

Group B
Assignment Number 1
MongoDB

Problem Statement:

Study of Open Source NOSQL Database: MongoDB (Installation, Basic CRUD operations, Execution)

PROGRAM

Basic CRUD Operation

- Creating & Inserting Values in new Database

```
> show dbs
admin    0.000GB
config   0.000GB
local    0.000GB
```

```
> use demodb
switched to db demodb
```

- Showing Database

```
> show dbs
admin    0.000GB
config   0.000GB
demodb   0.000GB
local    0.000GB
```

- Drop Database

```
> use demodb
switched to db demodb
```

```
> db.dropDatabase()
{"dropped": "demodb", "ok": 1}
```

```
> show dbs
admin    0.000GB
config   0.000GB
local    0.000GB
```

- Create Collection MongoDB can create collections automatically also.

```
> db.createCollection("newCollection")
{"ok": 1}
```

```
> show collections newCollection
```

- Drop Collection

```
> db.newCollection.drop()
True
```

```
> show collections
<
```

- Inserting Document into Collection

```
db.student.insert({"name": "Sudarshan"})
> WriteResult({"nInserted": 1})
> db.student.insert({"name": "Mandar"})
WriteResult({"nInserted": 1})
```

- Retriving Document from Collection

```
> db.student.find()
{"_id": ObjectId("60102acc3e5b451a6eeb4265"), "name": "Sudarshan"}
{"_id": ObjectId("60102ad53e5b451a6eeb4266"), "name": "Mandar"}

// Finding by name
> db.student.find({'name': 'Sudarshan'})
{"_id": ObjectId("6010291c3e5b451a6eeb4264"), "name": "Sudarshan"}
```

- Updating Document:

```
> db.student.update({'name': 'Mandar'}, {$set: {'name': 'Swaraj'}})
WriteResult({"nMatched": 1, "nUpserted": 8, "nModified": 1})
> db.student.find()
{"_id": ObjectId("60102acc3e5b451a6eeb4265"), "name": "Sudarshan"}
{"_id": ObjectId("60102ad53e5b451a6eeb4266"), "name": "Swaraj"}
```

- Delete Document:

```
> db.student.remove({'name': 'Swaraj'})
WriteResult({"nRemoved": 1})
> db.student.find()
{"_id": ObjectId("60102acc3e5b451a6eeb4265"), "name": "Sudarshan"}
```

Group B
Assignment Number 2
MongoDB

Problem Statement:

Design and Develop MongoDB Queries using CRUD operations. (Use CRUDOperations, SAVE method, logical operators)

PROGRAM

> use
Sudarshan;

```
switched to db Sudarshan
> db.createCollection('Student');
{ "ok" : 1 }
> db.Student.insert({'Rno':"1", "Name": "Piyush", "Class": "TECOMP"});
WriteResult({ "nInserted" : 1 })

> db.Student.insert({'Rno':"2", "Name": "mohit", "Class": "TECOMP"});
WriteResult({ "nInserted" : 1 })

> db.Student.insert({'Rno':"3", "Name": "Ashley", "Class": "TECOMP"});
WriteResult({ "nInserted" : 1 })

> db.Student.insert({'Rno':"4", "Name": "Htresh", "Class": "TECOMP"});
WriteResult({ "nInserted" : 1 })

> db.Student.insert({'Rno':"5", "Name": "Pratik", "Class": "TECOMP"});
WriteResult({ "nInserted" : 1 })

> db.Student.insert({'Rno':"6", "Name": "Pratik", "Class": "TECOMP"});
WriteResult({ "nInserted" : 1 })

> db.Student.find();
{'_id': ObjectId("5ba1d618f5bbacd4ad81568d"), "Rno": "1",
"Name": "Piyush", "Class": "TECOMP"}
{'_id': ObjectId("5ba1d625f5bbacd4ad81568e"), "Rno": "2",
"Name": "mohit", "Class": "TECOMP"}
{'_id': ObjectId("5ba1d63af5bbacd4ad81568f"), "Rno": "3",
"Name": "Ashley", "Class": "TECOMP"}
{'_id': ObjectId("5ba1d647f5bbacd4ad815690"), "Rno": "4",
"Name": "Htresh", "Class": "TECOMP"}
{'_id': ObjectId("5ba1d65ef5bbacd4ad815691"), "Rno": "5",
"Name": "Pratik", "Class": "TECOMP"} {'_id':
ObjectId("5ba1d66df5bbacd4ad815692"), "Rno": "6",
"Name": "Pratik", "Class": "TECOMP"}
> db.Student.find().pretty();
```

```

{
  "id": ObjectId("5ba1d618f5bbacd4ad81568d"),
  "Rno": "1",
  "Name": "Piyush",
  "Class": "TECOMP"
}
{
  "id": ObjectId("5ba1d625f5bbacd4ad81568e"),
  "Rno": "2", "Name": "mohit",
  "Class": "TECOMP"
}
{
  "_id": ObjectId("5ba1d63af5bbacd4ad81568f"),
  "Rno": "3",
  "Name": "Ashley",
  "Class": "TECOMP"
}
{
  "_id": ObjectId("5ba1d647f5bbacd4ad815690"),
  "Rno": "4",
  "Name": "Htesh",
  "Class": "TECOMP"
}
{
  "id": ObjectId("5ba1d65ef5bbacd4ad815691"),
  "Rno": "5",
  "Name": "Pratik",
  "Class": "TECOMP"
}
{
  "id": ObjectId("5ba1d66df5bbacd4ad815692"),
  "Rno": "6",
  "Name": "Pratik",
  "Class": "TECOMP"
}

> db.Student.update({Name: 'Htesh'}, {$set:
{Name: 'Henry'}});
WriteResult({ "nMatched": 1, "nUpserted": 8, "nModified": 1 })

> db.Student.find().pretty(); { "_id": ObjectId("5b8fad4ef80832a0a58b5036"),
  "Rno": "1",
  "Name": "Piyush",
  "Class": "TECOMP"
}
{
  "id": ObjectId("5b8fad62fe8832aea5eb5037"),
  "Rno": "2",
  "Name": "mohit",
  "Class": "TECOMP"
}

