Name: Tania Rajabally

Branch: Comps

Batch: C

Roll no: 43

UID:2017130047

Date: 23/10/2020

**Experiment 7**

**Aim:** To understand No SQL and write queries.

**Theory:**

A NoSQL originally referring to non SQL or non relational is a database that provides a mechanism for storage and retrieval of data. This data is modeled in means other than the tabular relations used in relational databases. Such databases came into existence in the late 1960s, but did not obtain the NoSQL moniker until a surge of popularity in the early twenty-first century. NoSQL databases are used in real-time web applications and big data and their use are increasing over time. NoSQL systems are also sometimes called Not only SQL to emphasize the fact that they may support SQL-like query languages.

A NoSQL database includes simplicity of design, simpler horizontal scaling to clusters of machines and finer control over availability. The data structures used by NoSQL databases are different from those used by default in relational databases which makes some operations faster in NoSQL. The suitability of a given NoSQL database depends on the problem it should solve. Data structures used by NoSQL databases are sometimes also viewed as more flexible than relational database tables.

MongoDB is a cross-platform, document oriented database that provides, high performance, high availability, and easy scalability. MongoDB works on concept of collection and document.

* Database

Database is a physical container for collections. Each database gets its own set of files on the file system. A single MongoDB server typically has multiple databases.

* Collection

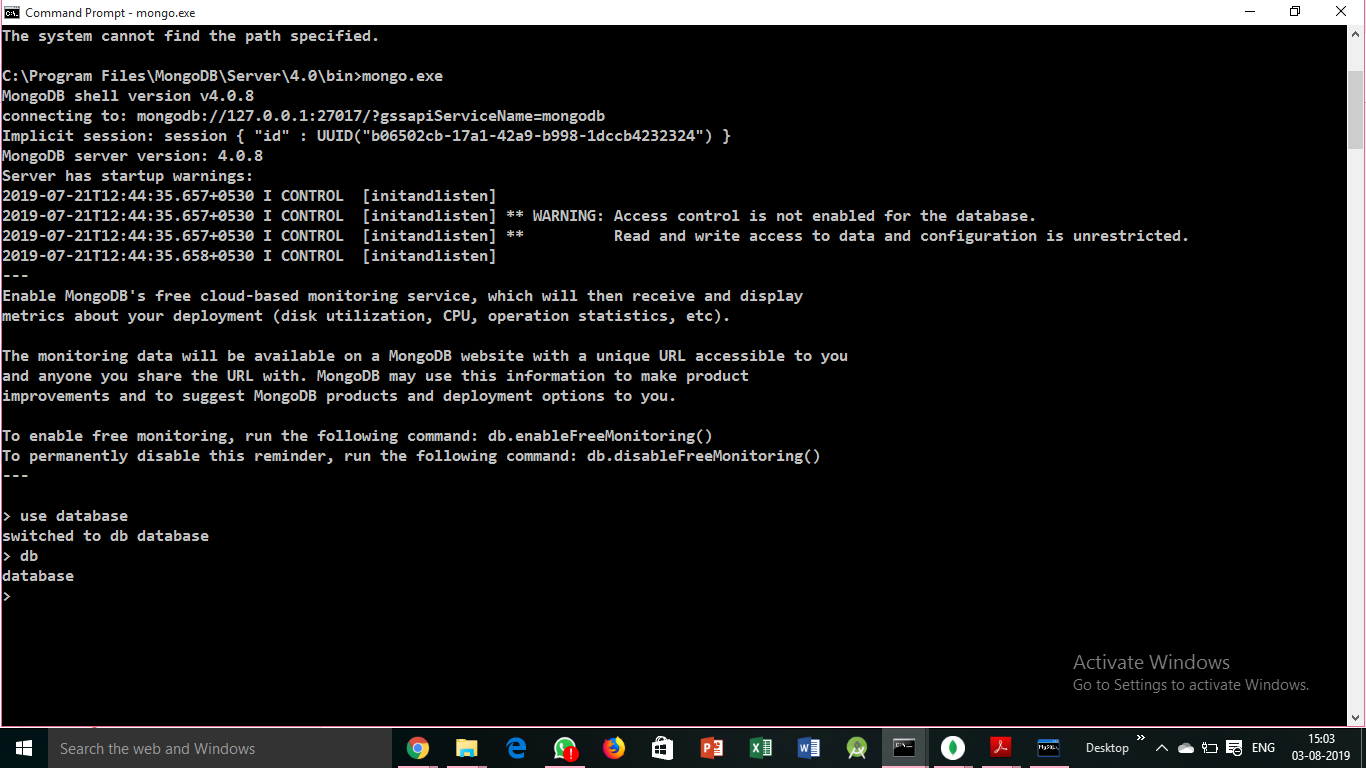
Collection is a group of MongoDB documents. It is the equivalent of an RDBMS table. A collection exists within a single database. Collections do not enforce a schema. Documents within a collection can have different fields. Typically, all documents in a collection are of similar or related purpose.

* Document

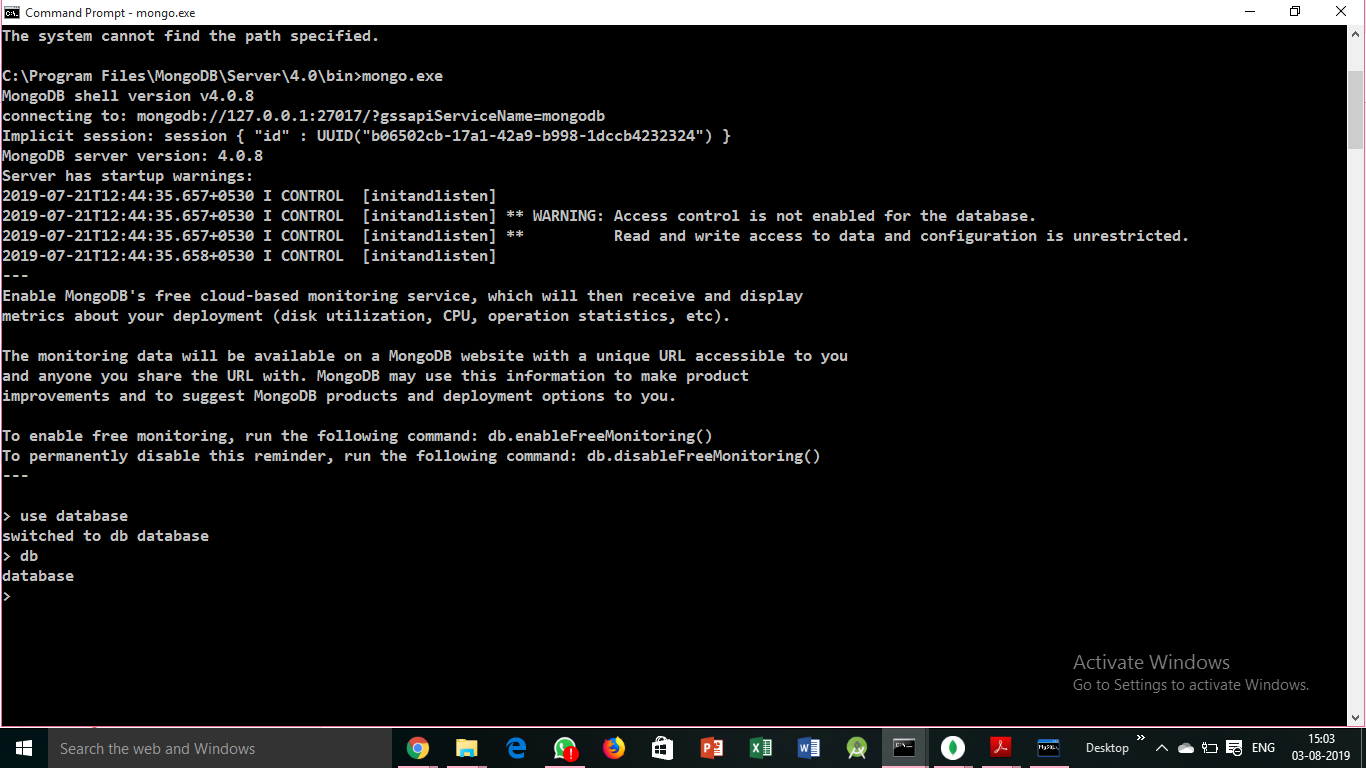
A document is a set of key-value pairs. Documents have dynamic schema. Dynamic schema means that documents in the same collection do not need to have the same set of fields or structure, and common fields in a collection's documents may hold different types of data.

