

NAME: SAI VIGNESH.N

REG.NO: 20BCE1128

CAMPUS: VIT – CHENNAI

PHONE: 9940096800

EMAIL: [saivignesh.n2020@vitstudent.ac.in](mailto:saivignesh.n2020@vitstudent.ac.in)

## WEEK 2 ASSIGNMENT – SQL AND MONGO

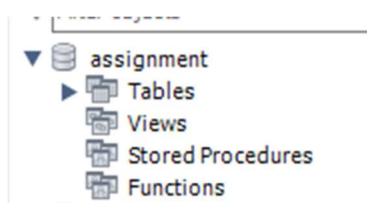
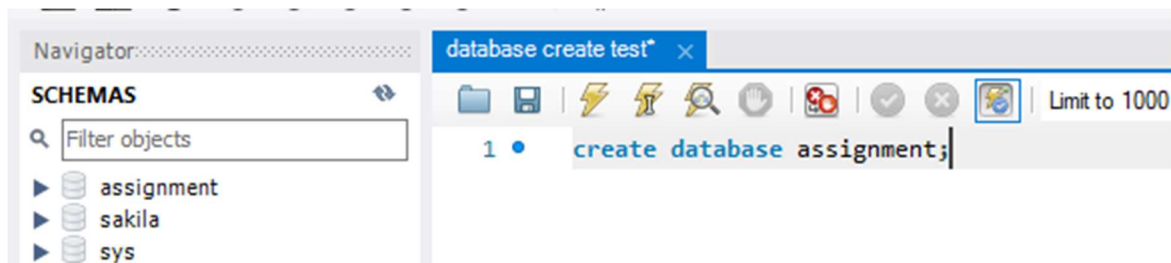
Dear Sir/Mam,

I have not yet completely set up the Mongo app since I have certain issues in it. So I am doing Mongo in online. I request you to accept this.

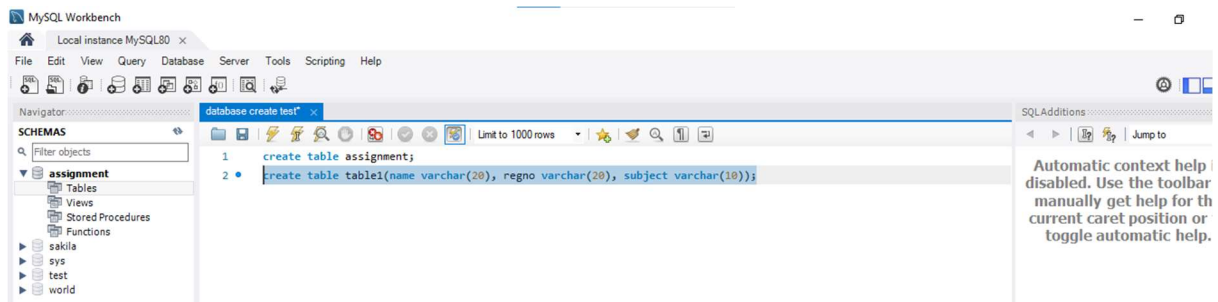
SQL:

**CREATE DATABASE:**

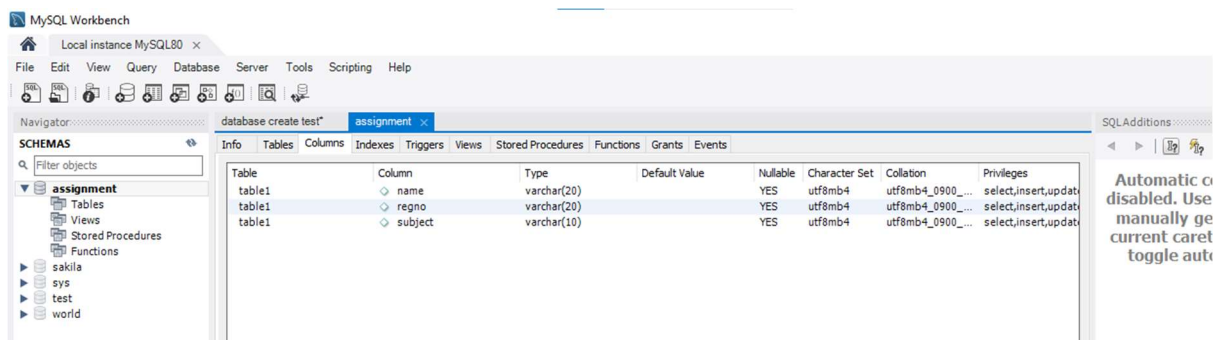
create database assignment;



