Design MongoDB database and perform following Map reduce
operation:
Create Employee collection by considering following Fields:
i. Name: Embedded Doc (FName, LName)
ii. Company Name: String
iii. Salary: Number
iv. Designation: String
v. Age: Number
vi. Expertise: Array
vii. DOB: String or Date
viii. Email id: String
ix. Contact: String
x. Address: Array of Embedded Doc (PAddr, LAddr)
Execute the following query:
1. Display the total salary of per company
2. Display the total salary of company Name:"TCS"
3. Return the average salary of company whose address is “Pune".
4. Display total count for “City=Pune”
5. Return count for city pune and age greater than 40.

```
// Create the "Employee" collection
db.createCollection("Employee");

// Insert sample entries into the "Employee" collection
db.Employee.insert([
    {
        "Name": { "FName": "John", "LName": "Doe" },
        "Company_Name": "Infosys",
        "Salary": 60000,
        "Designation": "Developer",
        "Age": 28,
        "Expertise": ["MongoDB", "SQL"],
        "DOB": "1995-01-15",
        "Email_id": "john.doe@example.com",
        "Contact": "9876543210",
        "Address": [{ "PAddr": "123 Main St", "LAddr": "Apt 45" }]
    },
    {
        "Name": { "FName": "Alice", "LName": "Smith" },
        "Company_Name": "TCS",
        "Salary": 35000,
        "Designation": "Tester",
        "Age": 25,
        "Expertise": ["Testing", "Automation"],
        "DOB": "1998-05-22",
        "Email_id": "alice.smith@example.com",
        "Contact": "8765432109",
        "Address": [{ "PAddr": "456 Oak St", "LAddr": "Suite 12" }]
    },
    // Insert three more documents
    // ...

    // Map-Reduce Operations:
```

```
    // Query 1: Display the total salary per company
    db.Employee.mapReduce(
        function () {
            emit(this.Company_Name, this.Salary);
        },
        function (key, values) {
            return Array.sum(values);
        },
        { out: "TotalSalaryPerCompany", finalize: null }
    );

    // Query 2: Display the total salary of company Name: "TCS"
    db.Employee.mapReduce(
        function () {
            if (this.Company_Name === "TCS") {
                emit(this.Company_Name, this.Salary);
            }
        },
        function (key, values) {
            return Array.sum(values);
        },
        { out: "TotalSalaryTCS", finalize: null }
    );

    // Query 3: Return the average salary of the company whose address is
“Pune".
    db.Employee.mapReduce(
        function () {
            this.Address.forEach(function (addr) {
                if (addr.PAddr === "Pune") {
                    emit(this.Company_Name, this.Salary);
                }
            });
        },
        function (key, values) {
            return Array.avg(values);
        },
        { out: "AverageSalaryPune", finalize: null }
    );

    // Query 4: Display total count for “City=Pune”
    db.Employee.mapReduce(
        function () {
            this.Address.forEach(function (addr) {
                if (addr.PAddr === "Pune") {
                    emit("City=Pune", 1);
                }
            });
        },
        function (key, values) {
            return Array.sum(values);
        },
        { out: "TotalCountCityPune", finalize: null }
    );

    // Query 5: Return count for city Pune and age greater than 40.
db.Employee.mapReduce(
    function () {
        // Check if there is an address in Pune and age > 40
        if (this.Address[0].PAddr === "Pune" && this.Age > 40) {
            emit("City=Pune, Age>40", 1);
        }
    },
    function (key, values) {
        return Array.sum(values);
    },
    { out: "CountCityPuneAge40Plus_NoForEach_MR", finalize: null }
);
]);
```