**Queries:**

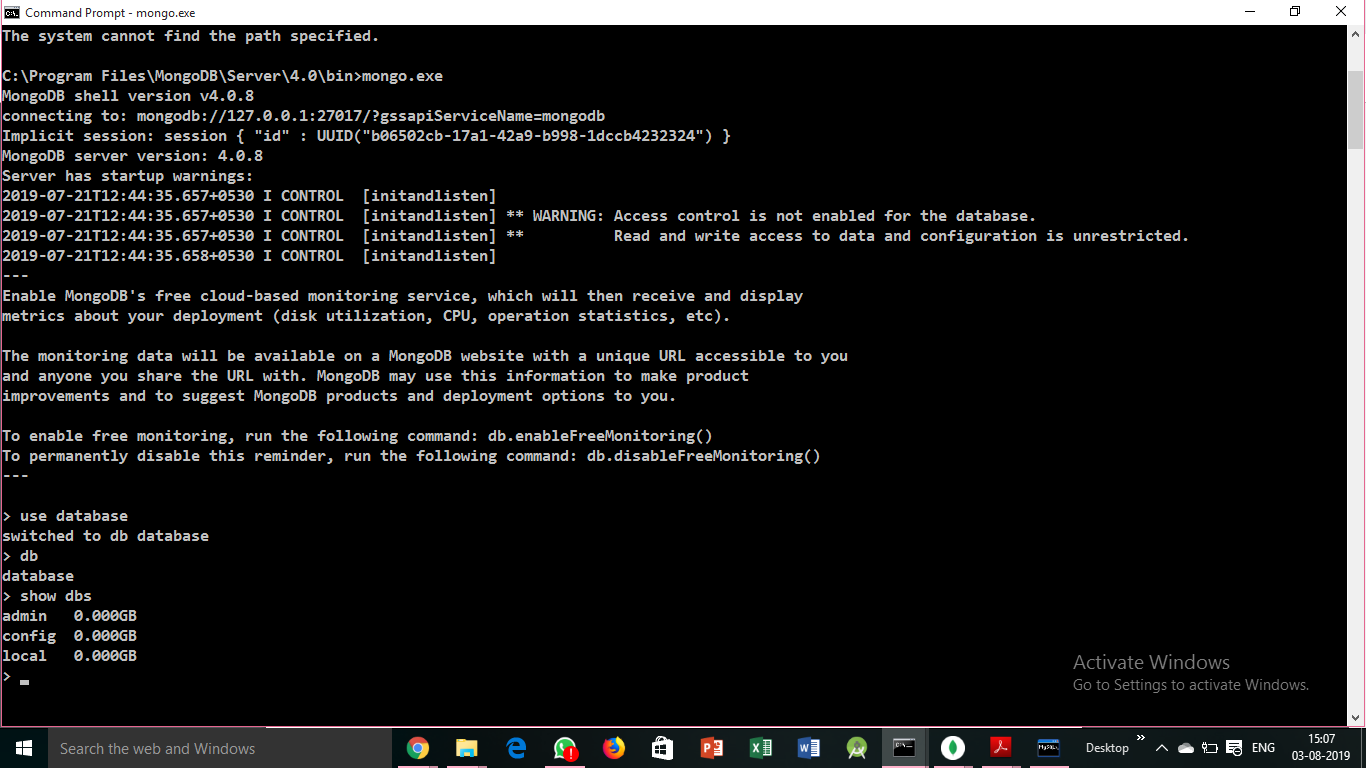
Creation of database



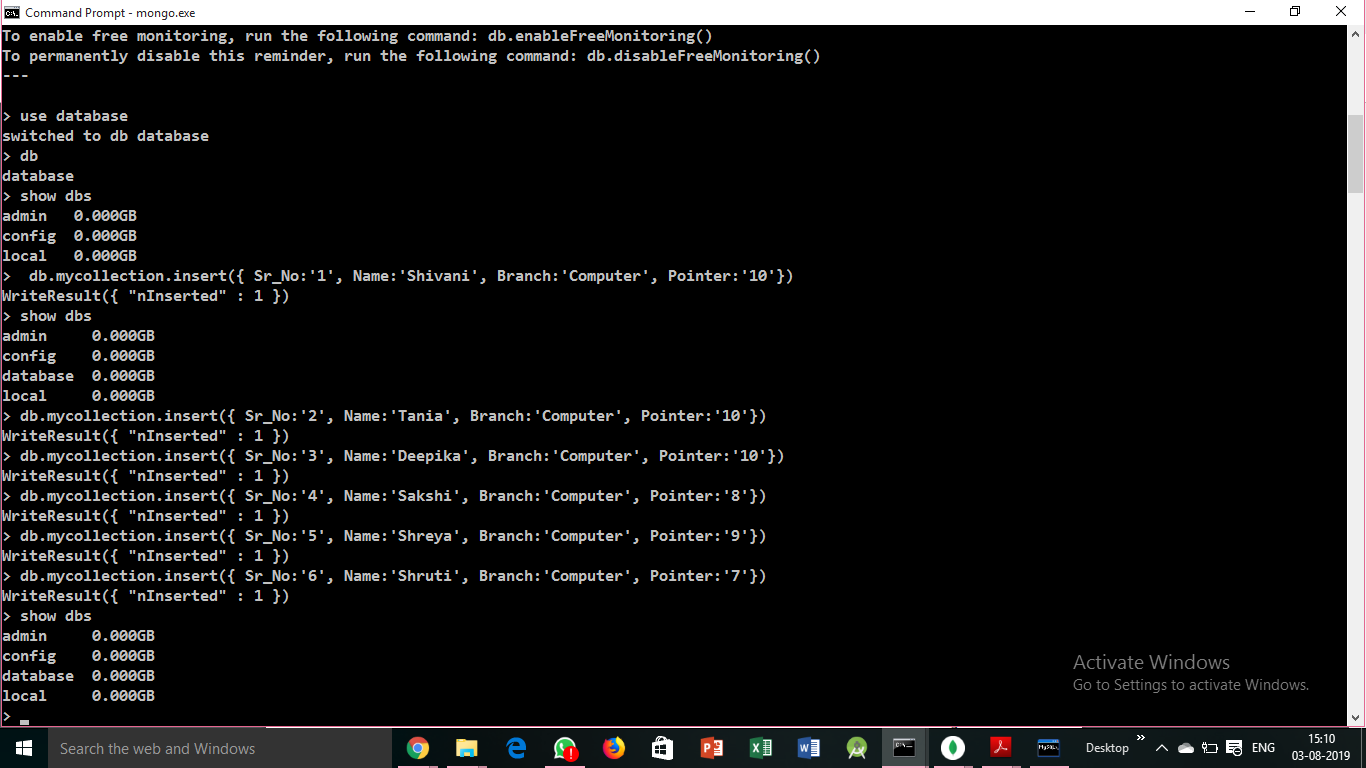
Check currently selected database



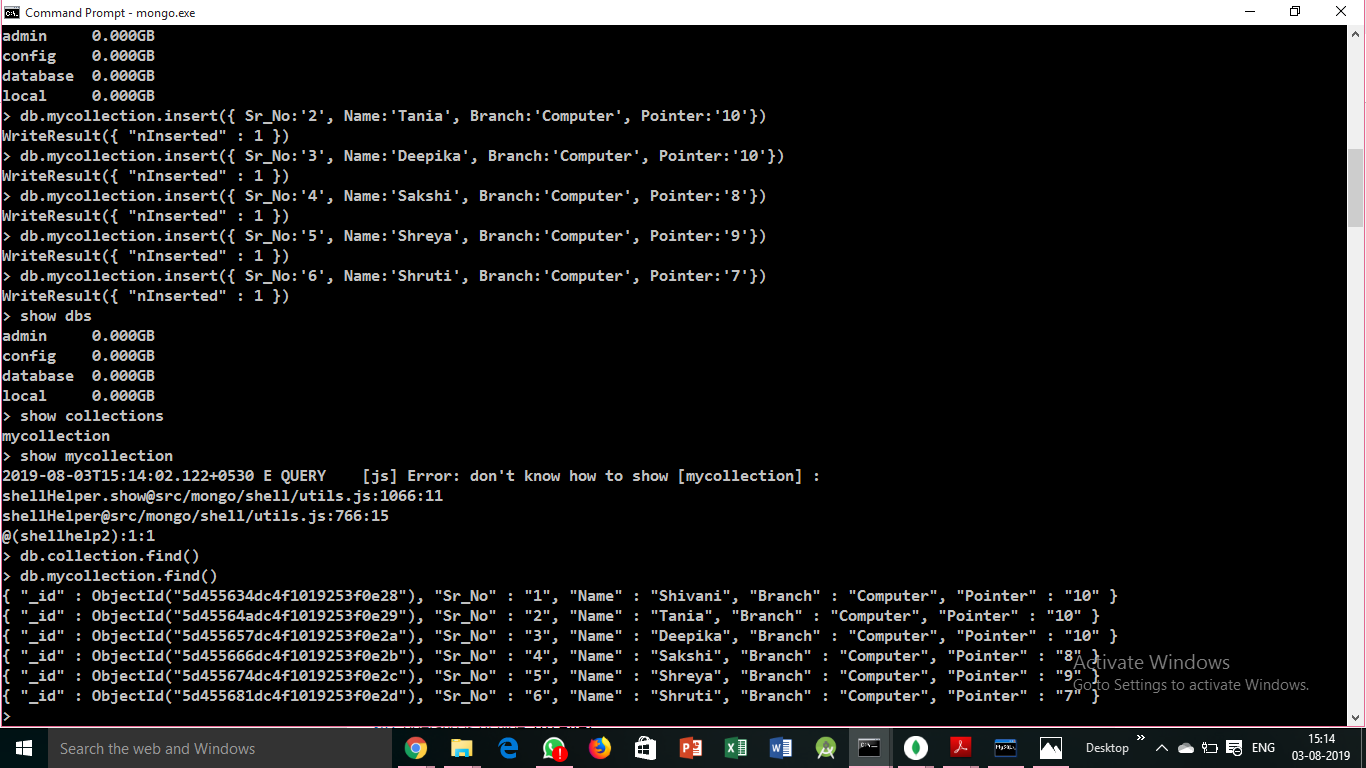
Check database list