```

```

{
  "id": ObjectId("5b8fad70fe0832a0a5065038"),
  "Rno": "3",
  "Name": "Ashley",
  "Class": "TECOMP"
}
{
  "id": ObjectId("5b8fad7ff00832a0a5055039"),
  "Rno": "4",
  "Name": "Henry",
  "Class": "TECOMP"
}
{
  "id": ObjectId("5b8fad8df00832a0a506503a"),
  "Rno": "5",
  "Name": "Pratik",
  "Class": "TECOMP"
}
{
  "id": ObjectId("5b8fada4f00832a0a58b583b"),
  "Rno": "6",
  "Name": "Pratik",
  "Class": "TECOMP"
}
{
  > db.Student.remove({"ADD":"MP"});
  WriteResult({ "nRemoved": 1 })
  > db.Student.find().pretty();

  "_id": ObjectId("5b8fad62f08832a0a5065837"),
  "Rno": "2",
  "Name": "mohit",
  "Class": "TECOMP"
}
{
  "_id": ObjectId("5b8fad70f08832a0a58b5038"),
  "Rno": "3",
  "Name": "Ashley",
  "Class": "TECOMP"
}
{
  "id": ObjectId("5b8fad7ff0e832a0a50b5039"),
  "Rno": "4",
  "Name": "Henry",
  "Class": "TECOMP"
}
{
  "_id": ObjectId("5b8fad8df00832a0a506503a"),
  "Rno": "5",
  "Name": "Pratik",
  "Class": "TECOMP"
}

```

```

{
  "_id": ObjectId("5b8fada4f00832a0a58b503b"),
  "Rno": "6",
  "Name": "Pratik",
  "Class": "TECOMP"
}

>db.Student.save({_id:ObjectId("5b8fad4ef00832a8a58b5036"), "RNO ":"1","NAME":"PIYUSH", "CLASS":"TECOMP","ADD":"MP"});
WriteResult({ "nMatched": 1, "nUpserted": 0, "nModified": 1 })

> db.Student.find().pretty();
{
  "_id": ObjectId("5b8fad4efe8832a0a5065036"),
  "RNO": "1",
  "NAME": "PIYUSH",
  "CLASS": "TECOMP",
  "ADD": "MP"
}
{
  "_id": ObjectId("5b8fad62f08832a0a58b5037"),
  "Rno": "2",
  "Name": "mohit",
  "Class": "TECOMP"
}

{
  "_id": ObjectId("5b8fad70f00832a8a5eb5838"),
  "Rno":"3",
  "Name": "Ashley",
  "Class": "TECOMP"
}
{
  "id": ObjectId("5b8fad7ff80832a0a58b5039"),
  "Rno": "4",
  "Name": "Henry",
  "Class": "TECOMP"
}
{
  id: ObjectId("5b8fad8df00832a0a5eb583a"),
  "no": "5",
  "Name": "Pratik",
  "Class": "TECOMP"
}
{
  "id": ObjectId("5b8fada4f00832aea50b5836"),
  "no":"6",
  "Name": "Pratik",
  "Class": "TECOMP"
}

