```
1. Command to display all the databases
cmd> show dbs;
MYDB    0.000GB
admin   0.000GB
config  0.000GB
fsDB    0.000GB
local   0.000GB

2. To create a new logical database.
cmd> use pwskills;
switched to db pwskills

3. To know current logical database.
cmd> db
pwskills

4. To create a collection with one document.
1. insert
2. insertMany
3. insertOne

eg
cmd> db.customer.insertOne({cno:10,cname:"sachin",caddr:"IND",billAmount:55000.0})
{
        "acknowledged" : true,
        "insertedId" : ObjectId("64f9e5b4f766e46c81c312c6")
}

cmd> db.customer.insertOne({cno:18,cname:"kohli"})
{
        "acknowledged" : true,
        "insertedId" : ObjectId("64f9e608f766e46c81c312c7")
}

cmd> db.customer.insertOne({cno:7,cname:"dhoni",mobileNo:9945345412})
{
        "acknowledged" : true,
        "insertedId" : ObjectId("64f9e62af766e46c81c312c8")
}

5. Inserting many documents (insertMany)
eg
cmd> db.customer.insertMany([{cno:9,cname:'lara',caddr:'WI'},
{cno:14,cname:'ponting'}])
{
        "acknowledged" : true,
        "insertedIds" : [
                ObjectId("64f9e7baf766e46c81c312c9"),
                ObjectId("64f9e7baf766e46c81c312ca")
        ]
}

6. To list out documents of the collection(db.customer().find())
eg
cmd> db.customer.find().pretty()
{
        "_id" : ObjectId("64f9e5b4f766e46c81c312c6"),
        "cno" : 10,
        "cname" : "sachin",
```

```
        "caddr" : "IND",
        "billAmount" : 55000
}
{
        "_id" : ObjectId("64f9e608f766e46c81c312c7"),
        "cno" : 18,
        "cname" : "kohli"
}
{
        "_id" : ObjectId("64f9e62af766e46c81c312c8"),
        "cno" : 7,
        "cname" : "dhoni",
        "mobileNo" : 9945345412
}
{
        "_id" : ObjectId("64f9e7baf766e46c81c312c9"),
        "cno" : 9,
        "cname" : "lara",
        "caddr" : "WI"
}
{
        "_id" : ObjectId("64f9e7baf766e46c81c312ca"),
        "cno" : 14,
        "cname" : "ponting"
}
```

7. To retrieve the record based on condition from collection

```
cmd > db.customer.find({caddr:"IND"}).pretty()
{
        "_id" : ObjectId("64f9e5b4f766e46c81c312c6"),
        "cno" : 10,
        "cname" : "sachin",
        "caddr" : "IND",
        "billAmount" : 55000
}

cmd > db.customer.find({cname:"ABD",cno:19}).pretty()
{
        "_id" : ObjectId("64f9ea32f766e46c81c312cb"),
        "cno" : 19,
        "cname" : "ABD",
        "mobileNo" : [
                9998887776,
                7776665554
        ]
}
```

8. Inserting document to a collection with array values.

```
cmd>db.customer.insertOne({cno:19,cname:"ABD",mobileNo:[9998887776,7776665554]})
{
        "acknowledged" : true,
        "insertedId" : ObjectId("64f9ea32f766e46c81c312cb")
}
```