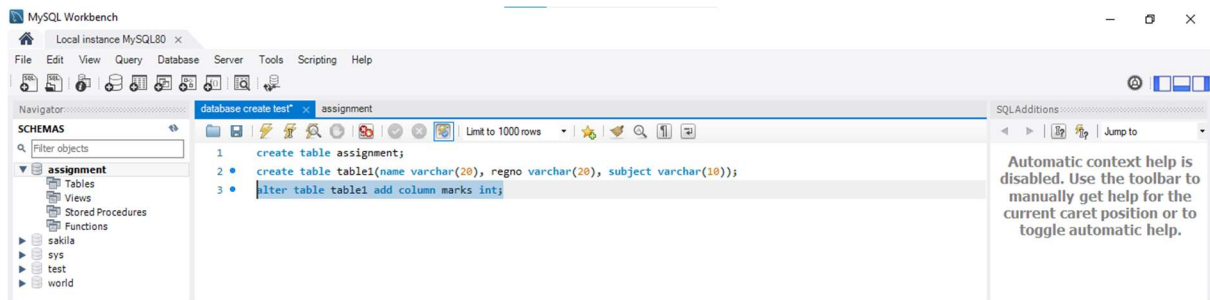
## CREATE TABLE:



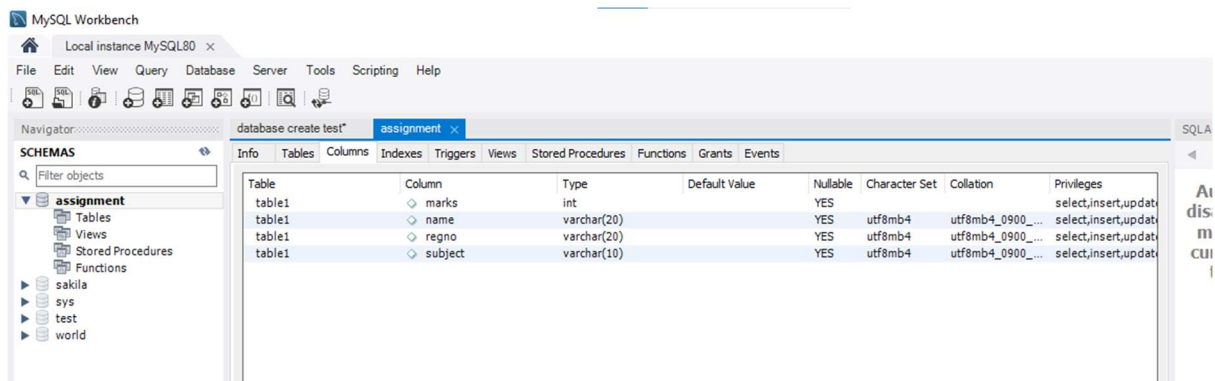
create table table1(name varchar(20), regno varchar(20), subject varchar(10));