```

```
}
```

```
> db.Student.find((Sand [{"Name":"Piyush"}, {"no":"2"}]));
> db.Student.find((Sand [{"Name":"Piyush"},
{"Rno":"1"}])).pretty();
{
  "_id": ObjectId("5ba1d61845bbacd4a81568d"),
  "Rno": "1",
  "Name": "Piyush",
  "Class": "TECOMP"
}
> db.Student.find((Sand [{"Name":"Piyush"},
{"Rno":"2"}])).pretty();
> db.Student.find((Sor:[{"Name":"Piyush"},
{"Rno":"2"}])).pretty();
{
  "_id": ObjectId("5ba1d618f5bbacd4ad81568d"),
  "Rno": "1",
  "Name": "Piyush",
  "Class": "TECOMP"
}
{
  "_id": ObjectId("5ba1d625f5bbacd4ad81568e"),
  "Rno": "2",
  "Name": "mohit",
  "Class": "TECOMP"
}
> db.Student.find((Sor:[{"Name":"Piyush"}, {"Class":"TECOMP"}])).pretty();
{
  "_id": ObjectId("5ba1d618f5bbacd4ad81568d"),
  "Rno": "1",
  "Name": "Piyush",
  "Class": "TECOMP"
}
{
  "_id": ObjectId("5ba1d625f5bbacd4ad81568e"),
  "Rno": "2",
  "Name": "mohit",
  "Class": "TECOMP"
}
{
  "id": ObjectId("5ba1d63af5bbacd4a81568"),
  "Rno": "3",
  "Name": "Ashley",
  "class": "TECOMP"
}
{
  "_id": ObjectId("5ba1d647f5bbacd4ad815690"),
  "Rno": "4",
  "Name": "Htresh",
  "Class": "TECOMP"
}
```

```

}
{
  "_id": ObjectId("5ba1d65ef5bbacd4ad815691"),
  "Rno": "5",
  "Name": "Pratik", "Class": "TECOMP"
}
{
  "_id": ObjectId("5ba1d66df5bbacd4ad815692"),
  "Rno": "6",
  "Name": "Pratik",
  "Class": "TECOMP"
}
> db.Student.find(($or:[{"Name":"Piyush"}, {"Class":"TECOMP"}])).pretty();
> db.Student.find(($or:[{"Name":"Piyush"},
{"Rno":"2"}])).pretty();
{
  "_id": ObjectId("5ba1d63af5bbacd4ad81568f"),
  "Rno": "3",
  "Name": "Ashley",
  "Class": "TECOMP"
}
{
  "_id": ObjectId("5ba1d647f5bbacd4ad815690"),
  "Rno": "4",
  "Name": "Htresh",
  "Class": "TECOMP"
}
{
  "_id": ObjectId("5ba1d65ef5bbacd4ad815691"), "Rno": "5",
  "Name": "Pratik",
  "Class": "TECOMP"
}
{
  "_id": ObjectId("5ba1d66df5bbacd4ad815692"),
  "Name": "Pratik",
  "Rno": "6",
  "Class": "TECOMP"
}
db.Student.find(("Rno": ($not: ($lt:"3")))).pretty();
{
  "_id": ObjectId("5ba1d63af5bbacd4ad81568f"),
  "Rno": "3",
  "Name": "Ashley", "Class": "TECOMP"
}
{
  "_id": ObjectId("5ba1d647f5bbacd4ad815690"),
  "Rno": "4",
  "Name": "Htresh",
  "Class": "TECOMP"
}
{
  "_id": ObjectId("5ba1d65ef5bbacd4ad815691"),

```



```

    "Rno": "5",
    "Name": "Pratik",
    "Class": "TECOMP"
  }
  {
    "_id": ObjectId("5ba1d66df5bbacd4ad815692"),
    "Rno": "6", "Name": "Pratik",
    "Class": "TECOMP"
  }
  > db.Student.find({"Rno": { $eq:"5"}}).pretty();
  {
    "_id": ObjectId("5ba1d65ef5bbacd4ad815691"),
    "Rno": "5",
    "Name": "Pratik",
    "Class": "TECOMP"
  }
  > db.Student.find({"Rno": { $ne:"5"}}).pretty();
  {
    "_id": ObjectId("5ba1d618f5bbacd4ad81568d"),
    "Rno": "1",
    "Name": "Piyush",
    "Class": "TECOMP"
  }
  {
    "_id": ObjectId("5ba1d625f5bbacd4ad81568e"),
    "Rno": "2",
    "Name": "mohit",
    "Class": "TECOMP"
  }
  {
    "_id": ObjectId("5ba1d63af5bbacd4ad81568f"),
    "Rno": "3", "Name": "Ashley",
    "Class": "TECOMP"
  }
  {
    "_id": ObjectId("5ba1d647f5bbacd4ad815698"),
    "Rno": "4", "Name": "Htेश",
    "Class": "TECOMP"
  }
  {
    "_id": ObjectId("5ba1d66df5bbacd4ad815692"),
    "Rno": "6",
    "Name": "Pratik",
    "Class": "TECOMP"
  }
  > db.Student.find( {"Rno": {$gt:"5"}}).pretty();
  {
    "_id": ObjectId("5ba1d66df5bbacd4ad815692"),
    "Rno": "6",
    "Name": "Pratik",
    "Class": "TECOMP"
  }
}

```

```

> db.Student.find({"Rno": {$gte:"5"}}).pretty();
{
  "_id": ObjectId("5ba1d65ef5bbacd4ad815691"),
  "Rno": "5",
  "Name": "Pratik",
  "Class": "TECOMP"
}
{
  "id": ObjectId("5ba1d66df5bbacd4ad815692"),
  "Rno": "6",
  "Name": "Pratik",
  "Class": "TECOMP"
}
> db.Student.find( ("Rno": { $lt:"5"})).pretty();
{
  "_id": ObjectId("5ba1d618f5bbacd4ad81568d"),
  "Rno": "1",
  "Name": "Piyush",
  "Class": "TECOMP"
}
{
  id: ObjectId("5ba1d625f5bbacd4ad81568e"),
  "Rno": "2",
  "Name": "mohit",
  "Class": "TECOMP"
}
{
  "_id": ObjectId("5ba1d63af5bbacd4ad81568f"),
  "Rno": "3",
  "Name": "Ashley",
  "Class": "TECOMP"
}
{
  "id": ObjectId("5ba1d647f5bbacd4ad815690"),
  "Rno": "4",
  "Name": "Htresh",
  "Class": "TECOMP"
}
> db.Student.find( ("Rno": { $lte:"5"})).pretty();
{
  "_id": ObjectId("5ba1d618f5bbacd4ad81568d"),
  "Rno": "1",
  "Name": "Piyush",
  "Class": "TECOMP"
}
{
  "id": ObjectId("5ba1d625f5bbacd4ad81568e"),
  "Rno": "2", "Name": "mohit",
  "Class": "TECOMP"
}
{
  "id": ObjectId("5ba1d63af5bbacd4ad81568f"),

