

**SMART BRIDGE EXTERNSHIP Modern  
ApplicationDevelopment (Java Spring Boot)**

***Assignment – 2***

***by***

***MySQL and Mongo DB queries***

***Name: KAMALESH KO***

***Reg-No: 20MIS0318***

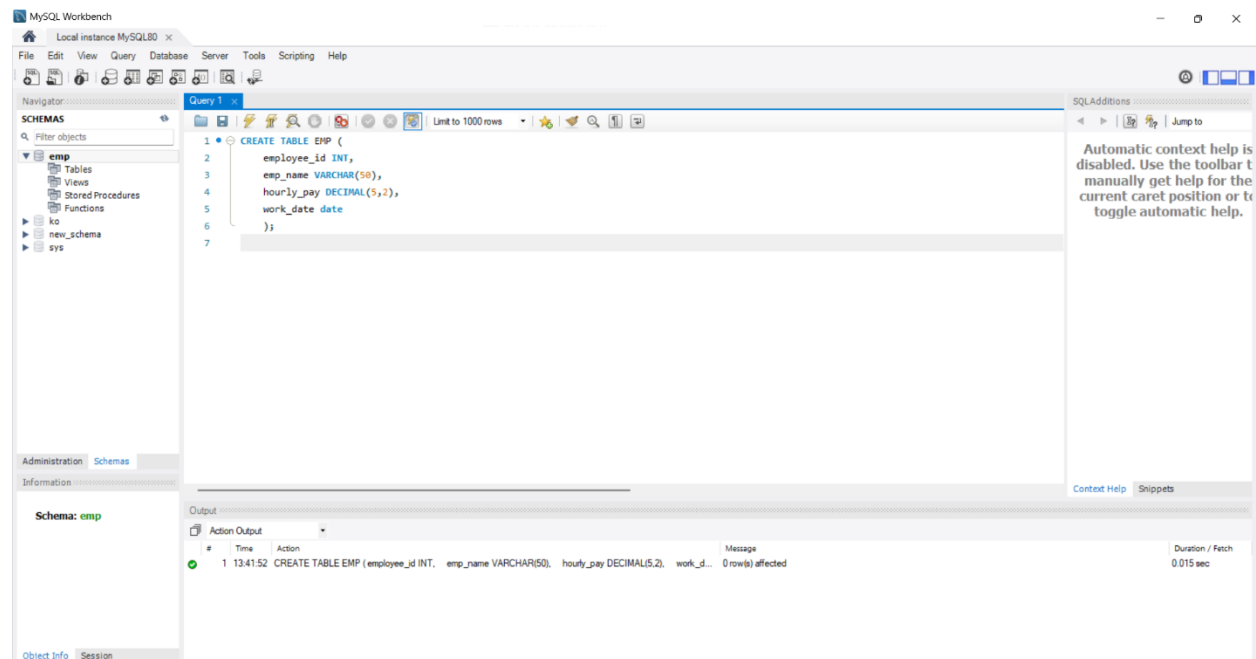
***Ph. No: 8825944747***

***email: [kamalesh.ko2020@vitstudent.ac.in](mailto:kamalesh.ko2020@vitstudent.ac.in)***

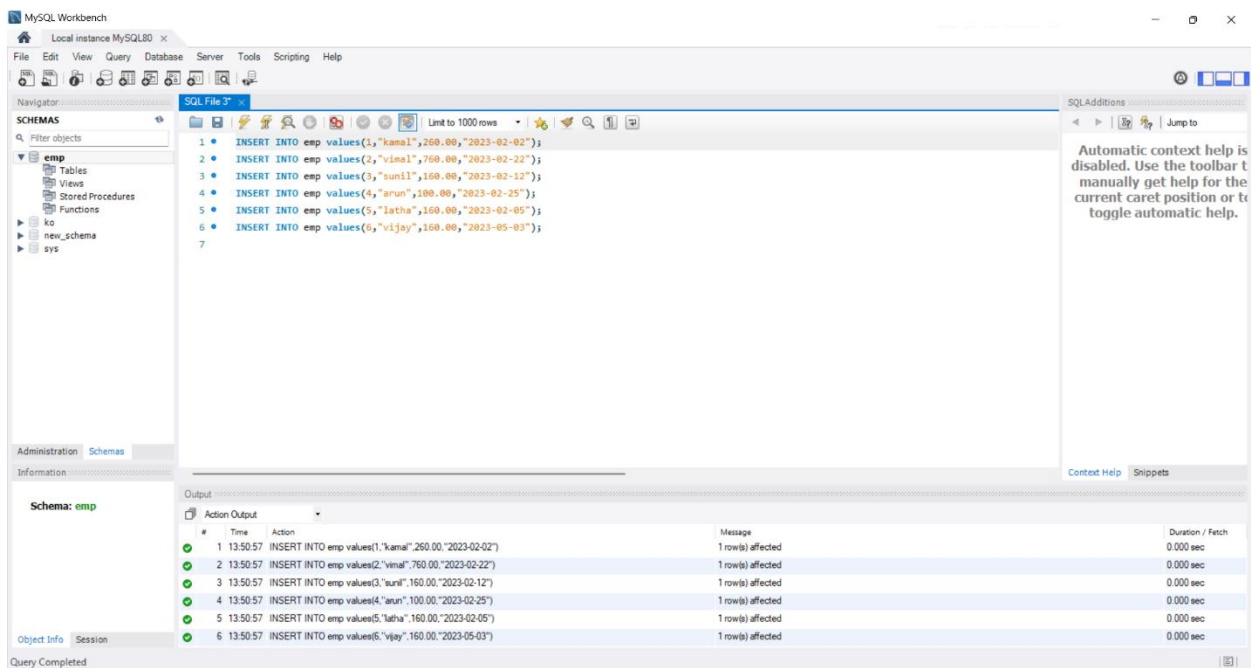
***campus: VIT Vellore***

# Screen shots of MySQL queries

## 1.create table



## 2.Insert values



### 3. Adding new column

The screenshot shows the MySQL Workbench interface. The SQL Editor contains the following queries:

1. `ALTER TABLE employees add age INT(2);`
2. `UPDATE employees SET age=22 WHERE employee_id= 1 ;`
3. `UPDATE employees SET age=21 WHERE employee_id= 2 ;`
4. `UPDATE employees SET age=22 WHERE employee_id= 3 ;`
5. `UPDATE employees SET age=23 WHERE employee_id= 5 ;`
6. `SELECT * FROM employees;`
- 7.

The Result Grid shows the output of the SELECT query:

