1. <u>Create company Database</u>
   use companydb

2. <u>Create collections</u>

   1. Employee

      db.createcollection("Employee")

   2. Department

      db.createcollection("Department")

3 ⓐ <u>Insert into Employee</u>

   1. db.employee.insert({_id: 1, eName: "John", designation: "Manager", salary: "60,000", yearsOfExperience: 12, dept: "Customer Mangement"})

   2a) db.employee.update({_id: 2, eName: "Bob", designation: "SDE2", salary: "60,000", yearsOfExperience: 4}, {$set: {dept: "Customer management"}}, {upsert: true});

   b) db.employee.update({_id: 3, eName: "Raj", designation: "SDE3", salary: "90,000", yearsOfExperience: 6}, {$set: {dept: "Customer management"}}, {upsert: false})

   3. db.employee.save({_id: 4, eName: "Ram", designation: "SDE1", salary: "45,000", yoE: 1, dept: "life Sciences"}

(4) Update employee collection to add new
field to an existing document

```
db. employee. update (
    { designation: "SDE1"},
    { $set : { salary: "50,000"}}
    { multi:true}
)
```

(5) Remove a field from an existing document

```
db. employee. update (
    {_id: 2 },
    { $unset : { dept: "Customer Management"}});
```

(6) Select all documents from both collections

```
db.employee. aggregate([{ $lookup: {from: "department",
localField: "deptNo", foreignField: "_id", as: "department"}}]);
```

(7) Select only employee name and department number
whose department no. falls between 1001 to 1006

```
db. employee.find({ }, {eName:1, deptNo.: 1,
```

(8) Select employees whose names start from 'A'

```
db.employee.find ({eName: /^A/ }).pretty ();
```

(9) Select employees whose age > 25

```
db. employee. find ({age: {$gt : 25} });
```