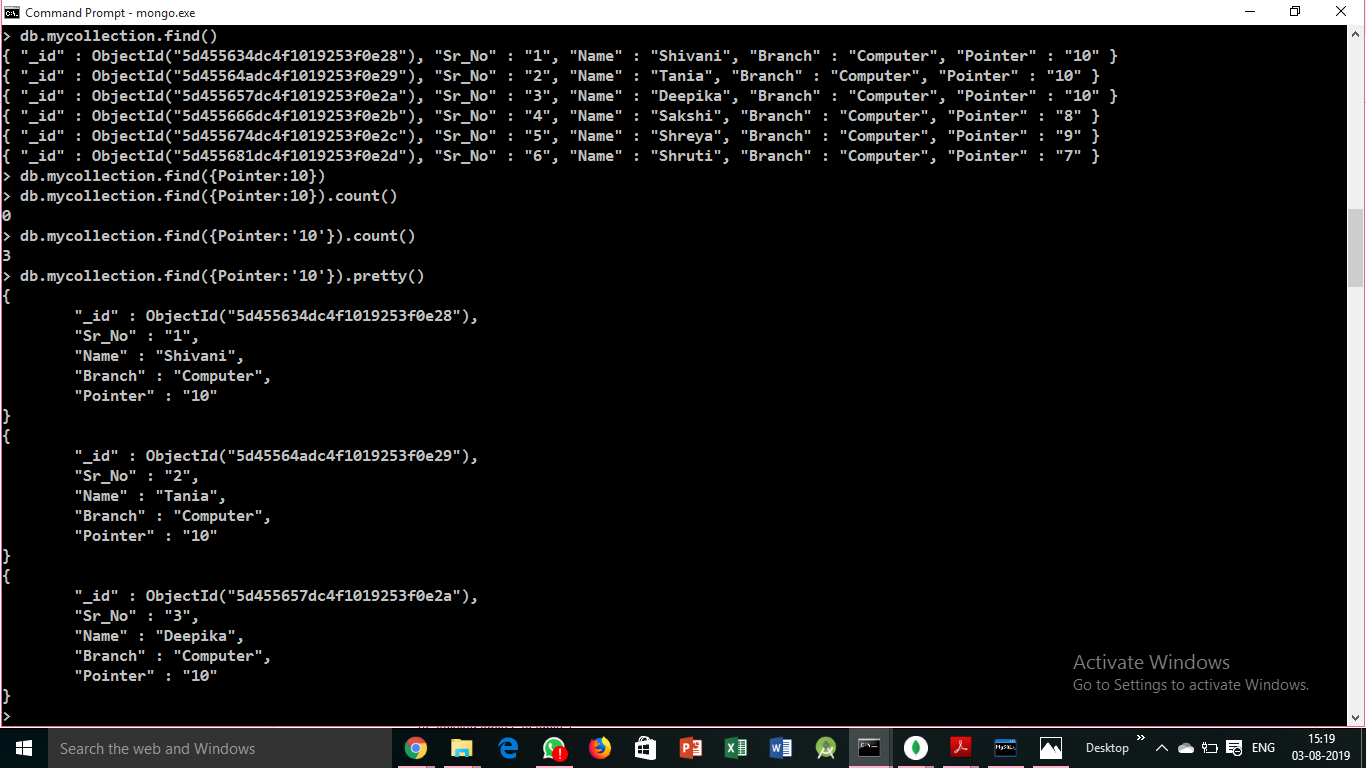
Insertion in database example



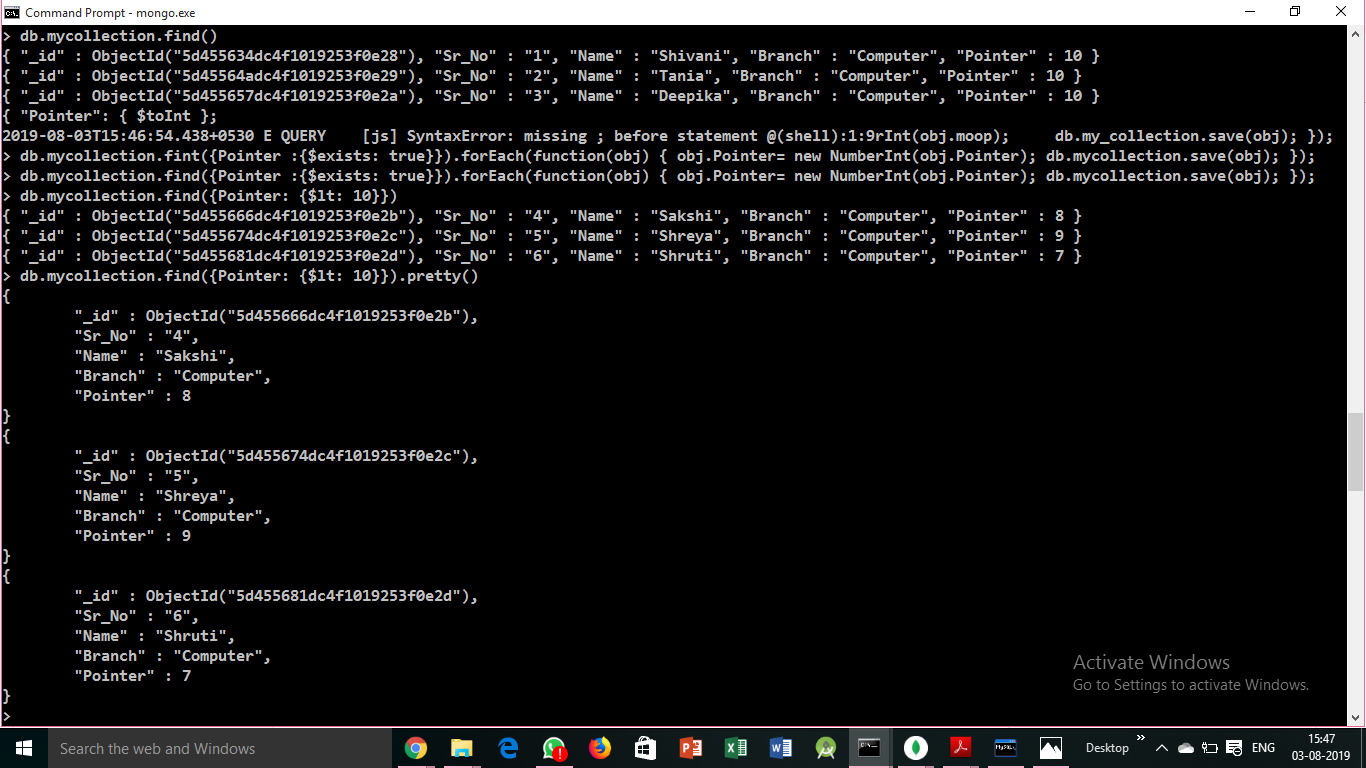
Displaying the info properly



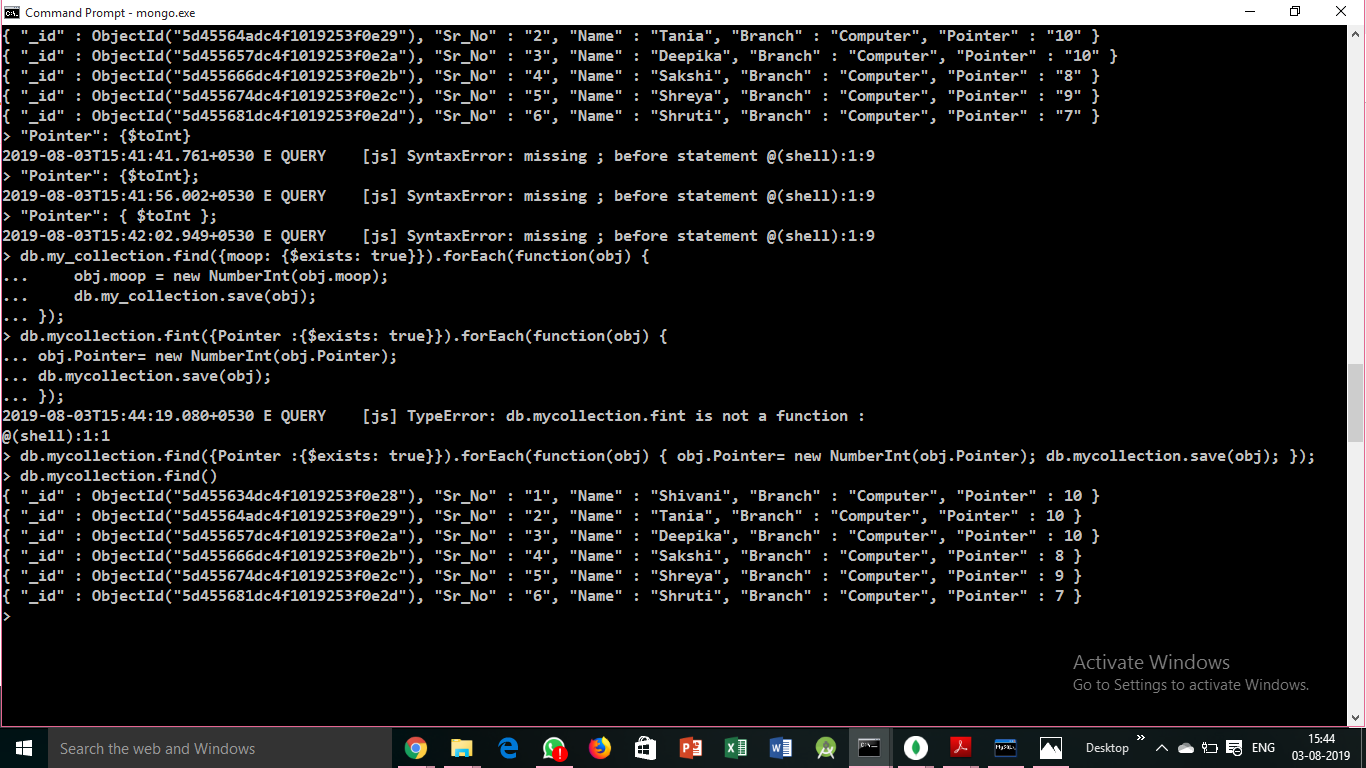
Find students with pointer=10



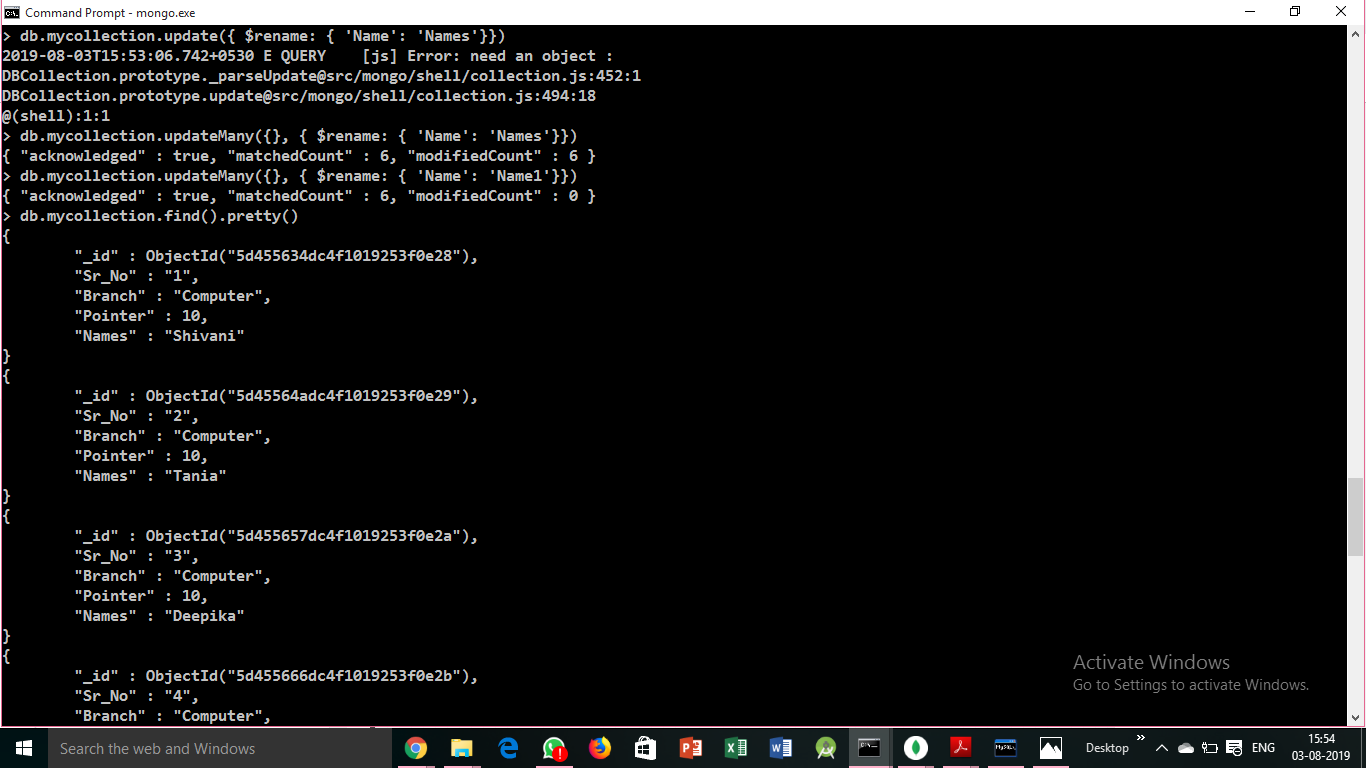