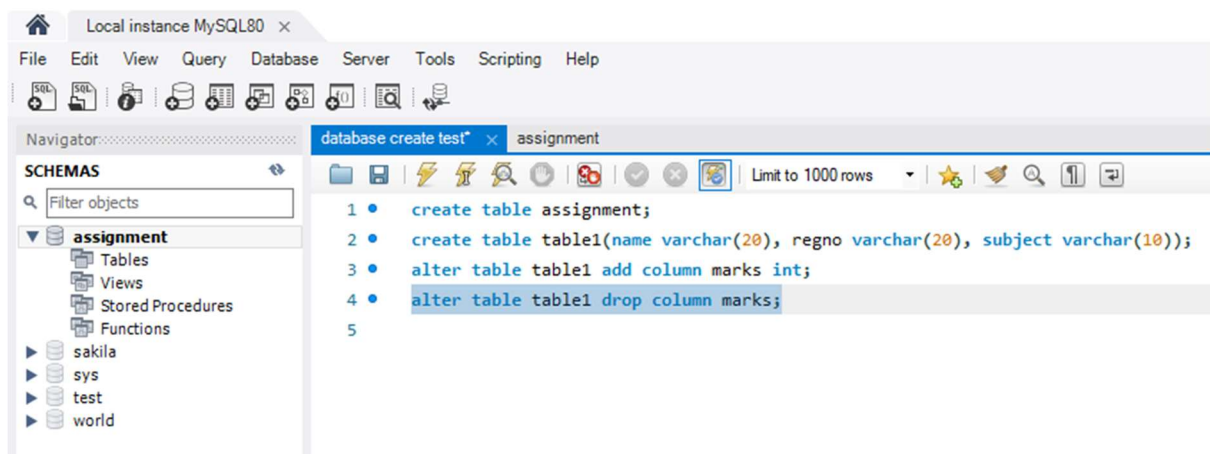
## ADD COLUMN



alter table table1 add column marks int;



## DROP COLUMN



alter table table1 drop column marks;

Table	Column	Type	Default Value	Nullable	Character Set	Collation	Privileges
table1	name	varchar(20)		YES	utf8mb4	utf8mb4_0900_...	select,insert,update
table1	regno	varchar(20)		YES	utf8mb4	utf8mb4_0900_...	select,insert,update
table1	subject	varchar(10)		YES	utf8mb4	utf8mb4_0900_...	select,insert,update

## MODIFY

database create test\* x assignment

Limit to 1000 rows

```

1 • alter table table1 add column marks varchar(20);
2 • alter table table1 modify column marks int;
3
4

```

SCHEMAS

Filter objects

- assignment
  - Tables
  - Views
  - Stored Procedures
  - Functions
- sakila

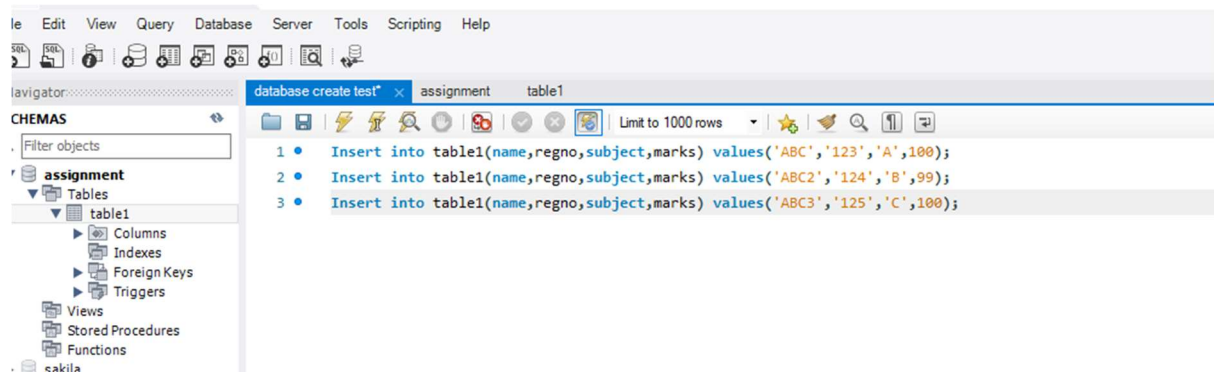
alter table table1 add column marks varchar(20);

Table	Column	Type	Default Value
table1	marks	varchar(20)	

alter table table1 modify column marks int;

Table	Column	Type
table1	marks	int

## INSERT



Insert into table1(name,regno,subject,marks) values('ABC','123','A',100);

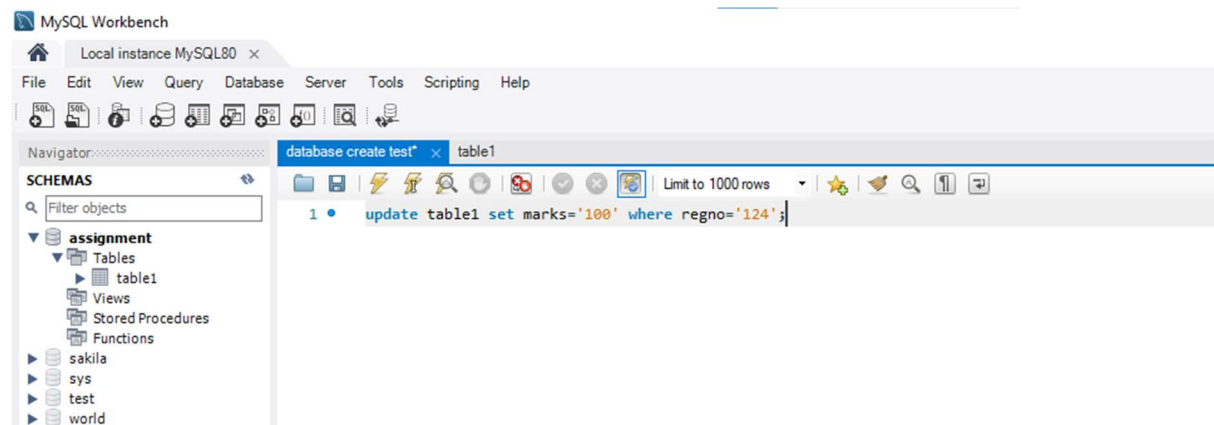
Insert into table1(name,regno,subject,marks) values('ABC2','124','B',99);

