

# WEEKLY ASSIGNMENT 2

**NAME: BARANIDHARAN S**

**REG.NO.: 20BCE0044**

**COURSE: MODERN APPLICATION DEVELOPMENT JAVA SPRING BOOT**

## 1. CREATE, UPDATE, DELETE COMMANDS IN MYSQL

### CREATE COMMAND

```
CREATE TABLE EMPLOYEE (  
    empId INTEGER PRIMARY KEY,  
    name TEXT NOT NULL,  
    dept TEXT NOT NULL  
);
```

### INSERT COMMAND

```
INSERT INTO EMPLOYEE VALUES (0001, 'Clark', 'Sales');  
INSERT INTO EMPLOYEE VALUES (0002, 'Dave', 'Accounting');  
INSERT INTO EMPLOYEE VALUES (0003, 'Ava', 'Sales');
```

### AFTER INSERTING THREE VALUES, FETCHING THE TABLE DATA



The screenshot shows a MySQL IDE interface. On the left, a code editor displays SQL commands for creating a table, inserting data, and fetching it. On the right, the output pane shows the results of the 'SELECT' command.

```
1 -- create  
2 CREATE TABLE EMPLOYEE (  
3     empId INTEGER PRIMARY KEY,  
4     name TEXT NOT NULL,  
5     dept TEXT NOT NULL  
6 );  
7  
8  
9 -- insert  
10 INSERT INTO EMPLOYEE VALUES (0001, 'Clark', 'Sales');  
11 INSERT INTO EMPLOYEE VALUES (0002, 'Dave', 'Accounting');  
12 INSERT INTO EMPLOYEE VALUES (0003, 'Ava', 'Sales');  
13  
14 -- fetch  
15 SELECT * FROM EMPLOYEE;  
16
```

Output:

empId	name	dept
1	Clark	Sales
2	Dave	Accounting
3	Ava	Sales

### UPDATE COMMAND

```
update EMPLOYEE set name='Jeeva',dept='HR' where empId=0003;
```

## AFTER UPDATING , FETCHING THE UPDATED DATA ENTRY FROM THE TABLE

The screenshot shows a MySQL query editor with the following SQL code:

```
1
2 -- create
3 CREATE TABLE EMPLOYEE (
4   empId INTEGER PRIMARY KEY,
5   name TEXT NOT NULL,
6   dept TEXT NOT NULL
7 );
8
9 -- insert
10 INSERT INTO EMPLOYEE VALUES (0001, 'Clark', 'Sales');
11 INSERT INTO EMPLOYEE VALUES (0002, 'Dave', 'Accounting');
12 INSERT INTO EMPLOYEE VALUES (0003, 'Ava', 'Sales');
13
14 -- fetch
15
16
17 -- update command
18 update EMPLOYEE set name='Jeeva',dept='HR' where empId=0003;
19
20 SELECT * FROM EMPLOYEE;
```

The output of the query is displayed on the right:

empId	name	dept
1	Clark	Sales
2	Dave	Accounting
3	Jeeva	HR

## DELETE COMMAND

The screenshot shows a MySQL query editor with the following SQL code:

```
1
2 CREATE TABLE EMPLOYEE (
3   empId INTEGER PRIMARY KEY,
4   name TEXT NOT NULL,
5   dept TEXT NOT NULL
6 );
7
8 INSERT INTO EMPLOYEE VALUES (0001, 'Clark', 'Sales');
9 INSERT INTO EMPLOYEE VALUES (0002, 'Dave', 'Accounting');
10 INSERT INTO EMPLOYEE VALUES (0003, 'Ava', 'Sales');
11
12
13 DELETE FROM EMPLOYEE WHERE empId=0003;
14
15 SELECT * FROM EMPLOYEE;
```

The output of the query is displayed on the right:

empId	name	dept
1	Clark	Sales
2	Dave	Accounting

## 2. CREATE TABLES AND PERFORM JOIN OPERATIONS IN MYSQL

### CREATING TWO TABLES NAMELY EMPLOYEE AND CUSTOMER AND INSERTING DATA

The screenshot shows a MySQL query editor with the following SQL code:

```
1
2 -- create
3 CREATE TABLE EMPLOYEE (
4   empId INTEGER PRIMARY KEY,
5   name TEXT NOT NULL,
6   dept TEXT NOT NULL
7 );
8
9 CREATE TABLE CUSTOMER (
10   cusId INTEGER PRIMARY KEY,
11   name TEXT NOT NULL,
12   address TEXT NOT NULL
13 );
14
15 -- insert
16 INSERT INTO EMPLOYEE VALUES (0001, 'Clark', 'Sales');
17 INSERT INTO EMPLOYEE VALUES (0002, 'Dave', 'Accounting');
18 INSERT INTO EMPLOYEE VALUES (0003, 'Ava', 'Sales');
19
20 INSERT INTO CUSTOMER VALUES (0001, 'John', 'Salem');
21 INSERT INTO CUSTOMER VALUES (0002, 'Ramesh', 'Chennai');
22 INSERT INTO CUSTOMER VALUES (0003, 'Kaviya', 'Vellore');
23
24
25 SELECT * FROM EMPLOYEE;
26 SELECT * FROM CUSTOMER;
```