9. To delete a document from a collection we use

```
cmd > db.customer.remove({cno:19})
    WriteResult({ "nRemoved" : 1 })
cmd > db.customer.find({cno:19}).pretty()
```

```
2.
> db.customer.deleteOne({cno:7})
      { "acknowledged" : true, "deletedCount" : 1 }
> db.customer.deleteMany({cno:18})
      { "acknowledged" : true, "deletedCount" : 1 }
> db.customer.find().pretty()
{
        "_id" : ObjectId("64f9e5b4f766e46c81c312c6"),
        "cno" : 10,
        "cname" : "sachin",
        "caddr" : "IND",
        "billAmount" : 55000
}
{
        "_id" : ObjectId("64f9e7baf766e46c81c312c9"),
        "cno" : 9,
        "cname" : "lara",
        "caddr" : "WI"
}
{
        "_id" : ObjectId("64f9e7baf766e46c81c312ca"),
        "cno" : 14,
        "cname" : "ponting"
}

10. To remove collection
cmd > db.customer.drop()
true

11. To list all the collection
cmd > show collections

12. To switch to other logical database
cmd > use IneuronDB

13. To update the collection we need to use the following command.
> db.customer.update({"caddr":"MI"},{$set:{"caddr":"IND"}})
  WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
> db.customer.find().pretty()
{
        "_id" : ObjectId("64f9ef48f766e46c81c312cc"),
        "cno" : 10,
        "cname" : "sachin",
        "caddr" : "IND"
}
{
        "_id" : ObjectId("64f9ef48f766e46c81c312cd"),
        "cno" : 7,
        "cname" : "dhoni",
        "caddr" : "CSK"
}
{
        "_id" : ObjectId("64f9ef48f766e46c81c312ce"),
        "cno" : 18,
        "cname" : "kohli",
        "caddr" : "RCB"
}

Note: _id key holds dynamically generated unique value as "PK_VALUE".
```

```
++++++++++++++++++++++++++++++++
Working with GUI tools for MongoDB
++++++++++++++++++++++++++++++++
a. compass(bit heavy weight)
b. robo3t(light weight)

Working with robo3t
++++++++++++++++++
 => Download from the following link
      https://robo-3t.software.informer.com/download/
 => Install the software just like another window software(just double click
install)


++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
Procedure to create a logical db with collection in MongoDB using Robo3T
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
  a. launch robo3t and connect to mongodb by creating a new connection
  b. use the existing database called "pwskills".
  c. use the existing collection called "customer"
  d. right click on customer collection and then run the query to see the document.

Steps in Robo3T to perform insert operation
      a. right click on customer document
      b. click on insert document
      c. {
            cno: 7,
            cname: "dhoni",
            caddr: "CSK"
         }
      d. perform save operation

Steps in Robo3T to perform update operation
      a. right click on customer document
      b. click on update document
      db.getCollection('customer').update(
            // query
            {
            cno:7
            },

            // update
            { $set:{caddr:"IND"} }

      );
Note: perform other operations like
      a. remove document
      b. rename collection
      c. drop collection

++++++++++++++++++++++++++++++++++++++++++++++
Performing authentication to MongoDB using Robo3T
++++++++++++++++++++++++++++++++++++++++++++++
 Expand logical database(pwskills) ---> users ---> addUser -->
      username : root
      password : root123
      select read,readwrite checkboxes
```

Disconnect from mongodb --> connect ---> delete the existing connection --> create a new connection
	name : con1
	click on authentication tab
		dbname : pwskills
		username: root
		password: root123
	click on connect

++
CustomerRelationshipManager(CRM) using SpringBoot and MongoDB
++

Customer.java
++++++++++++
```
package in.pwskills.nitin.document;

import org.springframework.data.annotation.Id;
import org.springframework.data.mongodb.core.mapping.Document;

@Document
public class Customer {

	@Id
	private String cid;
	private Integer cno;
	private String cname;
	private String caddr;
	private Double billAmount;
	private Long mobileNo;

	public String getCid() {
		return cid;
	}

	public void setCid(String cid) {
		this.cid = cid;
	}

	public Integer getCno() {
		return cno;
	}

	public void setCno(Integer cno) {
		this.cno = cno;
	}

	public String getCname() {
		return cname;
	}

	public void setCname(String cname) {
		this.cname = cname;
	}

	public String getCaddr() {
		return caddr;
	}
```

```
	public void setCaddr(String caddr) {
		this.caddr = caddr;
	}

	public Double getBillAmount() {
		return billAmount;
	}

	public void setBillAmount(Double billAmount) {
		this.billAmount = billAmount;
	}

	public Long getMobileNo() {
		return mobileNo;
	}

	public void setMobileNo(Long mobileNo) {
		this.mobileNo = mobileNo;
	}

	@Override
	public String toString() {
		return "Customer [cid=" + cid + ", cno=" + cno + ", cname=" + cname +
", caddr=" + caddr + ", billAmount="
				+ billAmount + ", mobileNo=" + mobileNo + "]";
	}

}
```