employee_id	emp_name	hourly_pay	hire_date	age
1	Elamaran R	250.00	2023-05-28	22
2	Sanjai	250.00	2023-04-28	21
3	Kumar	100.00	2023-05-25	22
5	Arun	160.00	2023-05-26	23

The Output tab shows the execution results of the UPDATE and SELECT queries.

#	Time	Action	Message	Duration / Fetch
23	10:53:57	UPDATE employees SET age=21 WHERE employee_id= 2	1 row(s) affected Rows matched: 1 Changed: 1 Warnings: 0	0.000 sec
24	10:53:57	UPDATE employees SET age=22 WHERE employee_id= 3	1 row(s) affected Rows matched: 1 Changed: 1 Warnings: 0	0.000 sec
25	10:53:57	UPDATE employees SET age=23 WHERE employee_id= 5	1 row(s) affected Rows matched: 1 Changed: 1 Warnings: 0	0.000 sec
26	10:53:57	SELECT * FROM employees LIMIT 0, 1000	4 row(s) returned	0.000 sec / 0.000 sec

### 4. Update

The screenshot shows the MySQL Workbench interface. The SQL Editor contains the following queries:

1. `UPDATE employees SET emp_name = "Elamaran R" where employee_id=1;`
2. `SELECT * FROM employees;`
- 3.