```

```

    "Rno": "3", "Name": "Ashley",
    "Class": "TECOMP"
  }
  {
    "_id": ObjectId("5ba1d647f5bbacd4ad815690"),
    "Rno": "4",
    "Name": "Htesh",
    "Class": "TECOMP"
  }
  {
    "_id": ObjectId("5ba1d65ef5bbacd4ad815691"),
    "Rno": "5",
    "Name": "Pratik", "Class": "TECOMP"
  }
}
> db.Student.find(("Rno": { $in: ["5", "2"]})) pretty();
{
  "_id": ObjectId("5ba1d63af5bbacd4a81568f"),
  "Rno": "3",
  "Name": "Ashley",
  "Class": "TECOMP"
}
{
  "_id": ObjectId("5ba1d647f5bbacd4ad815696"),
  "Rno": "4",
  "Name": "Htesh",
  "Class": "TECOMP"
}
> db.Student.find(["Rno": { $in: ["5", "2"]}], pretty);
{
  "_id": ObjectId("5ba1d625f5bbacd4ad81568e"),
  "Rno": "2",
  "Name": "mohit",
  "Class": "TECOMP"
}
{
  "_id": ObjectId("5ba1d63af5bbacd4ad81568f"),
  "Rno": "3",
  "Name": "Ashley",
  "Class": "TECOMP"
}
{
  "_id": ObjectId("5ba1d647f5bbacd4ad815690"),
  "Rno": "4",
  "Name": "Htesh",
  "Class": "TECOMP"
}
{
  "_id": ObjectId("5ba1d65ef5bbacd4ad815691"),
  "Rno": "5",
  "Name": "Pratik",
  "Class": "TECOMP"
}
}

```

```

> db.Student.find({"Rno": { $lte:"5", $gte:"2" }}).pretty();
{
  "_id": ObjectId("5ba1d63af5bbacd4ad815687"),
  "Rno": "3",
  "Name": "Ashley", "Class": "TECOMP"
}
{
  "_id": ObjectId("5ba1d647f5bbacd4ad815698"),
  "Rno": "4",
  "Name": "Htresh",
  "class": "TECOMP"
}
{
  "_id": ObjectId("5ba1d65ef5bbacd4ad815691"),
  "Rno": "5",
  "Name": "Pratik",
  "Class": "TECOMP"
}
> db.Student.find( {"Rno": { $lt:"5", $gte: "2" }}).pretty();
{
  "_id": ObjectId("5ba1d625f5bbacd4ad81568e"),
  "Rno": "2",
  "Name": "mohit",
  "Class": "TECOMP"
}
{
  "_id": ObjectId("5ba1d63af5bbacd4ad81568f"),
  "Rno": "3",
  "Name": "Ashley",
  "Class": "TECOMP"
}
{
  "_id": ObjectId("5ba1d647f5bbacd4ad815690"),
  "Rno": "4",
  "Name": "Htresh",
  "Class": "TECOMP"
}

```

Group B
Assignment Number 3
MongoDB

Problem Statement:

Design and Develop MongoDB Queries using aggregation and indexing with suitable example using MongoDB.