Find students with pointer<10



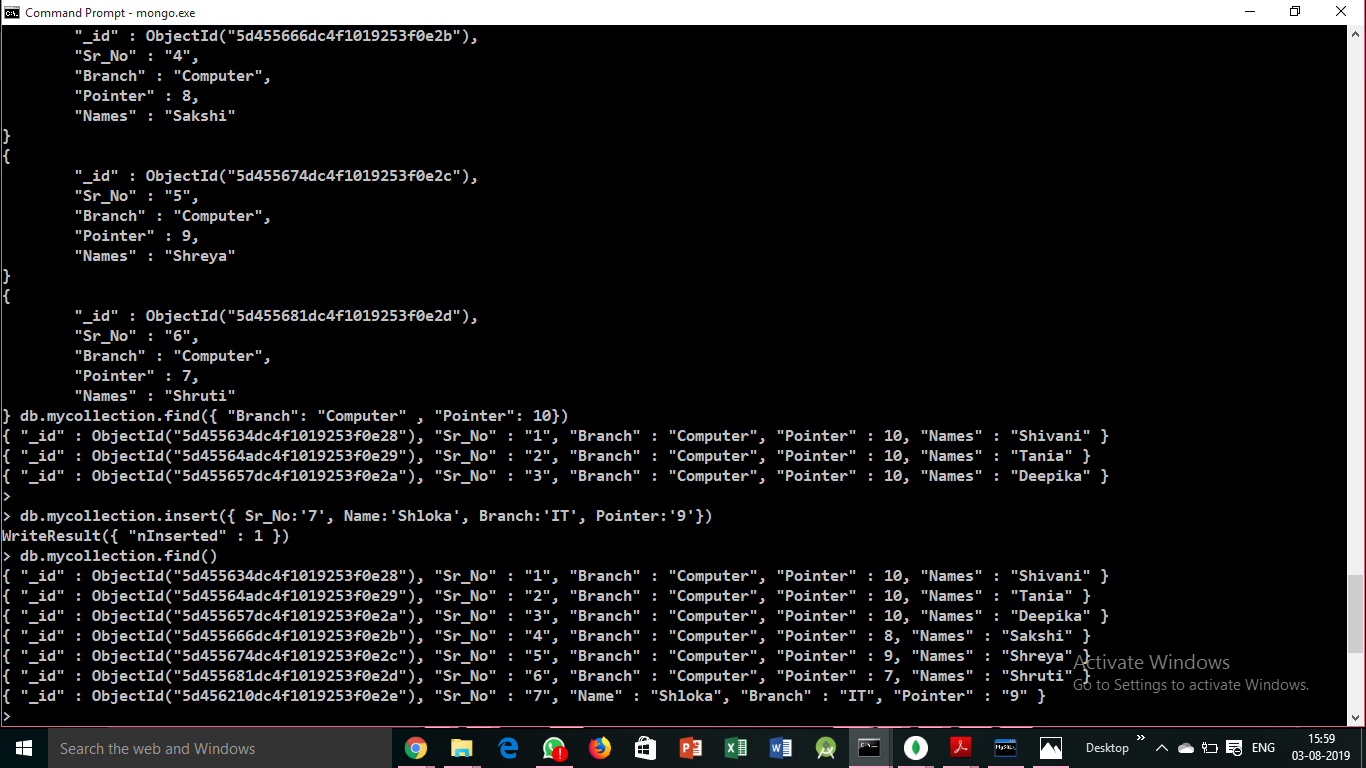
Update a database(Convert from string to integer)



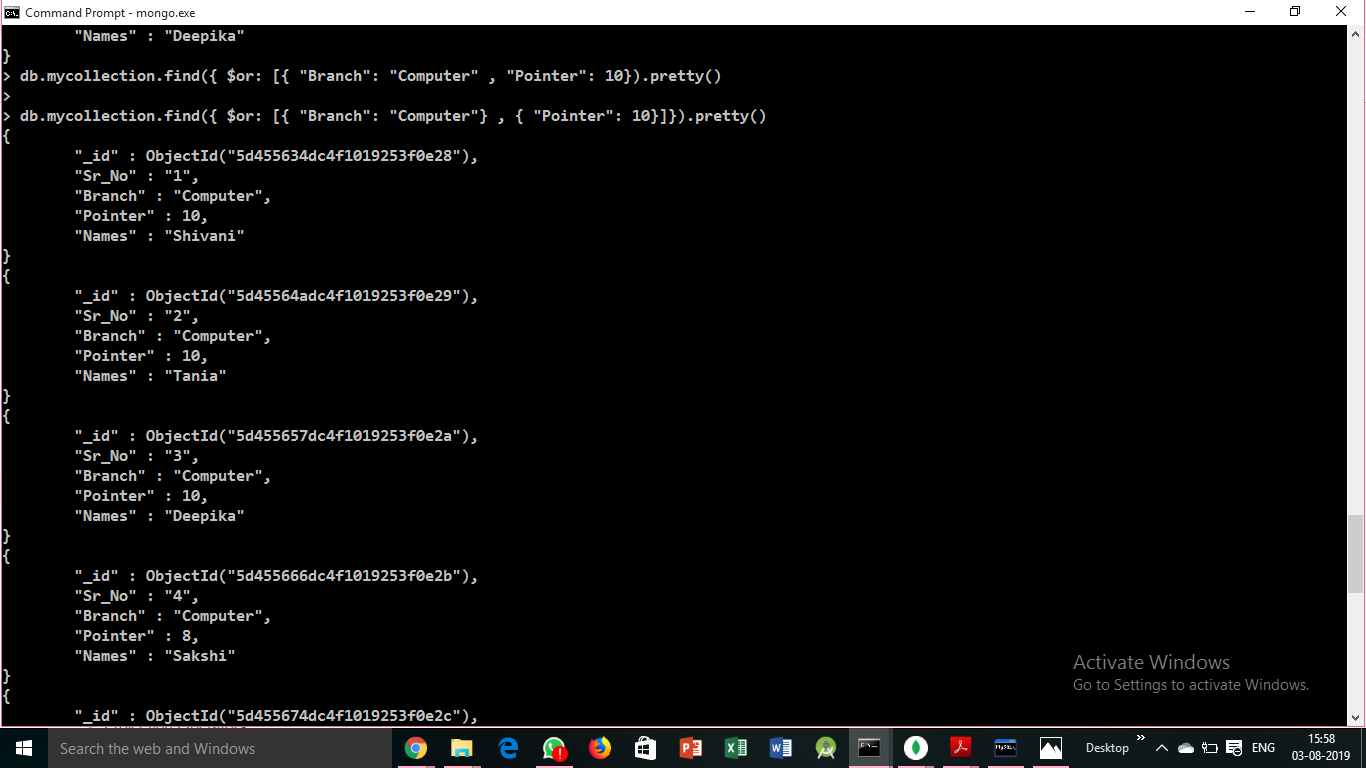
Update a database(Covert name- name1 to name2)



