReduce(function(){emit(this.prod\_id,this.price)},function(key,values){return Math.min.apply(Math,values)},{query:{status:"a"},out:"mr\_min"}).find().pretty()

{ "\_id" : "a1", "value" : 400 }

{ "\_id" : "b1", "value" : 700 }

4. Find the max price of each product whose status is A.

> db.mr1.mapReduce(function(){emit(this.prod\_id,this.price)},function(key,values){return Math.max.apply(Math,values)},{query:{status:"a"},out:"mr\_max"}).find().pretty()

{ "\_id" : "a1", "value" : 400 }

{ "\_id" : "b1", "value" : 800 }

5. Find the max price of each product.

> db.mr1.mapReduce(function(){emit(this.prod\_id,this.price)},function(key,values){return Math.max.apply(Math,values)},{out:"mr\_max"}).find().pretty()

{ "\_id" : "a1", "value" : 400 }

{ "\_id" : "b1", "value" : 800 }

{ "\_id" : "c1", "value" : 200 }

6. Find the min price of each product

> db.mr1.mapReduce(function(){emit(this.prod\_id,this.price)},function(key,values){return Math.min.apply(Math,values)},{out:"mr\_min"}).find().pretty()

{ "\_id" : "a1", "value" : 200 }

{ "\_id" : "b1", "value" : 300 }

{ "\_id" : "c1", "value" : 200 }

7. Find the avg price of each product.

> db.mr1.mapReduce(function(){emit(this.prod\_id,this.price)},function(key,values){return Array.avg(values)},{out:"mr\_avg"}).find().pretty()

{ "\_id" : "a1", "value" : 300 }

{ "\_id" : "b1", "value" : 600 }

{ "\_id" : "c1", "value" : 200 }