The Result Grid shows the output of the SELECT query:

employee_id	emp_name	hourly_pay	hire_date
1	Elamaran R	250.00	2023-05-28
2	Sanjai	250.00	2023-04-28
3	Kumar	100.00	2023-05-25
4	Vijay	130.00	2023-05-23
5	Arun	160.00	2023-05-26

The Output tab shows the execution results of the INSERT, UPDATE, and SELECT queries.

#	Time	Action	Message	Duration / Fetch
9	10:39:24	INSERT INTO employees values (5,"Arun", 160.00, "2023-05-26")	1 row(s) affected	0.015 sec
10	10:40:48	SELECT * FROM employees LIMIT 0, 1000	5 row(s) returned	0.000 sec / 0.000 sec
11	10:45:13	UPDATE employees SET emp_name = "Elamaran R" where employee_id=1	1 row(s) affected Rows matched: 1 Changed: 1 Warnings: 0	0.000 sec
12	10:45:13	SELECT * FROM employees LIMIT 0, 1000	5 row(s) returned	0.000 sec / 0.000 sec

## 5. Select

The screenshot shows the MySQL Workbench interface. The SQL Editor contains the query: `1 • SELECT * FROM employees;`. The Results grid displays the following data:

employee_id	emp_name	hourly_pay	hire_date
1	Elamran	250.00	2023-05-28
2	Sargai	250.00	2023-04-28
3	Kumar	100.00	2023-05-25
4	Vijay	130.00	2023-05-23
5	Arun	160.00	2023-05-26

The Output tab shows the execution log:

#	Time	Action	Message	Duration / Fetch
1	10:31:46	CREATE TABLE employees ( employee_id INT, emp_name VARCHAR(50), hourly_pay DECIMAL(5,2), hire_date DATE )	0 row(s) affected	0.250 sec
2	10:37:34	INSERT INTO employee values (1,"Elamran", 1500, "2023-05-28")	Error Code: 1146. Table java_springboot.employee doesn't exist	0.016 sec
3	10:38:10	INSERT INTO employees values (1,"Elamran", 1500, "2023-05-28")	Error Code: 1264. Out of range value for column hourly_pay/ at row 1	0.047 sec
4	10:38:55	INSERT INTO employees values (1,"Elamran", 1500.00, "2023-05-28")	Error Code: 1264. Out of range value for column hourly_pay/ at row 1	0.000 sec
5	10:39:24	INSERT INTO employees values (1 "Elamran" 250.00 "2023-05-28")	1 row(s) affected	0.016 sec

## 6. Foreign Key

The screenshot shows the MySQL Workbench interface. The SQL Editor contains the following queries:

```
1 • create table department (  
2     employee_id INT(3) primary key,  
3     dpt_id varchar(10),  
4     dpt_name varchar(10),  
5     foreign key(employee_id) references employees(employee_id)  
6 );  
7 • desc department;
```

The Results grid displays the structure of the 'department' table:

Field	Type	Null	Key	Default	Extra
employee_id	int	NO	PK		
dpt_id	varchar(10)	YES			
dpt_name	varchar(10)	YES			

The Output tab shows the execution log:

#	Time	Action	Message	Duration / Fetch
30	11:01:41	SELECT * FROM employees LIMIT 0. 1000	4 row(s) returned	0.000 sec / 0.000 sec
31	11:01:56	desc employees	5 row(s) returned	0.000 sec / 0.000 sec
32	11:14:58	create table department ( employee_id INT(3) primary key, dpt_id varchar(10), dpt_name varchar(10), foreign key(employee_id) references employees(employee_id) );	0 row(s) affected, 1 warning(s): 1681 Integer display width is deprecated and will be removed in a future release.	0.016 sec
33	11:15:25	desc department	3 row(s) returned	0.000 sec / 0.000 sec

## 7. Delete

The screenshot shows the MySQL Workbench interface. The SQL editor contains the following queries:

```
1 • DELETE FROM employees where employee_id = 4;  
2 • SELECT * FROM employees;  
3
```