PROGRAM

```
> use
comp;

switched to db comp
//CREATE COLLECTION WEBSITE
> db.createCollection('website');
{ "ok" : 1 }
//INSERT VALUES IN WEBSITE
> db.website.insert({roll: '1', 'name': 'harsh', 'amount': 1000, 'url': 'www.yahoo.com'});
WriteResult({ "nInserted" : 1 })

> db.website.insert({roll: '2', 'name': 'jitesh', 'amount': 2000, 'url': 'www.yahoo.com'});
WriteResult({ "nInserted" : 1 })

> db.website.insert({roll: '3', 'name': 'rinal', 'amount': 3000, 'url': 'www.google.com'});
WriteResult({ "nInserted" : 1 })

> db.website.insert({roll: '4', 'name': 'ash', 'amount': 4000, 'url': 'www.gmail.com'});
WriteResult({ "nInserted" : 1 })

> db.website.insert({roll: '5', 'name': 'ash', 'amount': 1000, 'url': 'www.pvg.com'});
WriteResult({ "nInserted" : 1 })

//SUM AGGREGATE
> db.website.aggregate({$group: {_id: "$name", "total": {$sum: "$amount"}}});
{"_id": "ash", "total": 5000}
{"_id": "rinal", "total": 3000}
{"_id": "jitesh", "total": 2000}
{"_id": "harsh", "total": 2000}

//AVG AGGREGATE
> db.website.aggregate({$group: {_id: "$name", "total": {$avg: "$amount"}}});
{"_id": "ash", "total": 2500}
{"_id": "rinal", "total": 3000}
{"_id": "jitesh", "total": 2000}
{"_id": "harsh", "total": 1000}

//MIN AGGREGATION
> db.website.aggregate({$group: {_id: "$name", "total": {$min: "$amount"}}});
{"_id": "ash", "total": 1000}
{"_id": "rinal", "total": 3000}
{"_id": "jitesh", "total": 2000}
```

```
{ "id": "harsh", "total": 1000 }
```

```
//MAX AGGREGATION
```

```
> db.website.aggregate({$group: { _id: "$name", "total": { $max: "$amount" }}});  
{ "_id": "ash", "total": 4000 }  
{ "_id": "rina", "total": 3000 }  
{ "_id": "jitesh", "total": 2000 }  
{ "_id": "harsh", "total": 1000 }
```

```
//FIRST AGGREGATION
```

```
> db.website.aggregate({$group: { _id: "$name", "total": { $first: "$amount" }}});  
{ "_id": "ash", "total": 4000 }  
{ "_id": "rina", "total": 3000 }  
{ "_id": "jitesh", "total": 2000 }  
{ "_id": "harsh", "total": 1000 }
```

```
//LAST AGGREGATION
```

```
> db.website.aggregate({$group: { _id: "$name", "total": { $last: "$amount" }}});  
{ "_id": "ash", "total": 1000 }  
{ "_id": "rina", "total": 3000 }  
{ "_id": "jitesh", "total": 2000 }  
{ "_id": "harsh", "total": 1000 }
```

```
//PUSH AGGREGATION
```

```
> db.website.aggregate({$group: { _id: "$name", "total": { $push: "$amount" }}});  
{ "_id": "ash", "total": [ 4000, 1808 ] }  
{ "_id": "rina", "total": [ 3000 ] }  
{ "_id": "jitesh", "total": [ 2008 ] }  
{ "_id": "harsh", "total": [ 1800, 1000 ] }
```

```
//COUNT AGGREGATION
```

```
> db.website.aggregate({$group: { _id: "$name", "total": { $sum: 1 }}});  
{ "_id": "ash", "total": 2 }  
{ "_id": "rina", "total": 1 }  
{ "_id": "jitesh", "total": 1 }  
{ "_id": "harsh", "total": 2 }
```

```
//ADDTOSSET AGGREGATE
```

```
> db.website.aggregate({$group: { _id: "$name", "total": { $addToSet: "$amount" }}});  
{ "_id": "ash", "total": [ 1000, 4008 ] }  
{ "_id": "rina", "total": [ 3000 ] }  
{ "_id": "jitesh", "total": [ 2000 ] }  
{ "_id": "harsh", "total": [ 1000 ] }
```

```
//INDEXING
```

```
> db.createCollection('website1');  
{ "ok" : 1 }  
> db.website1.insert({ "r": 1, "name": "harsh" });  
WriteResult({ "nInserted": 1 })  
> db.website1.find().pretty()  
{ "_id" : ObjectId("5ba3509a444926329738012d"), "roll": 1, "name": "harsh" }  
{ "_id" : ObjectId("5ba35293444926329738012e"), "roll": 1, "name": "harsh" }
```

```

> db.website1.createIndex({name:1})
{"numIndexesBefore": 2, "note": "all indexes already exist", "ok": 1}

//CREATE INDEXING
> db.website1.createIndex ({name:1})
{
  "createdCollectionAutomatically": false,
  "numIndexesBefore": 2,
  "numIndexesAfter": 3,
  "ok": 1
}

> db.website1.getIndexes()
2018-09-20T13:28:09.628+0530 TypeError: Property 'getIndexes' of object omwebsite is not a function
> db.website1.getIndexes()
[
  { "v": 1,
    "key": {
      "id": 1
    },
    "name": "_id_",
    "ns": "harsh.website1"
  },
  {
    "v": 1,
    "key": {
      "name": 1
    },
    "name": "name_1",
    "ns": "harsh.website1"
  },
  {
    "v": 1,
    "key": {
      "name": -1
    },
    "name": "name_-1",
    "ns": "harsh.website1"
  }
]
> db.website1.createIndex({"name":-1})
{"numIndexesBefore": 3, "note": "all indexes already exist", "ok": 1}

//DROP INDEX
> db.website1.dropIndex({name:-1})
{"numIndexesBefore": 3, "ok": 1}
> db.website1.dropIndex({name:1})
{"numIndexesBefore": 2, "ok": 1}
> db.website1.dropIndex({name:1})
{
  "numIndexesBefore": 1,
  "ok": 0,
  "errmsg": "can't find index with key: { name: 1.0}"
}