Insert into table1(name,regno,subject,marks) values('ABC3','125','C',100);

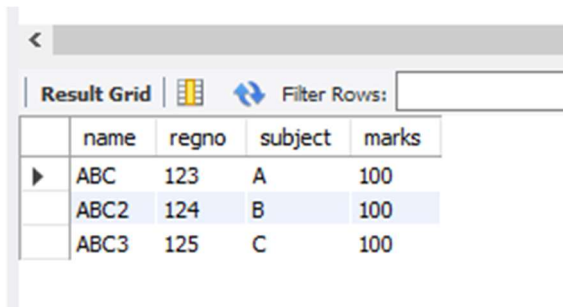
The screenshot shows the 'Result Grid' of MySQL Workbench. It displays the data inserted into 'table1' in the 'assignment' database. The table has four columns: name, regno, subject, and marks. The data is as follows:

	name	regno	subject	marks
▶	ABC	123	A	100
	ABC2	124	B	99
	ABC3	125	C	100

## UPDATE



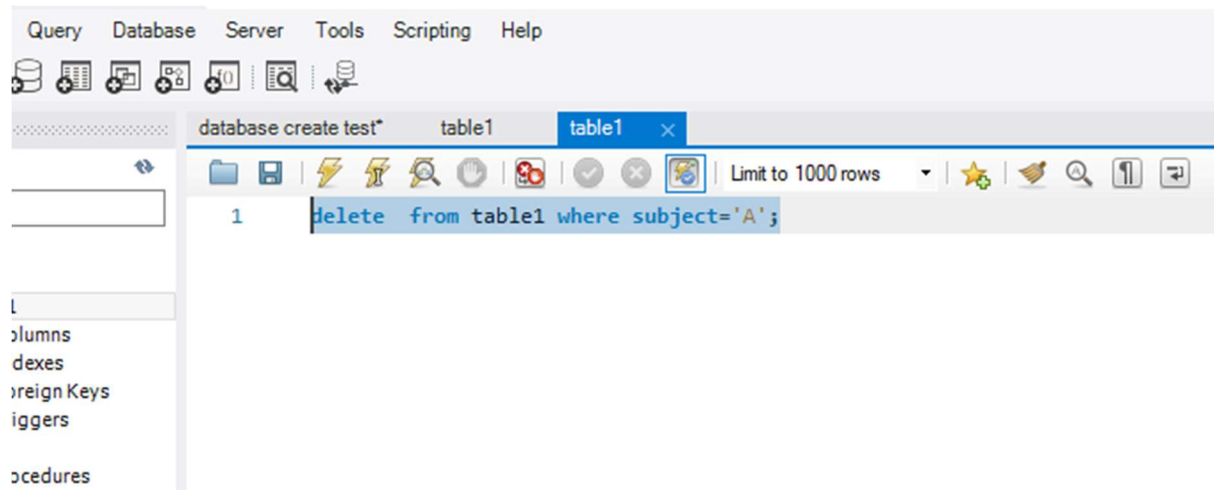
update table1 set marks='100' where regno='124';