The Result Grid displays the results of the SELECT query:

employee_id	emp_name	hourly_pay	hire_date
1	Elamaran R	250.00	2023-05-28
2	Sanjai	250.00	2023-04-28
3	Kumar	100.00	2023-05-25
5	Arun	160.00	2023-05-26

The Output tab shows the execution log:

#	Time	Action	Message	Duration / Fetch
13	10:46:48	commit	0 row(s) affected	0.000 sec
14	10:47:33	SELECT * FROM employees LIMIT 0, 1000	5 row(s) returned	0.000 sec / 0.000 sec
15	10:47:46	DELETE FROM employees where employee_id = 4	1 row(s) affected	0.015 sec
16	10:47:46	SELECT * FROM employees LIMIT 0, 1000	4 row(s) returned	0.000 sec / 0.000 sec

## 8. Inner Join

The screenshot shows the MySQL Workbench interface. The SQL editor contains the following query:

```
1 select * from  
2 employees inner join department  
3 on employees.employee_id = department.employee_id;
```

The Result Grid displays the results of the INNER JOIN query:

employee_id	emp_name	hourly_pay	hire_date	age	employee_id	dpt_id	dpt_name
1	Elamaran R	250.00	2023-05-28	22	1	D1	HR
2	Sanjai	250.00	2023-04-28	21	2	D2	Testing
3	Kumar	100.00	2023-05-25	22	3	D2	Testing
5	Arun	160.00	2023-05-26	23	5	D3	Design

The Output tab shows the execution log:

#	Time	Action	Message	Duration / Fetch
40	11:23:54	alter table department add constraint emp_foreign_key foreign key(employee_id) references employees(e...	4 row(s) affected Records: 4 Duplicates: 0 Warnings: 0	0.046 sec
41	11:24:43	alter table department drop constraint department_fk_1	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0	0.016 sec
42	11:25:28	select * from department LIMIT 0, 1000	4 row(s) returned	0.000 sec / 0.000 sec
43	11:28:11	select * from employees inner join department on employees.employee_id = department.employee_id LIM...	4 row(s) returned	0.000 sec / 0.000 sec

## 9. Right Join

The screenshot shows MySQL Workbench with a SQL query window. The query is a right join between the `employees` and `department` tables on the `employee_id` column. The result grid shows 5 rows, all from the `employees` table, as every employee belongs to a department.

```
1 select * from
2 employees right join department
3 on employees.employee_id = department.employee_id;
```

employee_id	emp_name	hourly_pay	hire_date	age	employee_id	dpt_id	dpt_name
1	Elamran R	250.00	2023-05-28	22	1	D1	HR
2	Sanjai	250.00	2023-04-28	21	2	D2	Testing
3	Kumar	100.00	2023-05-25	22	3	D2	Testing
5	Arun	160.00	2023-05-26	23	5	D3	Design

The output window shows the execution log with 4 rows returned for each of the 4 queries executed.

## 10. Left Join

The screenshot shows MySQL Workbench with a SQL query window. The query is a left join between the `employees` and `department` tables on the `employee_id` column. The result grid shows 5 rows, all from the `employees` table, as every employee belongs to a department.

```
1 select * from
2 employees left join department
3 on employees.employee_id = department.employee_id;
4
```