```

```

}

//GET INDEXING
> db.website1.getIndexes()
[
  {
    "v": 1,
    "key": {
      "_id": 1
    },
    "name": "_id_",
    "ns": "harsh.website"
  }
]
> db.website1.find().pretty()
{"_id": ObjectId("5ba3509a444926329738812d"), "roll": 1, "name": "harsh"}
{"_id": ObjectId("5ba35293444926329738012e"), "roll": 1, "name": "harsh"}

>
> db.website1.createIndex({name:1})
{
  "createdCollectionAutomatically": false,
  "numIndexesBefore": 1,
  "numIndexesAfter": 2,
  "ok": 1
}
> db.website1.getIndexes()
[
  {
    "v": 1,
    "key": {
      "_id": 1
    },
    "name": "_id_",
    "ns": "harsh.website"
  },
  {
    "v": 1,
    "key": { "name": 1 },
    "name": "name_",
    "ns": "harsh.website1"
  }
]
> db.website1.dropIndex({name:1})
{"numIndexesWas": 2, "ok": 1}
> db.website1.getIndexes()
[
  {
    "v": 1,
    "key": {
      "_id": 1
    },
    "name": "_id_",

```



```

    "ns" "harshwebsite1"
  }
]
> dbwebsite1.createIndex({name:1, 'r':-1})
{"createdCollectionAutomatically":false,
"numIndexesBefore": 1,
"numIndexesAfter": 2,
"ok":1
}

> dbwebsite1.getIndexes()
[
{
  "v": 1,
  "key": {
    "_id": 1
  },
  "name": "_id_",
  "ns": "harshwebsite1"
},
{
  "v": 1,
  "key": {
    "name": 1,
    "r" : -1
  },
  "name": "1_r_-1",
  "ns": "harshwebsite1"
}
]
(i-search) dbwebsite1.insert({'roll':1, 'name': 'harsh'});

```

Group B
Assignment Number 4
MongoDB

Problem Statement:

Implement Map reduces operation with suitable example using Mongo DB.