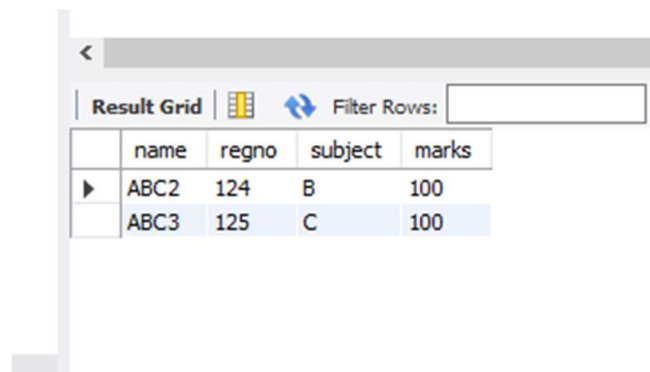
The screenshot shows a database interface with a 'Result Grid' tab. It displays a table with 5 columns: name, regno, subject, and marks. There are three rows of data. The first row has name 'ABC', regno '123', subject 'A', and marks '100'. The second row has name 'ABC2', regno '124', subject 'B', and marks '100'. The third row has name 'ABC3', regno '125', subject 'C', and marks '100'. The second row is highlighted with a blue background.

	name	regno	subject	marks
▶	ABC	123	A	100
	ABC2	124	B	100
	ABC3	125	C	100

### DELETE:



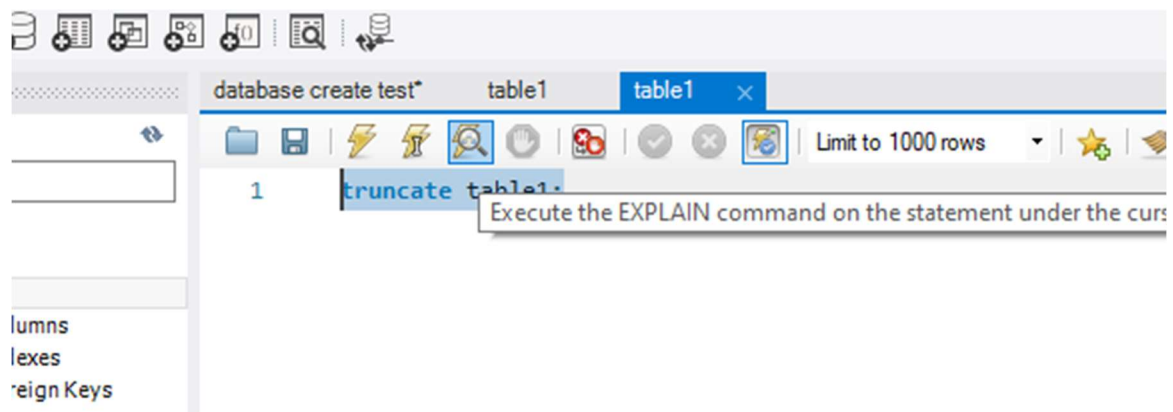
delete from table1 where subject='A';



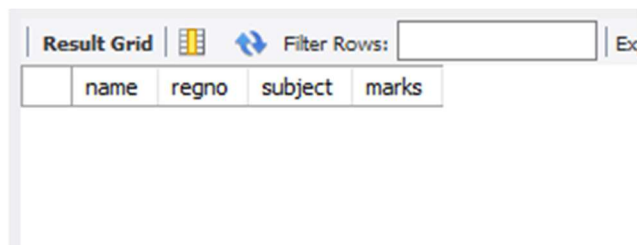
The screenshot shows a database interface with a 'Result Grid' tab. It displays a table with 5 columns: name, regno, subject, and marks. There are two rows of data. The first row has name 'ABC2', regno '124', subject 'B', and marks '100'. The second row has name 'ABC3', regno '125', subject 'C', and marks '100'. The second row is highlighted with a blue background.