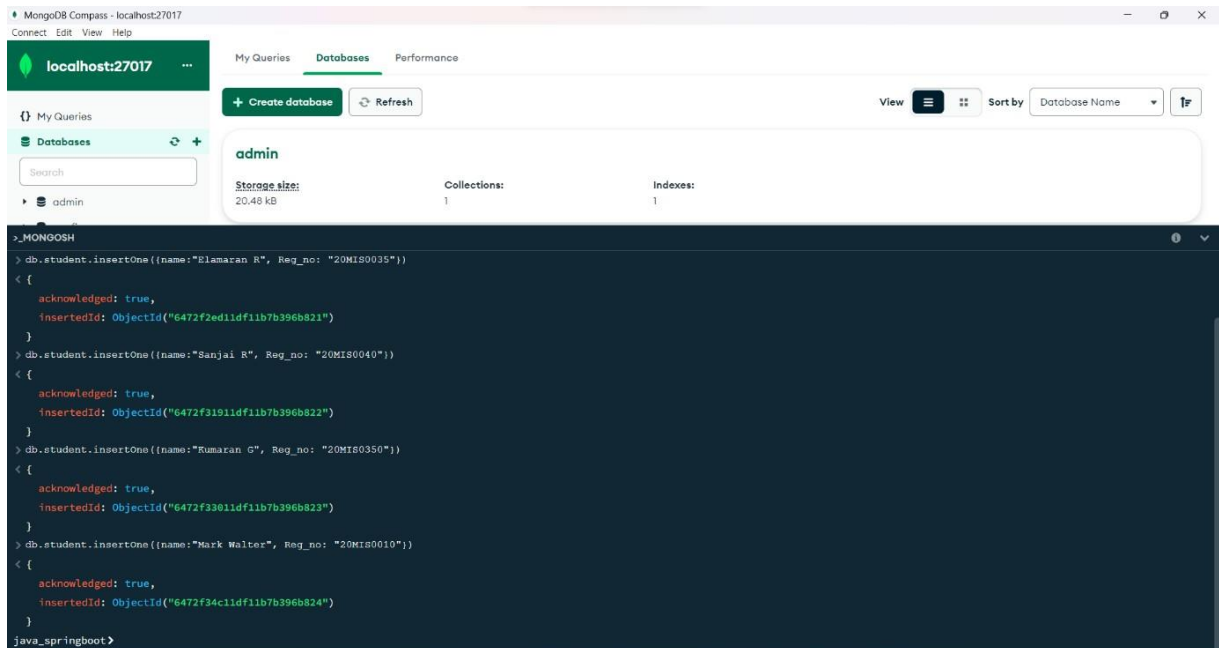
employee_id	emp_name	hourly_pay	hire_date	age	employee_id	dpt_id	dpt_name
1	Elamran R	250.00	2023-05-28	22	1	D1	HR
2	Sanjai	250.00	2023-04-28	21	2	D2	Testing
3	Kumar	100.00	2023-05-25	22	3	D2	Testing
4	Vijay	200.00	2023-05-25	20	4	D2	Testing
5	Arun	160.00	2023-05-26	23	5	D3	Design

The output window shows the execution log with 4 rows returned for each of the 4 queries executed.