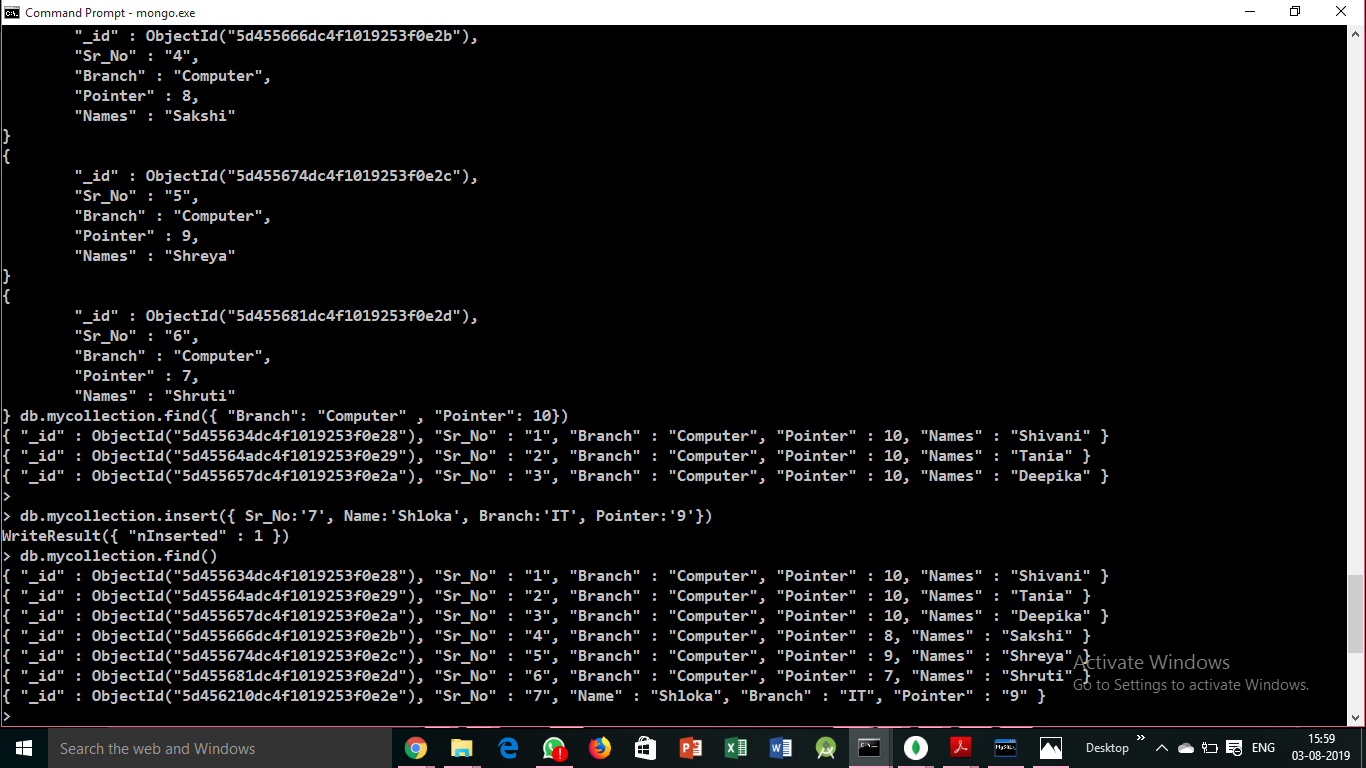
Display records with branch=computer and pointer=10



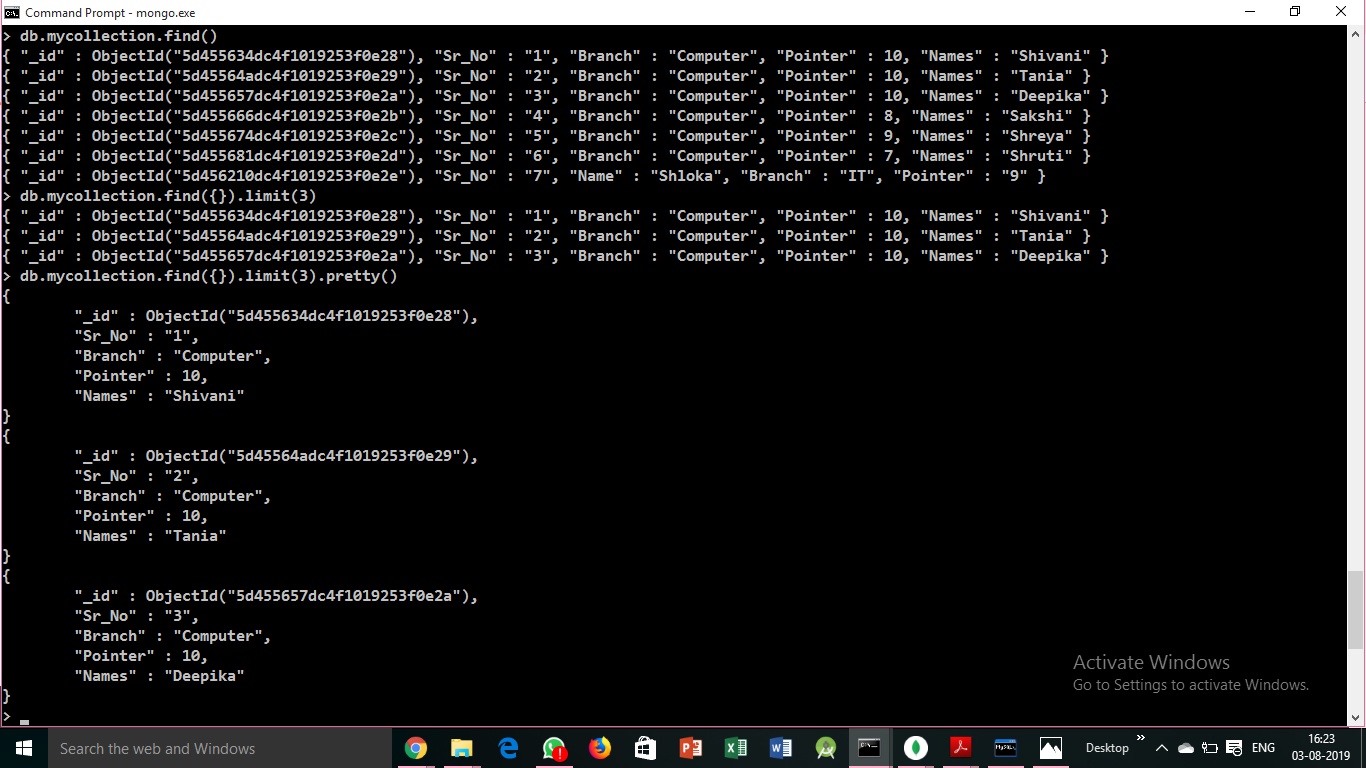
Display records with branch=computer or pointer=10