ICustomerRepo.java

```
++++++++++++++++++
package in.pwskills.nitin.repository;

import java.util.List;
import org.springframework.data.mongodb.repository.MongoRepository;
import in.pwskills.nitin.document.Customer;

public interface ICustomerRepo extends MongoRepository<Customer, String> {

	// syntax: <ReturnType> findBy<properties><Condition>(parameters...)
	public List<Customer> findByBillAmountBetween(double start, double end);

	// syntax: <ReturnType> findBy<properties><Condition>(parameters...)
	public List<Customer> findByCaddrInAndMobileNoIsNotNull(String... address);

}
```

MogoDBTestRunner.java

```
++++++++++++++++++++
package in.pwskills.nitin.runner;

import java.io.BufferedReader;
import java.io.InputStreamReader;
import java.util.Scanner;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.CommandLineRunner;
```

```
import org.springframework.stereotype.Component;

import in.pwskills.nitin.document.Customer;
import in.pwskills.nitin.generator.IdGenerator;
import in.pwskills.nitin.service.ICustomerService;

@Component
public class MongoDbTestRunner implements CommandLineRunner {

	@Autowired
	private ICustomerService service;

	@Override
	public void run(String... args) throws Exception {

		BufferedReader reader = new BufferedReader(new
InputStreamReader(System.in));
		while (true) {
			System.out.println("WELCOME TO CUSTOMER RELATIONSHIP PROJECT");
			System.out.println("1. CREATE");
			System.out.println("2. READ");
			System.out.println("3. UPDATE");
			System.out.println("4. DELETE");
			System.out.println("5. EXIT");

			System.out.print("Enter ur choice :: ");
			String choice = reader.readLine();

			switch (choice) {
			case "1":
				insertDocument();
				break;

			case "2":
				selectDocument();
				break;

			case "3":
				updateDocument();
				break;

			case "4":
				deleteDocument();
				break;

			case "5":
				System.out.println("Thanks for using the application");
				System.exit(0);

			default:
				System.out.println("Invalid choice.. PLZ try again...");
				break;
			}
		}

	}

	@SuppressWarnings("unused")
	private void findCustomersByAddressWhoseMobileNoIsNotNull() {
```

```
            service.fetchCustomerByUsingCaddrAndHavingMobileNot("RCB", "CSK",
"MI").forEach(System.out::println);
      }

      @SuppressWarnings("unused")
      private void findCustomersByBillRange() {
            service.fetchCustomersByBillAmountRange(2000.0,
10000.0).forEach(System.out::println);
      }

      private void deleteDocument() {
            String id = "e851df3c9a";
            System.out.println(service.removeCustomer(id));
      }

      private void updateDocument() {
            Customer customer = new Customer();
            String documentId = "e851df3c9a";
            customer.setCid(documentId);
            customer.setCno(9);
            customer.setCname("lara");
            customer.setCaddr("WI");
            customer.setBillAmount(4500.0);
            customer.setMobileNo(5556667778L);
            System.out.println(service.registerCustomer(customer));
      }

      private void selectDocument() {
            service.findAllCustomers().forEach(System.out::println);
      }

      private void insertDocument() {
            Customer customer1 = new Customer();
            customer1.setCid(IdGenerator.generateId());

            @SuppressWarnings("resource")
            Scanner scanner = new Scanner(System.in);

            System.out.print("Enter the customer id :: ");
            customer1.setCno(scanner.nextInt());
            System.out.print("Enter the customer name :: ");
            customer1.setCname(scanner.next());
            System.out.print("Enter the customer address :: ");
            customer1.setCaddr(scanner.next());
            System.out.print("Enter the customer billAmount:: ");
            customer1.setBillAmount(scanner.nextDouble());
            System.out.print("Enter the customer mobileNo :: ");
            customer1.setMobileNo(scanner.nextLong());

            System.out.println(service.registerCustomer(customer1));
      }

}
```

