Mongo DB

Exercise 1:

Create Database and Add A Collection

```
>
> show dbs;
admin   0.000GB
config  0.000GB
local   0.000GB
> use sample;
switched to db sample
> db.createCollection("person");
{ "ok" : 1 }
> show dbs
admin   0.000GB
config  0.000GB
local   0.000GB
sample  0.000GB
> show collections
person
>
```

Use command is used to create a database

Exercise 2

**Crud Operations**

Inserting  Single Document

```
> show collections
person
> db.person.insert({ sno:1,name:"Raj",city:"Chennai"});
WriteResult({ "nInserted" : 1 })
```

Inserting Multiple Documents

```
> db.person.insertMany([
... {sno:3,name:"Vishal",city:"Pune"},
... {sno:4,name:"Arjun",city:"Delhi"},
... {sno:5,name:"Rajan",city:"Mumbai"}
... ]);
{
        "acknowledged" : true,
        "insertedIds" : [
                ObjectId("64047f95821fedea387e45b7"),
                ObjectId("64047f95821fedea387e45b8"),
                ObjectId("64047f95821fedea387e45b9")
        ]
}
```

Exercise 3

Select Operation

- Find Function

```
> db.person.find();
{ "_id" : ObjectId("64047e3d821fedea387e45b5"), "sno" : 1, "name" : "Raj", "city" : "Chennai" }
{ "_id" : ObjectId("64047e8a821fedea387e45b6"), "sno" : 2, "name" : "Rahul", "city" : "Mumbai" }
{ "_id" : ObjectId("64047f95821fedea387e45b7"), "sno" : 3, "name" : "Vishal", "city" : "Pune" }
{ "_id" : ObjectId("64047f95821fedea387e45b8"), "sno" : 4, "name" : "Arjun", "city" : "Delhi" }
{ "_id" : ObjectId("64047f95821fedea387e45b9"), "sno" : 5, "name" : "Rajan", "city" : "Mumbai" }
>
```

For Every Document, Document id is generated represented by the property _id

- Find with pretty option

```
> db.person.find().pretty()
{
        "_id" : ObjectId("64047e3d821fedea387e45b5"),
        "sno" : 1,
        "name" : "Raj",
        "city" : "Chennai"
}
{
        "_id" : ObjectId("64047e8a821fedea387e45b6"),
        "sno" : 2,
        "name" : "Rahul",
        "city" : "Mumbai"
}
{
        "_id" : ObjectId("64047f95821fedea387e45b7"),
        "sno" : 3,
        "name" : "Vishal",
        "city" : "Pune"
}
{
        "_id" : ObjectId("64047f95821fedea387e45b8"),
        "sno" : 4,
        "name" : "Arjun",
        "city" : "Delhi"
}
{
        "_id" : ObjectId("64047f95821fedea387e45b9"),
        "sno" : 5,
        "name" : "Rajan",
        "city" : "Mumbai"
}
>
```

- Operators and Selection

Equal to operator

```
> db.person.find({sno:1});
{ "_id" : ObjectId("64047e3d821fedea387e45b5"), "sno" : 1, "name" : "Raj", "city" : "Chennai" }
> db.person.find({city:"Mumbai"})
{ "_id" : ObjectId("64047e8a821fedea387e45b6"), "sno" : 2, "name" : "Rahul", "city" : "Mumbai" }
{ "_id" : ObjectId("64047f95821fedea387e45b9"), "sno" : 5, "name" : "Rajan", "city" : "Mumbai" }
```

Selection using Comparison

```
> db.person.find({sno:{$gt:3}});
{ "_id" : ObjectId("64047f95821fedea387e45b8"), "sno" : 4, "name" : "Arjun", "city" : "Delhi" }
{ "_id" : ObjectId("64047f95821fedea387e45b9"), "sno" : 5, "name" : "Rajan", "city" : "Mumbai" }
> db.person.find({sno:{$lte:3}});
{ "_id" : ObjectId("64047e3d821fedea387e45b5"), "sno" : 1, "name" : "Raj", "city" : "Chennai" }
{ "_id" : ObjectId("64047e8a821fedea387e45b6"), "sno" : 2, "name" : "Rahul", "city" : "Mumbai" }
{ "_id" : ObjectId("64047f95821fedea387e45b7"), "sno" : 3, "name" : "Vishal", "city" : "Pune" }
```

Here $gt and $lte are the relational operators

- Logical Operators and Selection

```
> db.person.find(
... { $or:
...  [ {city:"Chennai"},
...    {city: "Mumbai"}
... ]}
... );
{ "_id" : ObjectId("64047e3d821fedea387e45b5"), "sno" : 1, "name" : "Raj", "city" : "Chennai" }
{ "_id" : ObjectId("64047e8a821fedea387e45b6"), "sno" : 2, "name" : "Rahul", "city" : "Mumbai" }
{ "_id" : ObjectId("64047f95821fedea387e45b9"), "sno" : 5, "name" : "Rajan", "city" : "Mumbai" }
> db.person.find(
... {
...  $and:
...   [ {sno:{$gt:3}},
...     {city: "Mumbai"}
...   ]}
... );
{ "_id" : ObjectId("64047f95821fedea387e45b9"), "sno" : 5, "name" : "Rajan", "city" : "Mumbai" }
>
```

In the first statement, we select document where city is either Chennai or Mumbai and "or" is the logical operator

In the Second Statement, we select documents where city is Mumbai and the sno greater than 3

And "and " is the logical operator

Exercise 4:

- Projection Operation

Selecting only selected properties

```
> db.person.find(
... {},
... {name:1}
... );
{ "_id" : ObjectId("64047e3d821fedea387e45b5"), "name" : "Raj" }
{ "_id" : ObjectId("64047e8a821fedea387e45b6"), "name" : "Rahul" }
{ "_id" : ObjectId("64047f95821fedea387e45b7"), "name" : "Vishal" }
{ "_id" : ObjectId("64047f95821fedea387e45b8"), "name" : "Arjun" }
{ "_id" : ObjectId("64047f95821fedea387e45b9"), "name" : "Rajan" }
> db.person.find(
... {},
... {_id:0,name:1}
... );
{ "name" : "Raj" }
{ "name" : "Rahul" }
{ "name" : "Vishal" }
{ "name" : "Arjun" }
{ "name" : "Rajan" }
> db.person.find( {sno:3}, {_id:0,name:1} );
{ "name" : "Vishal" }
```

In the First Statement, we select only name property and _id is also displayed

In the second Statement, We avoid _id column also

In the third statement, we use condition in the first parameter

Exercise 5

Update Statement

```
> db.person.find({},{_id:0});
{ "sno" : 1, "name" : "Raj", "city" : "Chennai" }
{ "sno" : 2, "name" : "Rahul", "city" : "Mumbai" }
{ "sno" : 3, "name" : "Vishal", "city" : "Pune" }
{ "sno" : 4, "name" : "Arjun", "city" : "Delhi" }
{ "sno" : 5, "name" : "Rajan", "city" : "Mumbai" }
> db.person.updateMany({sno:{$in:[3,4]}},
... {$set:{city:"Gandhi Nagar"}}
... );
{ "acknowledged" : true, "matchedCount" : 2, "modifiedCount" : 2 }
> db.person.find({},{_id:0});
{ "sno" : 1, "name" : "Raj", "city" : "Chennai" }
{ "sno" : 2, "name" : "Rahul", "city" : "Mumbai" }
{ "sno" : 3, "name" : "Vishal", "city" : "Gandhi Nagar" }
{ "sno" : 4, "name" : "Arjun", "city" : "Gandhi Nagar" }
{ "sno" : 5, "name" : "Rajan", "city" : "Mumbai" }
```

City is set to Gandhi Nagar After update, also we have used in operator to select the sno.

**Note:** there is a function called update which update only the first match , use updateMany

**Exercise 6**

- Remove Operator

```
> db.person.find({},{_id:0});
{ "sno" : 1, "name" : "Raj", "city" : "Chennai" }
{ "sno" : 2, "name" : "Rahul", "city" : "Mumbai" }
{ "sno" : 3, "name" : "Vishal", "city" : "Gandhi Nagar" }
{ "sno" : 4, "name" : "Arjun", "city" : "Gandhi Nagar" }
{ "sno" : 5, "name" : "Rajan", "city" : "Mumbai" }
> db.person.remove({city:"Mumbai"});
WriteResult({ "nRemoved" : 2 })
```

The above command removes two documents

Exercise 7

More Selection Operators

- $where

```
> db.person.find({},{_id:0});
{ "sno" : 1, "name" : "Raj", "city" : "Chennai" }
{ "sno" : 3, "name" : "Vishal", "city" : "Gandhi Nagar" }
{ "sno" : 4, "name" : "Arjun", "city" : "Gandhi Nagar" }
{ "sno" : 2, "name" : "Rahul", "city" : "Mumbai" }
{ "sno" : 5, "name" : "Rajan", "city" : "Mumbai" }
> db.person.find({$where:"this.sno>2 && this.sno<5"} , {_id:0} );
{ "sno" : 3, "name" : "Vishal", "city" : "Gandhi Nagar" }
{ "sno" : 4, "name" : "Arjun", "city" : "Gandhi Nagar" }
> db.person.find({$where:"this.city.length>5"},{_id:0});
{ "sno" : 1, "name" : "Raj", "city" : "Chennai" }
{ "sno" : 3, "name" : "Vishal", "city" : "Gandhi Nagar" }
{ "sno" : 4, "name" : "Arjun", "city" : "Gandhi Nagar" }
{ "sno" : 2, "name" : "Rahul", "city" : "Mumbai" }
{ "sno" : 5, "name" : "Rajan", "city" : "Mumbai" }
> db.person.find({$where:"this.city.length>8"},{_id:0});
{ "sno" : 3, "name" : "Vishal", "city" : "Gandhi Nagar" }
{ "sno" : 4, "name" : "Arjun", "city" : "Gandhi Nagar" }
>
```

- Regular Expressions

```
> db.person.find({},{_id:0});
{ "sno" : 1, "name" : "Raj", "city" : "Chennai" }
{ "sno" : 3, "name" : "Vishal", "city" : "Gandhi Nagar" }
{ "sno" : 4, "name" : "Arjun", "city" : "Gandhi Nagar" }
{ "sno" : 2, "name" : "Rahul", "city" : "Mumbai" }
{ "sno" : 5, "name" : "Rajan", "city" : "Mumbai" }
> db.person.find({name:{$regex:/^R[a-z]+/}})
{ "_id" : ObjectId("64047e3d821fedea387e45b5"), "sno" : 1, "name" : "Raj", "city" : "Chennai" }
{ "_id" : ObjectId("6404888d821fedea387e45ba"), "sno" : 2, "name" : "Rahul", "city" : "Mumbai" }
{ "_id" : ObjectId("6404888d821fedea387e45bb"), "sno" : 5, "name" : "Rajan", "city" : "Mumbai" }
>
```