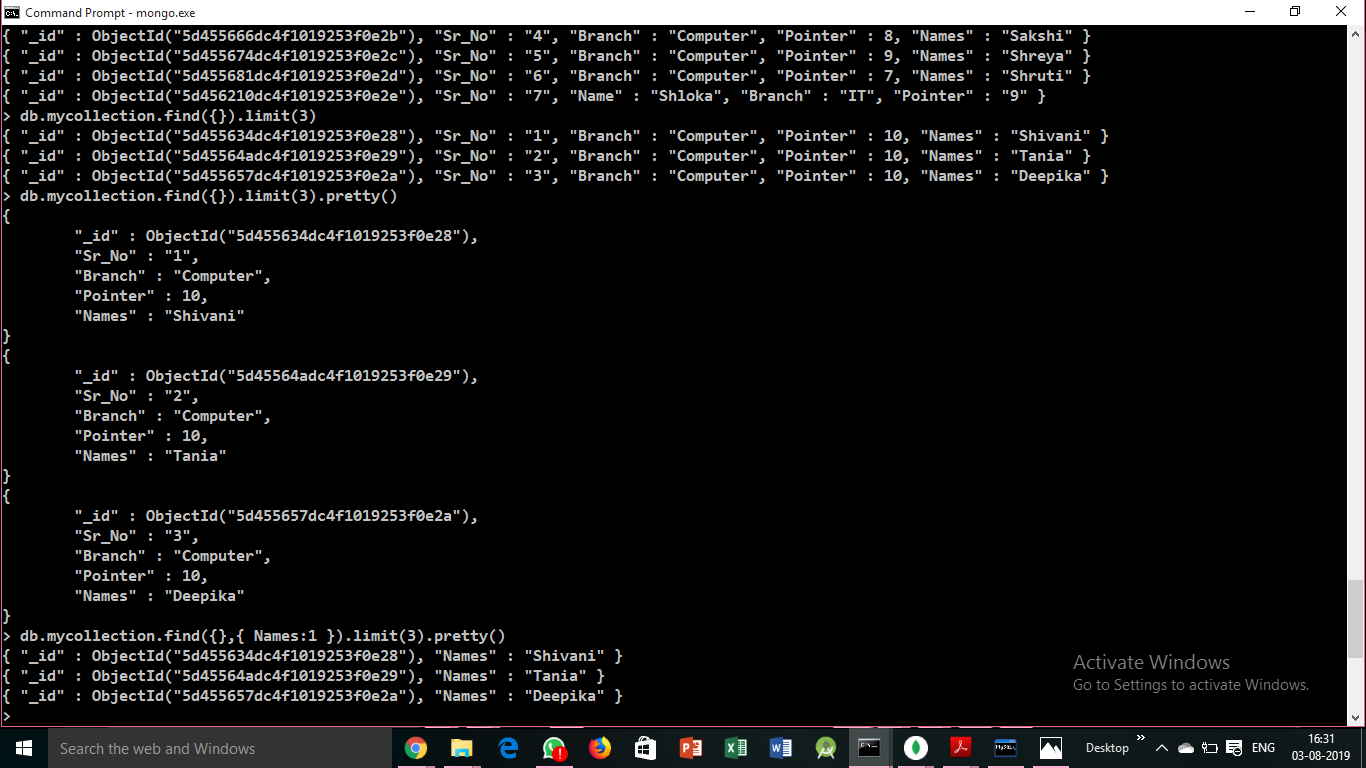
Insert a new record



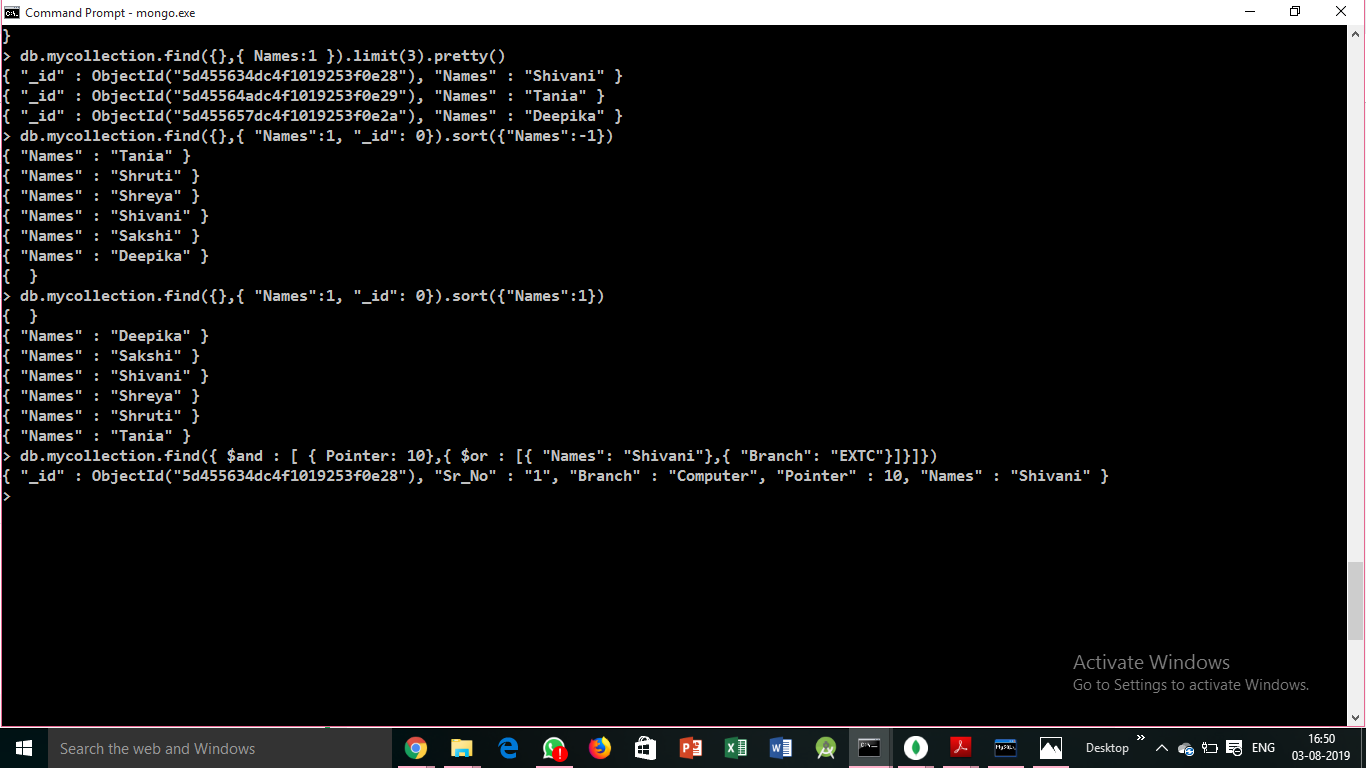
Display first 3 serial numbers



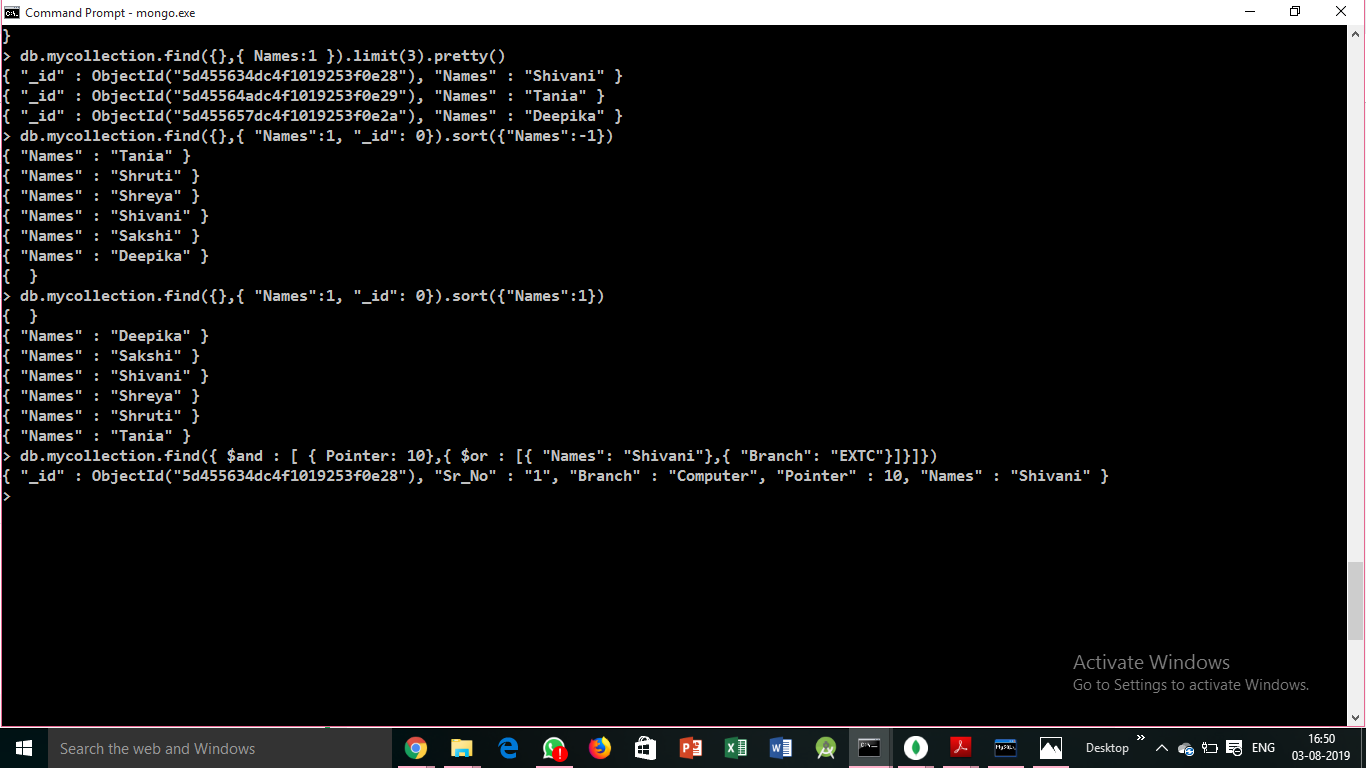
Display first 3 names



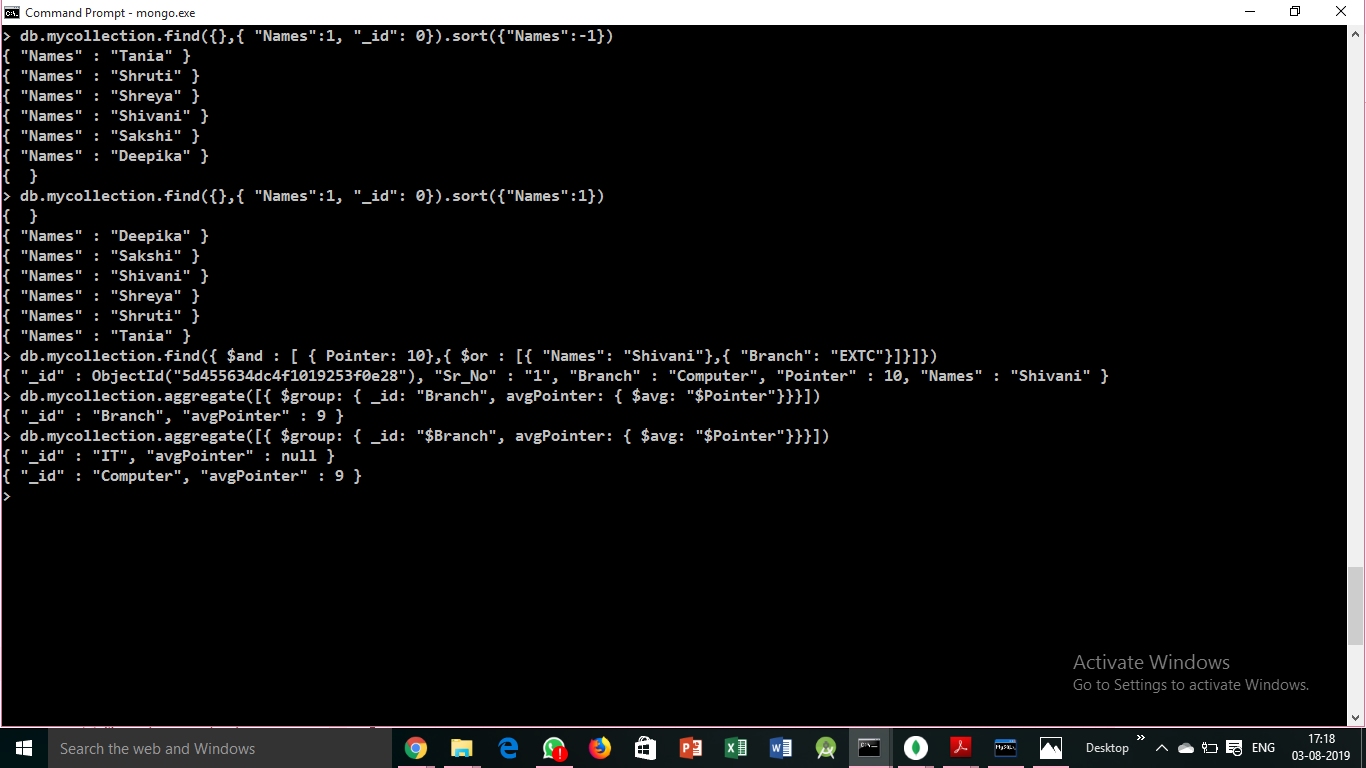