PROGRAM

> use
Sudarshandb

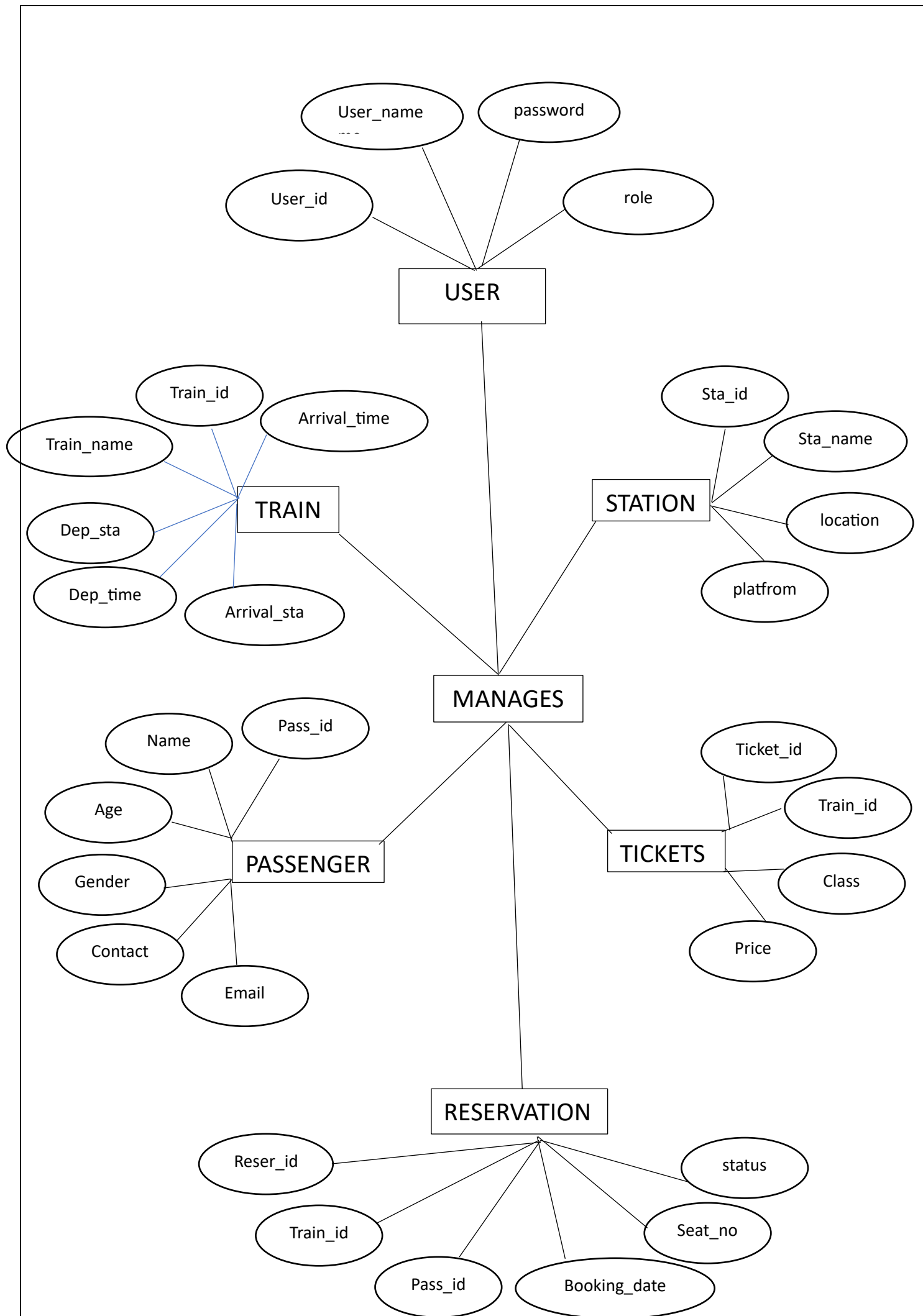
```
switched to db Sudarshandb
> db.createCollection("Journal");
{"ok": 1}
> db.Journal.insert({"book_id":1, 'book_name':"Javacd OOP", "amt":500, "status": 'Available' });
WriteResult({"nInserted": 1})
> db.Journal.insert({"book_id":1, "book_name":"Java OOP", "amt:400, "status":"Not Available"});
WriteResult({"nInserted": 1})
> db.Journal.insert({"book_id":1, 'book_name':"Java", "amt": 300, "status:'Not Available'});
WriteResult({"nInserted": 1})
> db.Journal.insert({"book_id":2, "book_name":"Java", 'amt:300, "status":"Available"});
WriteResult({"nInserted": 1})>
> db.Journal.insert({"book_id":2, "book_name": 'OFP', amt:200, 'status Available'});
WriteResult({"nInserted": 1})
> db.Journal.insert({"book_id":2, 'book_name': 'C++', amt":200, 'status: 'Available'});
WriteResult({"nInserted": 1})
> db.Journal.insert({"book_id":3, 'book_name': 'C++', amt":150, "status":"Available"});
WriteResult({"nInserted": 1})
> db.Journal.insert({"book_id":3, 'book_name':"C++", amt":200, 'status: 'Not Available'});
WriteResult({"nInserted": 1})
> db.Journal.insert({"book_id":4, "book_name": 'OFP C++', amt":300, 'status': 'Not Available'});
WriteResult({"nInserted": 1})
> db.Journal.insert({"book_id":5, "book_name": 'OFP C++', 'amt:400, 'status: 'Available'});
WriteResult({"nInserted": 1})
> db.Journal.insert({"book_id":5, "book_name": "C", "amt":400, "status": "Available"});
WriteResult({"nInserted": 1})
> db.Journal.insert({"book_id":5, "book_name": "C++ Java", 'amt:400, 'status: Not Available'});
WriteResult({"nInserted": 1})
>
>
>
> var mapfunction=function(){ emit(this.book_id, this.amt);
> var reducefunction=function (key, value) {return
Array.sum(value);};

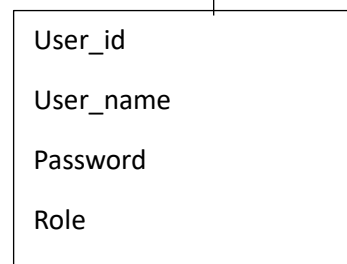
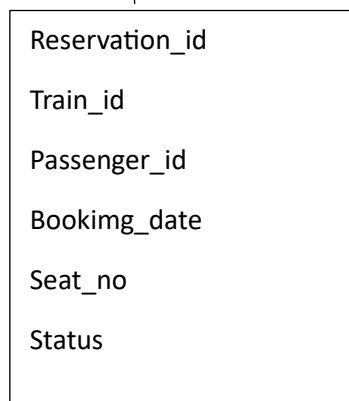
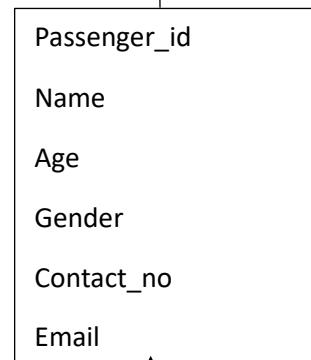
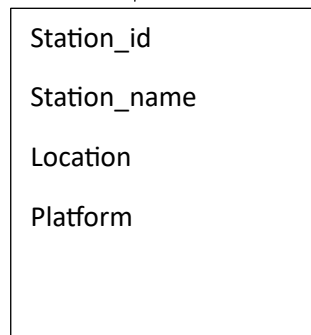
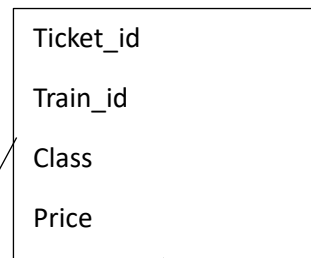
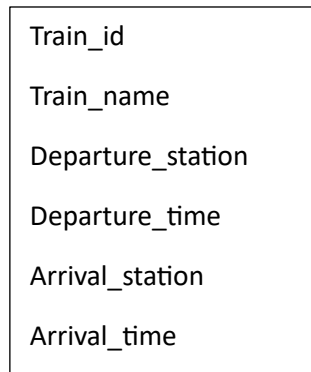
> db.Journal.mapReduce (mapfunction, reduce function,
{out:'new'});
{
"result": "new",
```

```
"timeMillis": 49,"counts": {  
  "input": 12,  
  "emit": 12,  
  "reduce": 4,  
  "output": 5  
},  
"ok": 1  
}
```

```
> db.journal.mapReduce (mapfunction, reducefunction,  
  {"out":"new"}).find().pretty();  
{ "_id": 1, "value": 1200 }  
{ "_id": 2, "value": 700 }  
{ "_id": 3, "value": 350 }  
{ "_id": 4, "value": 300 }  
{ "_id": 5, "value": 1200 }
```

```
> db.new.find().pretty();  
{ "_id": 1, "value": 1200 }  
{ "_id": 2, "value": 700 }  
{ "_id": 3, "value": 350 }  
{ "_id": 4, "value": 300 }  
{ "_id": 5, "value": 1200 }
```