++++++++++++++++++++++++
CustomerServiceImpl.java
++++++++++++++++++++++++

```
package in.pwskills.nitin.service;
```

```
import java.util.List;
import java.util.Optional;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

import in.pwskills.nitin.document.Customer;
import in.pwskills.nitin.repository.ICustomerRepo;

@Service("customerService")
public class CustomerServiceImpl implements ICustomerService {

	@Autowired
	private ICustomerRepo repo;

	@Override
	public String registerCustomer(Customer customer) {
		// performs both insert/update operation
		return "Customer is saved with the id :: " +
repo.save(customer).getCid();
	}

	@Override
	public List<Customer> findAllCustomers() {
		List<Customer> customers = repo.findAll();
		return customers;
	}

	@Override
	public String removeCustomer(String id) {

		Optional<Customer> optional = repo.findById(id);
		if (optional.isPresent()) {
			repo.delete(optional.get());
			return "Document is deleted";
		}

		return "Document not found";
	}

	@Override
	public List<Customer> fetchCustomersByBillAmountRange(double start, double
end) {
		return repo.findByBillAmountBetween(start, end);
	}

	@Override
	public List<Customer> fetchCustomerByUsingCaddrAndHavingMobileNot(String...
address) {
		return repo.findByCaddrInAndMobileNoIsNotNull(address);
	}
}
```

```
Output
WELCOME TO CUSTOMER RELATIONSHIP PROJECT
1. CREATE
2. READ
3. UPDATE
```

```
4. DELETE
5. EXIT
Enter ur choice :: 1
Enter the customer id :: 7
Enter the customer name :: dhoni
Enter the customer address ::
CSK
Enter the customer billAmount::
35000
Enter the customer mobileNo ::
8767868787
Customer is saved with the id :: ef9c233cff
WELCOME TO CUSTOMER RELATIONSHIP PROJECT
1. CREATE
2. READ
3. UPDATE
4. DELETE
5. EXIT
Enter ur choice :: 1
Enter the customer id :: 18
Enter the customer name :: kohli
Enter the customer address :: RCB
Enter the customer billAmount:: 349089
Enter the customer mobileNo :: 99988877765
Customer is saved with the id :: 382d15728e
WELCOME TO CUSTOMER RELATIONSHIP PROJECT
1. CREATE
2. READ
3. UPDATE
4. DELETE
5. EXIT
Enter ur choice :: 2
Customer [cid=e851df3c9a, cno=10, cname=sachin, caddr=MI, billAmount=45000.0,
mobileNo=8989887665]
Customer [cid=ef9c233cff, cno=7, cname=dhoni, caddr=CSK, billAmount=35000.0,
mobileNo=8767868787]
Customer [cid=382d15728e, cno=18, cname=kohli, caddr=RCB, billAmount=349089.0,
mobileNo=99988877765]
WELCOME TO CUSTOMER RELATIONSHIP PROJECT
1. CREATE
2. READ
3. UPDATE
4. DELETE
5. EXIT
Enter ur choice :: 4
Document is deleted
WELCOME TO CUSTOMER RELATIONSHIP PROJECT
1. CREATE
2. READ
3. UPDATE
4. DELETE
5. EXIT
Enter ur choice :: 2
Customer [cid=ef9c233cff, cno=7, cname=dhoni, caddr=CSK, billAmount=35000.0,
mobileNo=8767868787]
Customer [cid=382d15728e, cno=18, cname=kohli, caddr=RCB, billAmount=349089.0,
mobileNo=99988877765]
WELCOME TO CUSTOMER RELATIONSHIP PROJECT
1. CREATE
```

```
2. READ
3. UPDATE
4. DELETE
5. EXIT
Enter ur choice :: 3
Customer is saved with the id :: e851df3c9a
WELCOME TO CUSTOMER RELATIONSHIP PROJECT
1. CREATE
2. READ
3. UPDATE
4. DELETE
5. EXIT
Enter ur choice :: 2
Customer [cid=ef9c233cff, cno=7, cname=dhoni, caddr=CSK, billAmount=35000.0,
mobileNo=8767868787]
Customer [cid=382d15728e, cno=18, cname=kohli, caddr=RCB, billAmount=349089.0,
mobileNo=99988877765]
Customer [cid=e851df3c9a, cno=9, cname=lara, caddr=WI, billAmount=4500.0,
mobileNo=5556667778]
WELCOME TO CUSTOMER RELATIONSHIP PROJECT
1. CREATE
2. READ
3. UPDATE
4. DELETE
5. EXIT
Enter ur choice :: 5
Thanks for using the application
```