The output of the query is displayed on the right:

empId	name	dept
1	Clark	Sales
2	Dave	Accounting
3	Ava	Sales

  

cusId	name	address
1	John	Salem
2	Ramesh	Chennai
3	Kaviya	Vellore

## COMMANDS FOR CREATING THE ABOVE TWO TABLES

CREATE TABLE EMPLOYEE (

```

empId INTEGER PRIMARY KEY,

name TEXT NOT NULL,

dept TEXT NOT NULL

);

```

```

CREATE TABLE CUSTOMER (

cusId INTEGER PRIMARY KEY,

name TEXT NOT NULL,

address TEXT NOT NULL

);

```

## PERFORMING JOIN OPERATIONS ON THE ABOVE TWO TABLES

### INNER JOIN



The screenshot shows a MySQL query editor with a query named 'queries.sql'. The query creates two tables, EMPLOYEE and CUSTOMER, and inserts data into them. It then performs an INNER JOIN on the two tables based on the condition EMPLOYEE.place = CUSTOMER.address. The output shows three rows of data.

```

1 -- create
2
3 CREATE TABLE EMPLOYEE (
4   empId INTEGER PRIMARY KEY,
5   name TEXT NOT NULL,
6   dept TEXT NOT NULL,
7   place TEXT NOT NULL
8 );
9
10 CREATE TABLE CUSTOMER (
11   cusId INTEGER PRIMARY KEY,
12   name TEXT NOT NULL,
13   address TEXT NOT NULL
14 );
15 -- insert
16 INSERT INTO EMPLOYEE VALUES (0001, 'Clark', 'Sales', 'Chennai');
17 INSERT INTO EMPLOYEE VALUES (0002, 'Dave', 'Accounting', 'Salem');
18 INSERT INTO EMPLOYEE VALUES (0003, 'Ava', 'Sales', 'Chennai');
19 INSERT INTO CUSTOMER VALUES (0001, 'John', 'Salem');
20 INSERT INTO CUSTOMER VALUES (0002, 'Ramesh', 'Chennai');
21 INSERT INTO CUSTOMER VALUES (0003, 'Kaviya', 'Vellore');
22
23 SELECT EMPLOYEE.empId, EMPLOYEE.name, CUSTOMER.cusId, CUSTOMER.name, EMPLOYEE.place FROM
24 EMPLOYEE INNER JOIN CUSTOMER ON EMPLOYEE.place=CUSTOMER.address;
25

```

Output:

empId	name	cusId	name	place
2	Dave	1	John	Salem
3	Ava	2	Ramesh	Chennai
1	Clark	2	Ramesh	Chennai

### LEFT JOIN



The screenshot shows a MySQL query editor with a query named 'queries.sql'. The query creates two tables, EMPLOYEE and CUSTOMER, and inserts data into them. It then performs a LEFT JOIN on the two tables based on the condition EMPLOYEE.place = CUSTOMER.address. The output shows three rows of data, including rows from the EMPLOYEE table that do not have a matching row in the CUSTOMER table.

```

1 -- create
2
3 CREATE TABLE EMPLOYEE (
4   empId INTEGER PRIMARY KEY,
5   name TEXT NOT NULL,
6   dept TEXT NOT NULL,
7   place TEXT NOT NULL
8 );
9
10 CREATE TABLE CUSTOMER (
11   cusId INTEGER PRIMARY KEY,
12   name TEXT NOT NULL,
13   address TEXT NOT NULL
14 );
15 -- insert
16 INSERT INTO EMPLOYEE VALUES (0001, 'Clark', 'Sales', 'Chennai');
17 INSERT INTO EMPLOYEE VALUES (0002, 'Dave', 'Accounting', 'Salem');
18 INSERT INTO EMPLOYEE VALUES (0003, 'Ava', 'Sales', 'Chennai');
19
20 INSERT INTO CUSTOMER VALUES (0001, 'John', 'Salem');
21 INSERT INTO CUSTOMER VALUES (0002, 'Ramesh', 'Chennai');
22 INSERT INTO CUSTOMER VALUES (0003, 'Kaviya', 'Vellore');
23
24
25 SELECT EMPLOYEE.empId, EMPLOYEE.name, CUSTOMER.cusId, CUSTOMER.name, EMPLOYEE.place FROM
26 EMPLOYEE LEFT JOIN CUSTOMER ON EMPLOYEE.place=CUSTOMER.address;
27

```

Output:

empId	name	cusId	name	place
1	Clark	2	Ramesh	Chennai
2	Dave	1	John	Salem
3	Ava	2	Ramesh	Chennai