Assignment 3 – JOINS

1. Inner join

Autocommit Rows 10 Save Run

```
SELECT P.Name, Tr.Train_Name
FROM P
INNER JOIN Re ON P.Passenger_ID = Re.Passenger_ID
INNER JOIN Tr ON Re.Train_ID = Tr.Train_ID;
```

Results Explain Describe Saved SQL History

NAME	TRAIN_NAME
John Doe	Express 101
Jane Smith	Local 202
Bob Johnson	Superfast 303

3 rows returned in 0.00 seconds [Download](#)

2. Left join

Autocommit Rows 10 Save Run

```
SELECT Tr.Train_Name, St.Station_Name AS Departure_Station
FROM Tr
LEFT JOIN St ON Tr.Departure_Station = St.Station_ID;
```

Results Explain Describe Saved SQL History

TRAIN_NAME	DEPARTURE_STATION
Express 101	Station A
Local 202	Station B
Superfast 303	Station C

3 rows returned in 0.00 seconds [Download](#)

3. Right join

Autocommit Rows 10 Save Run



```
SELECT St.Station_Name, Tr.Train_Name
FROM St
RIGHT JOIN Tr ON St.Station_ID = Tr.Departure_Station;
```

Results Explain Describe Saved SQL History

STATION_NAME	TRAIN_NAME
Station A	Express 101
Station B	Local 202
Station C	Superfast 303

3 rows returned in 0.01 seconds [Download](#)

4. Self join

☒ Autocommit Rows 10   Save Run



```
SELECT A.Station_Name AS Station1, B.Station_Name AS Station2
FROM St A
INNER JOIN St B ON A.Location = B.Location
WHERE A.Station_ID <> B.Station_ID;
```

Results Explain Describe Saved SQL History

STATION1	STATION2
Station C	Station B
Station B	Station C
Station Z	Station Y
Station Y	Station Z

4 rows returned in 0.01 seconds [Download](#)

5. Cross join

☒ Autocommit Rows 10   Save Run



```
SELECT St.Station_Name, Tr.Train_Name
FROM St
CROSS JOIN Tr;
```

Results Explain Describe Saved SQL History

STATION_NAME	TRAIN_NAME
Station A	Express 101
Station B	Express 101
Station C	Express 101
Station Y	Express 101
Station Z	Express 101
Station A	Local 202
Station B	Local 202
Station C	Local 202
Station Y	Local 202
Station Z	Local 202

More than 10 rows available. Increase rows selector to view more rows.
10 rows returned in 0.00 seconds [Download](#)

6. Self Join for Hierarchy

☒ Autocommit Rows 10   Save Run

```
SELECT A.Station_Name AS Parent_Station, B.Station_Name AS Child_Station
FROM St A
LEFT JOIN St B ON A.Station_ID = B.Location
WHERE B.Station_ID IS NOT NULL;
```

Results Explain Describe Saved SQL History

PARENT_STATION	CHILD_STATION
Station A	Station B
Station A	Station C

2 rows returned in 0.01 seconds [Download](#)

7. Join with multiple condition

```
Autocommit Rows 10 Save Run
```



```
SELECT P.Name, Re.Status
FROM P
INNER JOIN Re ON P.Passenger_ID = Re.Passenger_ID
WHERE Re.Status = 'Confirmed';
```

Results Explain Describe Saved SQL History

NAME	STATUS
John Doe	Confirmed
Jane Smith	Confirmed

2 rows returned in 0.01 seconds [Download](#)

8. Join with subquery

☒ Autocommit
 Rows


Save Run

```

SELECT P.Name
FROM P
WHERE P.Passenger_ID IN (
    SELECT Re.Passenger_ID
    FROM Re
    INNER JOIN Tr ON Re.Train_ID = Tr.Train_ID
    WHERE Tr.Departure_Station = 1 -- Replace with the station ID for 'Station A'
);
    
```

Results
[Explain](#)
[Describe](#)
[Saved SQL](#)
[History](#)

NAME
John Doe

1 rows returned in 0.03 seconds [Download](#)

9. Join with aggregation

[illegible]

10. Full outer join

Autocommit

ROWS

Save

Run

```
SELECT P.Name, COALESCE(Tr.Train_Name, 'Not Reserved') AS Reserved_Train
FROM P
LEFT JOIN Re ON P.Passenger_ID = Re.Passenger_ID
LEFT JOIN Tr ON Re.Train_ID = Tr.Train_ID;
```

Results

Explain

Describe

Saved SQL

History

NAME	RESERVED_TRAIN
John Doe	Express 101
Jane Smith	Local 202
Bob Johnson	Superfast 303

3 rows returned in 0.01 seconds [Download](#)