Display names in descending order



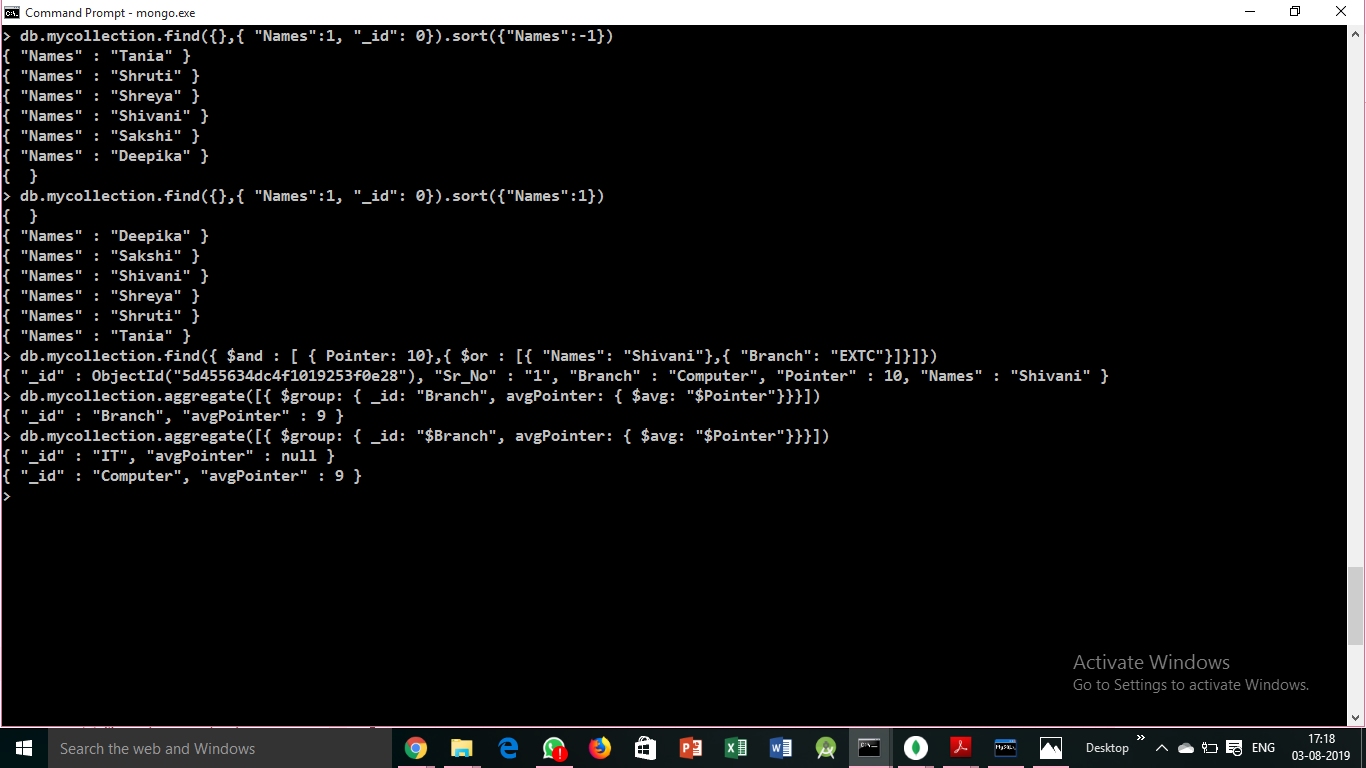
Display names in ascending order



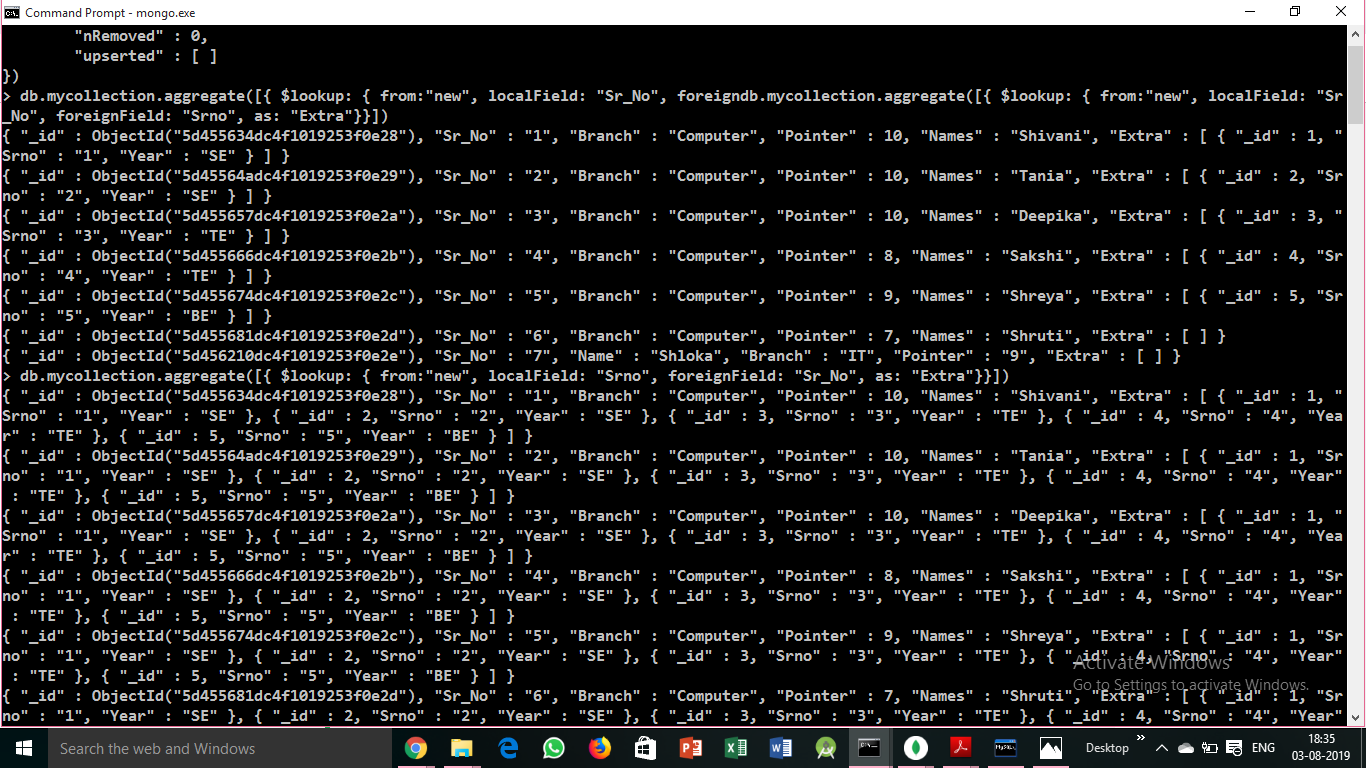
Display records with pointer 10 and(name=shivani or branch=extc)