## RIGHT JOIN

queries.sql3z9wgpfee

```
1 -- create
2 CREATE TABLE EMPLOYEE (
3   empId INTEGER PRIMARY KEY,
4   name TEXT NOT NULL,
5   dept TEXT NOT NULL,
6   place TEXT NOT NULL
7 );
8
9 CREATE TABLE CUSTOMER (
10  cusId INTEGER PRIMARY KEY,
11  name TEXT NOT NULL,
12  address TEXT NOT NULL
13 );
14
15 -- insert
16 INSERT INTO EMPLOYEE VALUES (0001, 'Clark', 'Sales', 'Chennai');
17 INSERT INTO EMPLOYEE VALUES (0002, 'Dave', 'Accounting', 'Salem');
18 INSERT INTO EMPLOYEE VALUES (0003, 'Ava', 'Sales', 'Chennai');
19
20 INSERT INTO CUSTOMER VALUES (0001, 'John', 'Salem');
21 INSERT INTO CUSTOMER VALUES (0002, 'Ramesh', 'Chennai');
22 INSERT INTO CUSTOMER VALUES (0003, 'Kaviya', 'Vellore');
23
24
25 SELECT EMPLOYEE.empId,EMPLOYEE.name,CUSTOMER.cusId,CUSTOMER.name,EMPLOYEE.place FROM
26 EMPLOYEE RIGHT JOIN CUSTOMER ON EMPLOYEE.place=CUSTOMER.address;
27
```

STDIN

Input for the program ( Optional )

Output:

empId	name	cusId	name	place
2	Dave	1	John	Salem
3	Ava	2	Ramesh	Chennai
1	Clark	2	Ramesh	Chennai
NULL	NULL	3	Kaviya	NULL

## 3. CREATE, UPDATE, DELETE COMMANDS IN MONGODB

### CREATE COMMAND

script.js3z9wh5d2p

```
1 db.employee.insertOne({
2   name: "Hari",
3   empId: 002,
4   age: 22,
5   place: "chennai"
6 });
7
8 db.employee.insertOne({
9   name: "Jasmine",
10  empId: 003,
11  age: 21,
12  place: "Chennai"
13 });
14
15 db.employee.insertOne({
16  name: "Kilian",
17  empId: 004,
18  age: 26,
19  place: "Chennai",
20  designation: "Manager"
21 });
```

STDIN

Input for the program ( Optional )

Output:

```
{
  "acknowledged" : true,
  "insertedId" : ObjectId("64732a5131e337dbfc72e838")
}
{
  "acknowledged" : true,
  "insertedId" : ObjectId("64732a5131e337dbfc72e839")
}
{
  "acknowledged" : true,
  "insertedId" : ObjectId("64732a5131e337dbfc72e83a")
}
```

created 1 year ago

### UPDATE COMMAND

script.js3z9wh5d2p

```
1 db.employee.insertOne({
2   name: "Hari",
3   empId: 002,
4   age: 22,
5   place: "chennai"
6 });
7
8 db.employee.insertOne({
9   name: "Jasmine",
10  empId: 003,
11  age: 21,
12  place: "Chennai"
13 });
14
15 db.employee.insertOne({
16  name: "Kilian",
17  empId: 004,
18  age: 26,
19  place: "Chennai",
20  designation: "Manager"
21 });
22
23
24 //updating one row of data in the collection using update command
25 db.employee.update({designation:"Manager"},{$set:{name:"Kiran Sharma"}});
```

STDIN

Input for the program ( Optional )

Output:

```
{
  "acknowledged" : true,
  "insertedId" : ObjectId("64732c61f5226299872bad85")
}
{
  "acknowledged" : true,
  "insertedId" : ObjectId("64732c61f5226299872bad86")
}
{
  "acknowledged" : true,
  "insertedId" : ObjectId("64732c61f5226299872bad87")
}
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 0 })
```

## DELETE COMMAND

```
script.js 3z9wh5d2p NEW MONGODB RUN
```

```
1 name: "Neha",
2 empid: 001,
3 age: 22,
4 place: "Chennai"
5 });
6
7
8 db.employee.insertOne({
9   name: "Jasmine",
10  empid: 003,
11  age: 21,
12  place: "Chennai"
13 });
14
15 db.employee.insertOne({
16   name: "Kilian",
17   empid: 004,
18   age: 26,
19   place: "Chennai",
20   designation: "Manager"
21 });
22
23
24 //updating one row of data in the collection using update command
25 db.employee.update({designation:"Manager"},{$set:{name:"Kiran Sharma"}});
26
27 //DELETING ONE ROW OF DATA IN THE COLLECTION USING DELETE command
28 db.employee.deleteOne({age:21});
```

STDIN  
Input for the program ( Optional )

Output:

```
{
  "acknowledged" : true,
  "insertedId" : ObjectId("64732cd738049e374985f46")
}
{
  "acknowledged" : true,
  "insertedId" : ObjectId("64732cd738049e374985f46")
}
{
  "acknowledged" : true,
  "insertedId" : ObjectId("64732cd738049e374985f46")
}
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 0 },
{ "acknowledged" : true, "deletedCount" : 1 })
```

created 1 year ago