**QUERIES**

**1. Demonstrate the usage of $match, $group, aggregate pipelines**.

>db.employee.aggregate([{$match:{"years\_of\_service":3}},{$group:{\_id:"$years\_of\_service",total:{$sum:"$salary"}}}]);

: {"\_id":3,"total":1115592}

**2. Demonstrate the Map-Reduce aggregate function on this dataset.**

> var mapFunction=function(){emit(this.no\_of\_increments,this.salary);};

> var reduceFunction=function(increments,salaries){return Array.sum(salaries);};

> db.employee.mapReduce(mapFunction,reduceFunction,{out:"output1"})

{ "result" : "output1", "ok" : 1 }

> db.output1.find().pretty()

{ "\_id" : 2, "value" : 2363764 }

{ "\_id" : 3, "value" : 1644801 }

{ "\_id" : 5, "value" : 1034267 }

{ "\_id" : 4, "value" : 3778908 }

{ "\_id" : 1, "value" : 1461265 }

{ "\_id" : 0, "value" : 216786 }

**3.Find out all the Employees, who are paying tax>50000 and are not eligible for pay raise.**

> db.employee.find({$and:[{tax:{$gt:50000}},{eligible\_for\_pay\_raise:false}]}).pretty();

{

"\_id" : ObjectId("60efea9730fc66fdd0e606b2"),

"name" : "Sneha",

"years\_of\_service" : 3,

"no\_of\_increments" : 1,

"eligible\_for\_pay\_raise" : false,

"salary" : 209856,

"tax" : 80000

}

{

"\_id" : ObjectId("60efea9730fc66fdd0e606bc"),

"name" : "Varun",

"years\_of\_service" : 5,

"no\_of\_increments" : 2,

"eligible\_for\_pay\_raise" : false,

"salary" : 409876,

"tax" : 57000

}

**4. Demonstrate the Alter and Drop commands on this dataset**

> db.employee.updateOne({name:"Rakshitha"},{$set:{salary:1008766},});

{ "acknowledged" : true, "matchedCount" : 1, "modifiedCount" : 1 }

> db.employee.find({name:"Rakshitha"}).pretty();

{

"\_id" : ObjectId("60efea9730fc66fdd0e606a9"),

"name" : "Rakshitha",

"years\_of\_service" : 8,

"no\_of\_increments" : 4,

"eligible\_for\_pay\_raise" : true,

"salary" : 1008766,

"tax" : 100000

}

> db.employee.drop()

true

>