	name	regno	subject	marks
▶	ABC2	124	B	100
	ABC3	125	C	100

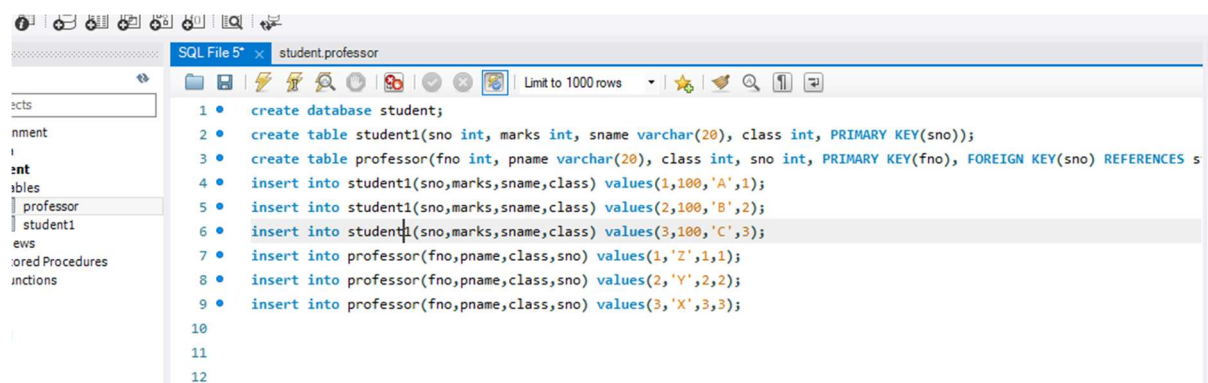
## TRUNCATE



truncate table1;



## JOINS:



create database student;

create table student1(sno int, marks int, sname varchar(20), class int, PRIMARY KEY(sno));

```
create table professor(fno int, pname varchar(20), class int, sno int, PRIMARY KEY(fno), FOREIGN  
KEY(sno) REFERENCES student1(sno));
```

```
insert into student1(sno,marks,sname,class) values(1,100,'A',1);
```

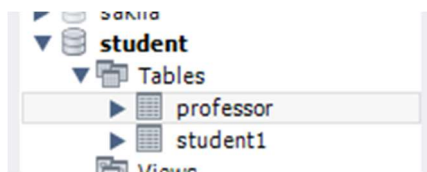
```
insert into student1(sno,marks,sname,class) values(2,100,'B',2);
```

```
insert into student1(sno,marks,sname,class) values(3,100,'C',3);
```

```
insert into professor(fno,pname,class,sno) values(1,'Z',1,1);
```

```
insert into professor(fno,pname,class,sno) values(2,'Y',2,2);
```

```
insert into professor(fno,pname,class,sno) values(3,'X',3,3);
```



A screenshot of a database management tool's 'Result Grid' showing the data for the 'professor' table. The grid has four columns: 'fno', 'pname', 'class', and 'sno'. It contains three rows of data and a final row with all NULL values.

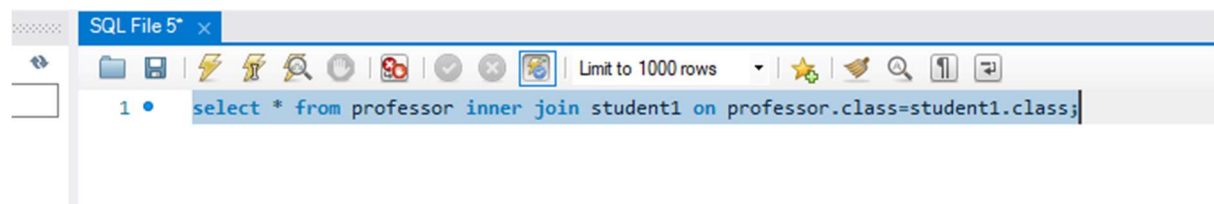
	fno	pname	class	sno
▶	1	Z	1	1
	2	Y	2	2
	3	X	3	3
*	NULL	NULL	NULL	NULL

A screenshot of a database management tool's 'Result Grid' showing the data for the 'student1' table. The grid has four columns: 'sno', 'marks', 'sname', and 'class'. It contains three rows of data and a final row with all NULL values.

	sno	marks	sname	class
▶	1	100	A	1
	2	100	B	2
	3	100	C	3
*	NULL	NULL	NULL	NULL



## INNER JOIN

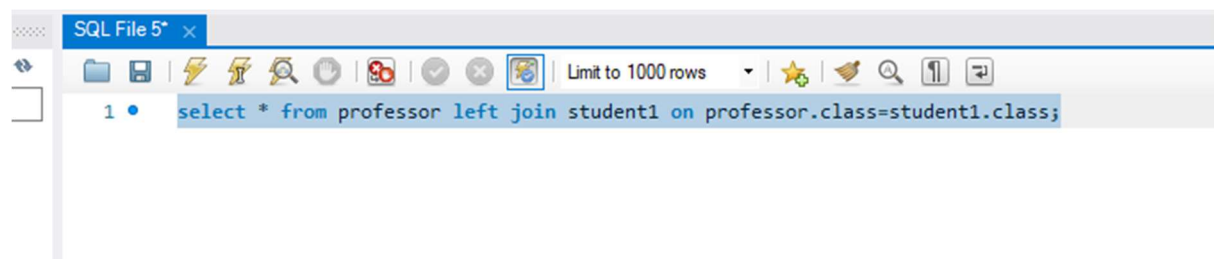


select \* from professor inner join student1 on professor.class=student1.class;