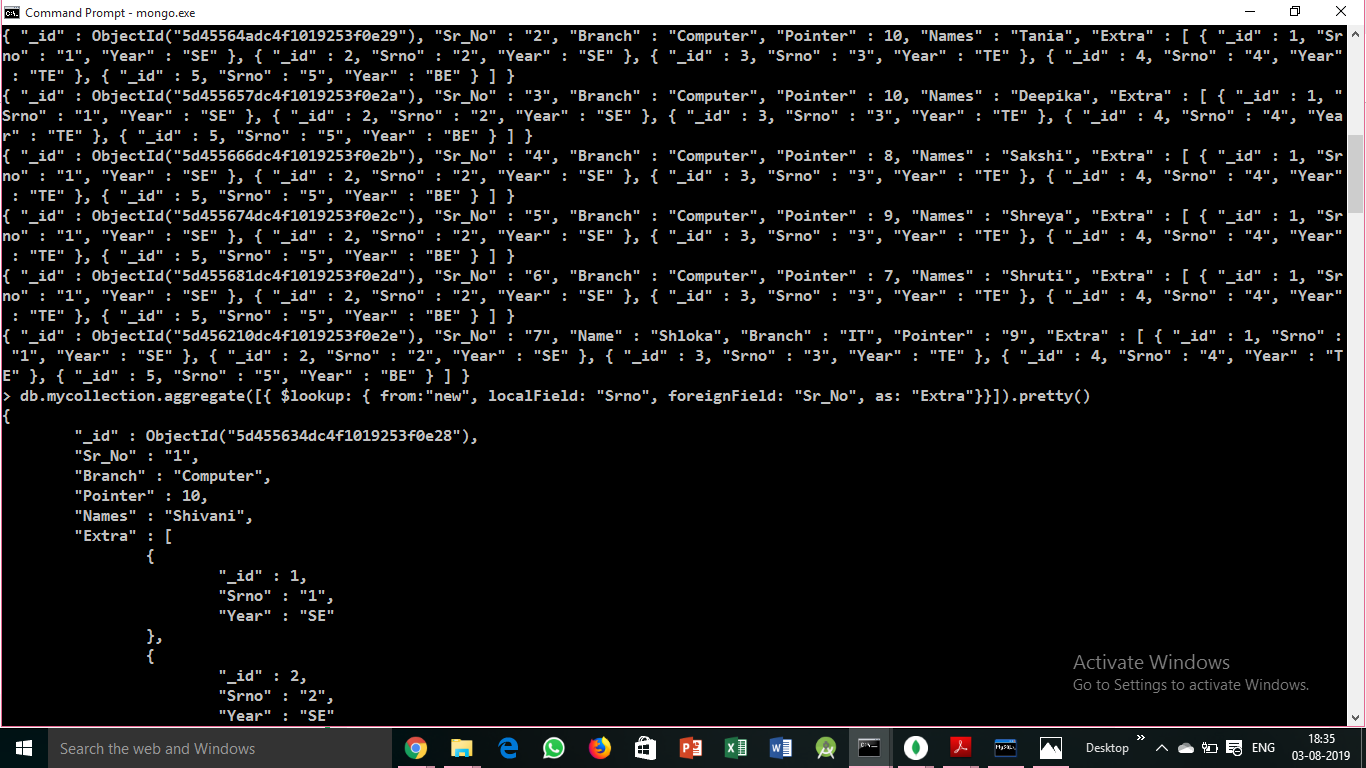


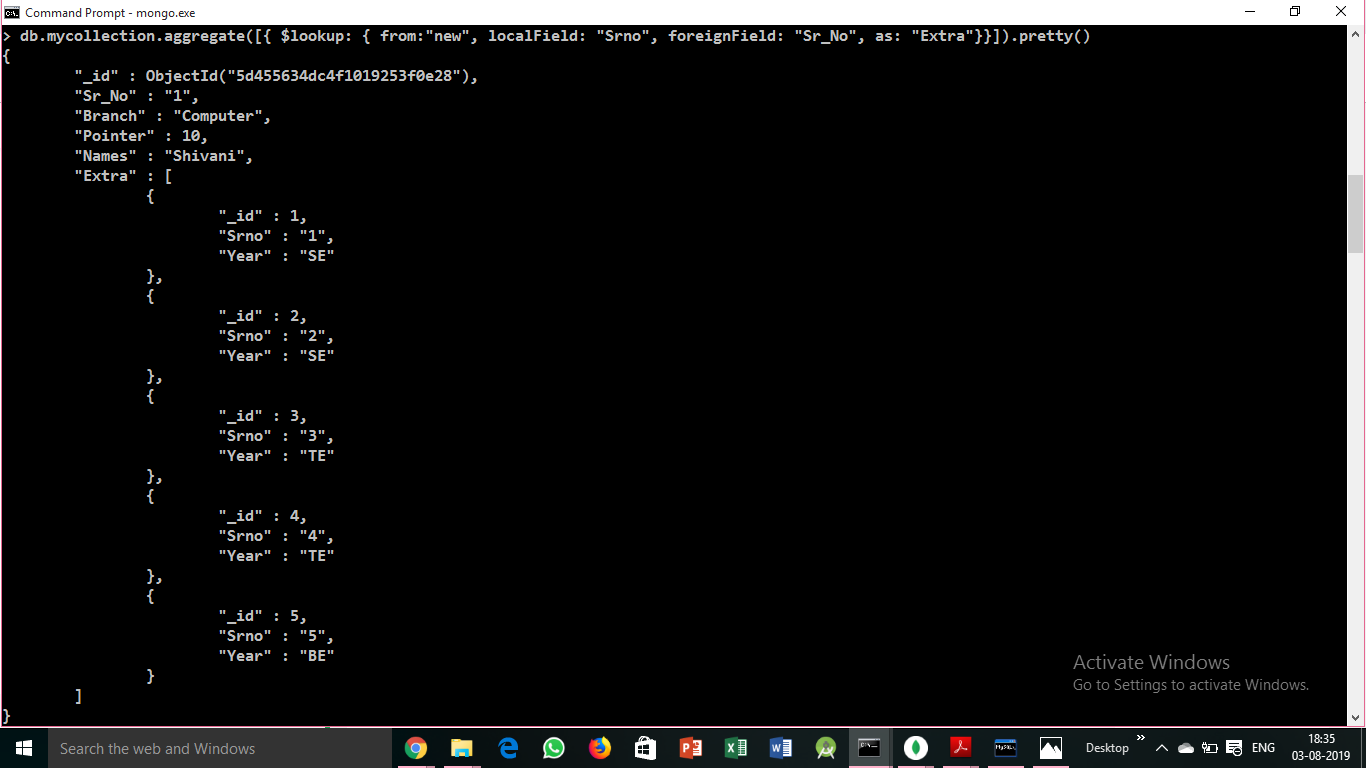
Display avg pointer of each branch



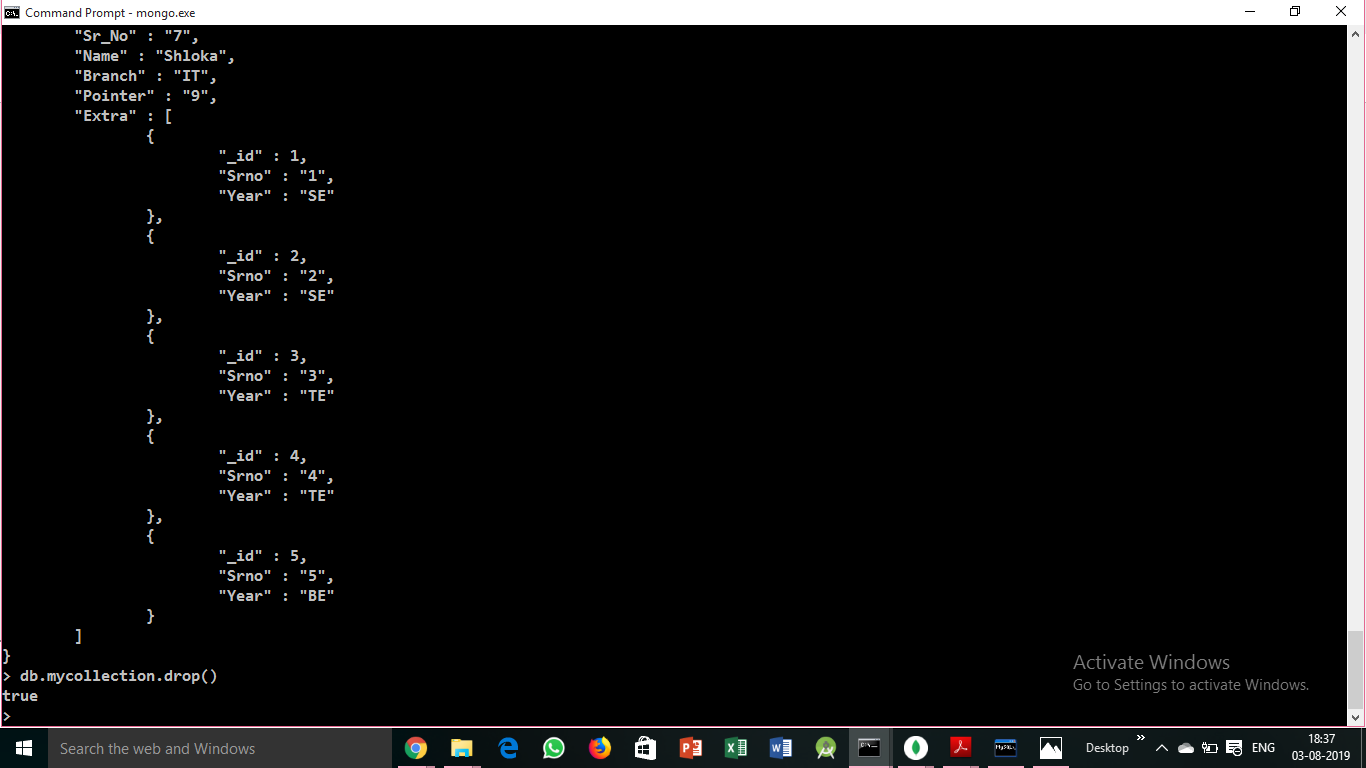
Joining table 2 to table 1







Dropping collection



**Retrieving data from MongoDB through Node.js**

Create database

var url = "mongodb://localhost:27017/newdb";

var MongoClient = require('mongodb').MongoClient;

MongoClient.connect(url, function(err, db) {

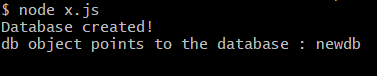
if (err) throw err;

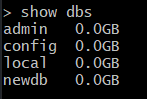
console.log("Database created!");

console.log("db object points to the database : "+ db.databaseName);

db.close();

});

****

****

Create Collection

var MongoClient = require('mongodb').MongoClient;

var url = "mongodb://localhost:27017/";

MongoClient.connect(url, function(err, db) {

if (err) throw err;

var dbo = db.db("newdb");

dbo.createCollection("customers", function(err, res) {

if (err) throw err;

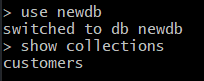
console.log("Collection created!");

db.close();

});

});

****

****

Insert into database

var MongoClient = require('mongodb').MongoClient;

var url = "mongodb://localhost:27017/";

MongoClient.connect(url, function(err, db) {

if (err) throw err;

var dbo = db.db("newdb");

var myobj = { name: "Company Inc", address: "Highway 37" };

dbo.collection("customers").insertOne(myobj, function(err, res) {

if (err) throw err;

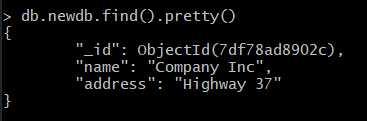
console.log("1 document inserted");

db.close();

});

});

****

****

Finding from database

var MongoClient = require('mongodb').MongoClient;

var url = "mongodb://localhost:27017/";

MongoClient.connect(url, function(err, db) {

if (err) throw err;

var dbo = db.db("newdb");

dbo.collection("customers").findOne({}, function(err, result) {

if (err) throw err;

console.log(result.name);

db.close();

});

});

****

Querying the database

var MongoClient = require('mongodb').MongoClient;

var url = "mongodb://localhost:27017/";

MongoClient.connect(url, function(err, db) {

if (err) throw err;

var dbo = db.db("newdb");

var query = { address: "Highway 37" };

dbo.collection("customers").find(query).toArray(function(err, result) {

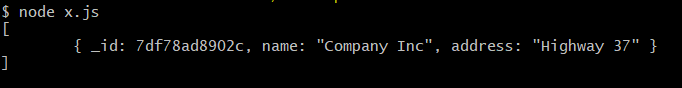
if (err) throw err;

console.log(result);

db.close();

});

});

****

Update database

var MongoClient = require('mongodb').MongoClient;

var url = "mongodb://127.0.0.1:27017/";

MongoClient.connect(url, function(err, db) {

if (err) throw err;

var dbo = db.db("newdb");

var myquery = { address: "Highway 37" };

var newvalues = { $set: {name: "ABC", address: "XYZ" } };

dbo.collection("customers").updateOne(myquery, newvalues, function(err, res) {

if (err) throw err;

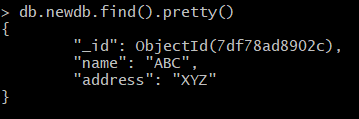
console.log("1 document updated");

db.close();

});

});





Drop collection

var MongoClient = require('mongodb').MongoClient;

var url = "mongodb://localhost:27017/";

MongoClient.connect(url, function(err, db) {

if (err) throw err;

var dbo = db.db("newdb");

dbo.collection("customers").drop(function(err, delOK) {

if (err) throw err;

if (delOK) console.log("Collection deleted");

db.close();

});

});

****

Drop database

var url = "mongodb://localhost:27017/newdb";

var MongoClient = require('mongodb').MongoClient;

MongoClient.connect(url, function(err, db) {

if (err) throw err;

db.dropDatabase(function(err, result){

console.log("Error : "+err);

if (err) throw err;

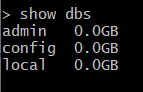
console.log("Database Dropped");

db.close();

});

});

****

****

**Conclusion:**

In this experiment, we have learnt how to write queries in MongoDB and find the required data from the database. We learned how to perform CRUD operations on the database and how to filter the database to get the necessary data.