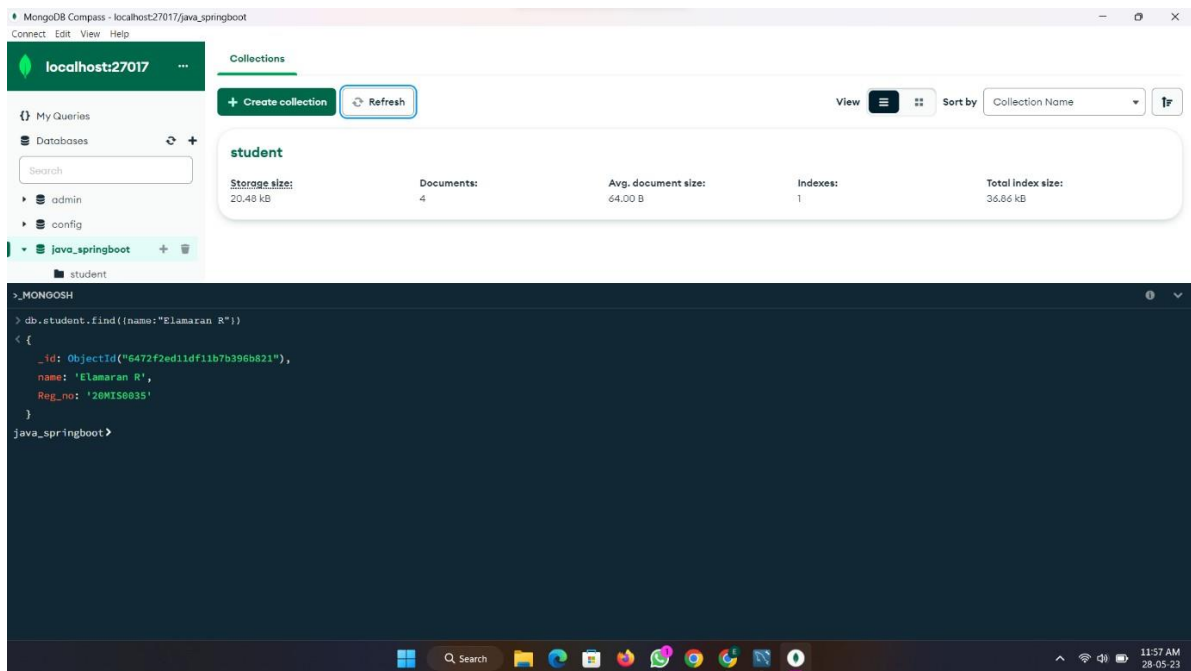


# Mongo DB queries:

## 1. Creation and Insertion



## 2. Selection of Find



### 3.Update

The screenshot shows the MongoDB Compass interface. The left sidebar displays the database structure: 'admin', 'config', 'java\_springboot', and 'student'. The 'student' collection is selected. The main panel shows the 'Documents' tab for 'java\_springboot.student'. A filter bar is present with a search input and a 'Find' button. Below the filter bar, there are buttons for 'ADD DATA' and 'EXPORT DATA'. The document list shows one document with the following fields: '\_id: ObjectId('6472f2ed11df11b7b396b821')', 'name: "Elanaran R"', and 'Reg\_no: "20MIS0035"'. Below the document list, a terminal window shows the following commands and output:

```
>_MONOOOSH
> db.student.updateOne({'_id:ObjectId('6472f2ed11df11b7b396b821')'}, {'$set: {name: "Kavin R"}})
< {
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 0,
  upsertedCount: 0
}
> db.student.findOne({'name: "Kavin R"}')
< {
  _id: ObjectId("6472f2ed11df11b7b396b821"),
  name: 'Kavin R',
  Reg_no: '20MIS0035'
}
java_springboot>
```

### 4.Delete

The screenshot shows the MongoDB Compass interface. The left sidebar displays the database structure: 'admin', 'config', 'java\_springboot', and 'student'. The 'student' collection is selected. The main panel shows the 'Documents' tab for 'java\_springboot.student'. A filter bar is present with a search input and a 'Find' button. Below the filter bar, there are buttons for 'ADD DATA' and 'EXPORT DATA'. The document list shows three documents with the following fields: '\_id: ObjectId('6472f2ed11df11b7b396b821')', 'name: 'Kavin R'', 'Reg\_no: '20MIS0035'', '\_id: ObjectId('6472f31911df11b7b396b822')', 'name: 'Sanjai R'', 'Reg\_no: '20MIS0040'', and '\_id: ObjectId('6472f33011df11b7b396b823')', 'name: 'Kumaran G'', 'Reg\_no: '20MIS0350''. Below the document list, a terminal window shows the following commands and output:

```
>_MONOOOSH
> db.student.deleteOne({'name: "Mark Walter"}')
< {
  acknowledged: true,
  deletedCount: 1
}
> db.student.find()
< {
  _id: ObjectId("6472f2ed11df11b7b396b821"),
  name: 'Kavin R',
  Reg_no: '20MIS0035'
}
{
  _id: ObjectId("6472f31911df11b7b396b822"),
  name: 'Sanjai R',
  Reg_no: '20MIS0040'
}
{
  _id: ObjectId("6472f33011df11b7b396b823"),
  name: 'Kumaran G',
  Reg_no: '20MIS0350'
}
java_springboot>
```