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

	fno	pname	class	sno	sno	marks	sname	class
▶	1	Z	1	1	1	100	A	1
	2	Y	2	2	2	100	B	2
	3	X	3	3	3	100	C	3

## LEFT OUTER JOIN:

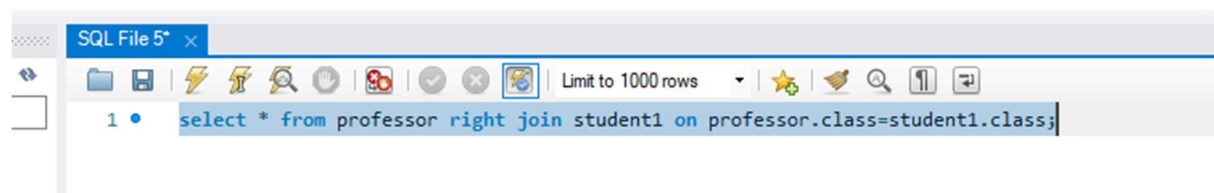


select \* from professor left join student1 on professor.class=student1.class;

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

	fno	pname	class	sno	sno	marks	sname	class
▶	1	Z	1	1	1	100	A	1
	2	Y	2	2	2	100	B	2
	3	X	3	3	3	100	C	3

## RIGHT OUTER JOIN:

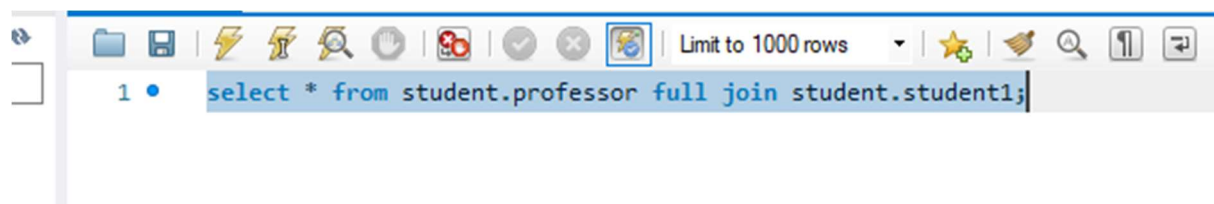


select \* from professor right join student1 on professor.class=student1.class;

Result Grid | Filter Rows: | Export: | Wrap C

	fno	pname	class	sno	sno	marks	sname	class
▶	1	Z	1	1	1	100	A	1
	2	Y	2	2	2	100	B	2
	3	X	3	3	3	100	C	3

## FULL JOIN:



select \* from student.professor full join student.student1;

Result Grid | Filter Rows: | Export: | Wrap Cell Cont

	fno	pname	class	sno	sno	marks	sname	class
▶	3	X	3	3	1	100	A	1
	2	Y	2	2	1	100	A	1
	1	Z	1	1	1	100	A	1
	3	X	3	3	2	100	B	2
	2	Y	2	2	2	100	B	2
	1	Z	1	1	2	100	B	2
	3	X	3	3	3	100	C	3
	2	Y	2	2	3	100	C	3
	1	Z	1	1	3	100	C	3

Result 5 x

## MONGO DB:

### INSERT:

```
use database1
```

```
db.createCollection("details")
```

```
db.details.insertOne({
```

```
  name: "abc",
```

```
  id: "1",
```

```
  gender: "f",
```

```
  age:20,
```

```
  date: Date()
```

```
})
```

```
db.details.insertOne({
```

```
  name: "abc",
```

```
  id: "1",
```

```
  gender: "f",
```

```
  age:20,
```

```
  date: Date()
```

```
})
```

```
db.details.find()
```



The screenshot shows a web browser window with the URL `jdoodle.com/online-mongodb-terminal/`. The page title is "Online mongoDB Terminal - pra: x". The main content area displays the MongoDB logo and version "3.2.4". Below the logo is a terminal window with a black background and white text. The terminal output shows the following commands and results:

```
Welcome to JDoodle - online mongo Terminal, Starting mongo Terminal, Please wait...
New mongoDB session started...
>use database1
switched to db database1

>db.createCollection("details")
{ "ok" : 1 }

>db.details.insertOne({
  name: "abc",
  id: "1",
  gender: "f",
  age:20,
  date: Date()
})
{
  "acknowledged" : true,
  "insertedId" : ObjectId("647382d97101d54adf994859")
}

>db.details.find()
{ "_id" : ObjectId("647382d97101d54adf994859"), "name" : "abc", "id" : "1", "gender" : "f", "age" : 20, "date" : "Sun May 28 2023 16:35:37 GMT+0000 (UTC)" }
```

```
db.details.insertMany([
```

```
{
  name: "ijk",
  id: "2",
  gender: "m",
  age: 20,
  date: Date()
},
```

```
{
  name: "lmn",
  id: "3",
  gender: "f",
  age: 20,
  date: Date()
}
```

```
])
```

```
db.details.find()
```

```
>db.details.insertMany([
  {
    name: "ijk",
    id: "2",
    gender: "m",
    age: 20,
    date: Date()
  },
  {
    name: "lmn",
    id: "3",
    gender: "f",
    age: 20,
    date: Date()
  }
])
{
  "acknowledged" : true,
  "insertedIds" : [
    ObjectId("647383659b45e4ce9c1bd86c"),
    ObjectId("647383659b45e4ce9c1bd86d")
  ]
}

>db.details.find()
{ "_id" : ObjectId("647382d97101d54adf994859"), "name" : "abc", "id" : "1", "gender" : "f", "age" : 20, "date" : "Sun May 28 2023 16:35:37 GMT+0000 (UTC)" }
{ "_id" : ObjectId("647383659b45e4ce9c1bd86c"), "name" : "ijk", "id" : "2", "gender" : "m", "age" : 20, "date" : "Sun May 28 2023 16:37:57 GMT+0000 (UTC)" }
{ "_id" : ObjectId("647383659b45e4ce9c1bd86d"), "name" : "lmn", "id" : "3", "gender" : "f", "age" : 20, "date" : "Sun May 28 2023 16:37:57 GMT+0000 (UTC)" }
```

## UPDATE:

## GETTING ERROR

```
>db.details.updateOne({name:"abc"},{$set:{id:"7"}})
2023-05-28T16:44:10.944+0000 E QUERY [thread1] SyntaxError: invalid property id @(shell):1:35
```

## DELETE:

```
db.details.deleteOne({ id: "3" })
```

```
db.details.find()
```

```
>db.details.deleteOne({id:"3"})
{ "acknowledged" : true, "deletedCount" : 1 }

>db.details.find()
{ "_id" : ObjectId("647382d97101d54ad994859"), "name" : "abc", "id" : "1", "gender" : "f", "age" : 20, "date" : "Sun May 28 2023 16:35:37 GMT+0000 (UTC)" }
{ "_id" : ObjectId("647383659b45e4ce9c1bd86c"), "name" : "ijk", "id" : "2", "gender" : "m", "age" : 20, "date" : "Sun May 28 2023 16:37:57 GMT+0000 (UTC)" }
>
```

```
db.details.insertOne({
```

```
  name: "abc",
```

```
  id: "1",
```

```
  gender: "f",
```

```
  age:20,
```

```
  date: Date()
```

```
})
```

```
db.posts.deleteMany({ gender: "f" })
```

```
db.details.find()
```

```
>db.details.insertOne({
  name: "abc",
  id: "1",
  gender: "f",
  age:20,
  date: Date()
})
{
  "acknowledged" : true,
  "insertedId" : ObjectId("647385e3d712bd9d202c59f7")
}

>db.details.deleteMany({ gender: "f" })
{ "acknowledged" : true, "deletedCount" : 2 }

>db.details.find()
{ "_id" : ObjectId("647383659b45e4ce9c1bd86c"), "name" : "ijk", "id" : "2", "gender" : "m", "age" : 20, "date" : "Sun May 28 2023 16:37:57 GMT+0000 (UTC)" }
>
```

## GOOGLE DRIVE LINK:

[https://drive.google.com/drive/folders/1LG896PORGiUQkeavqxSQMql1lppE8Qml?usp=share\\_link](https://drive.google.com/drive/folders/1LG896PORGiUQkeavqxSQMql1lppE8Qml?usp=share_link)