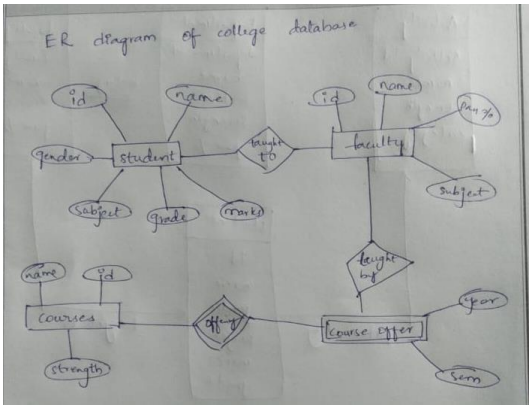
Experiment –1

ER DIAGRAM OF COLLEGE DATABASE

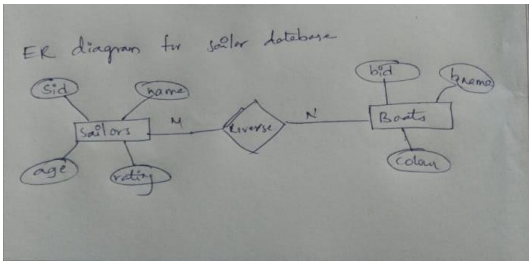
Aim:To draw a er diagram of college database



Experiment-2

ER DIAGRAM FOR SAILOR BOAT DATABASE

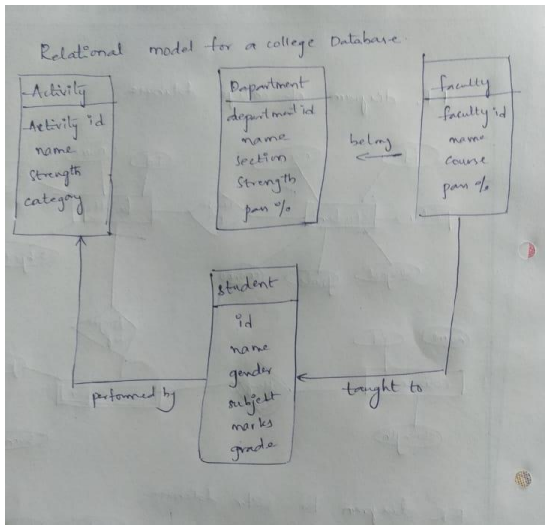
Aim: Draw a ER diagram of sailor boat database



Experiment-3

RELATIONAL MODEL FOR COLLEGE DATABASE

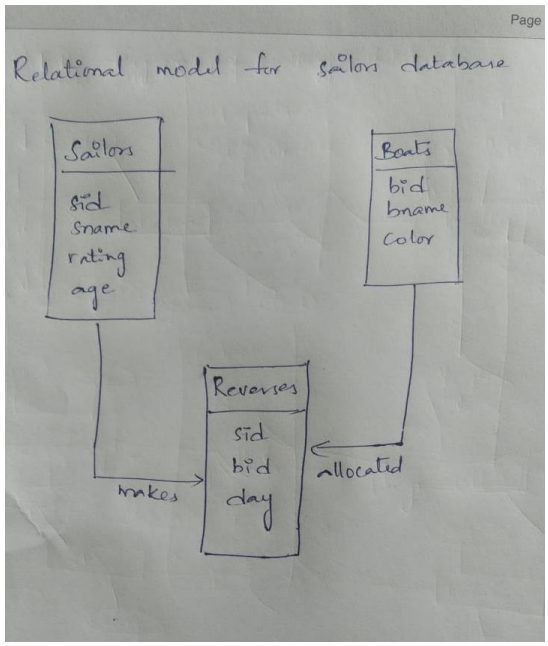
Aim:Draw a relational model for a college database



Experiment-4

RELATIONAL MODEL FOR SAILOR BOAT DATABASE

Aim:draw a relational model for a sailor boat database



Experiment-5

Activity – 1

1.Create schema college

```
2 •  CREATE SCHEMA Bvrit_549;
```

2.Create table-student and attributes are studid,studname,gender,subject,marks ,grade

3.Create table-Faculty and attributes are fac_id,fac_name,course,pass percentage

4.Create table-department and attributes are dept-no,dept_name,section,no of students ,pass percentage

5.Create table-activity and attributes are act_id,act_name,no of students opted,category of activity

```
3 •  CREATE TABLE student(studid integer,studname varchar(20),gender varchar(10),subject varchar(20),marks integer,grade varchar(10));
4 •  desc student;
5 •  CREATE TABLE faculty(fac_id integer,fac_name varchar(20),course varchar(10),passpercetange integer);
6 •  desc faculty;
7 •  CREATE TABLE department(dept_no integer,dept_name varchar(20),section varchar(10),noofstudents integer, passpercentage integer);
8 •  desc department;
```

```
19 •  desc faculty;
```

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

Field	Type	Null	Key	Default	Extra
fac_id	int	NO	PRI	NULL	
fac_name	varchar(20)	YES		NULL	
course	varchar(10)	YES		NULL	
passpercetange	int	YES		NULL	
total_mentor_details	varchar(100)	YES		NULL	

Student table:

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

Field	Type	Null	Key	Default	Extra
studid	int	NO	PRI	NULL	
studname	varchar(50)	YES		NULL	
gender	varchar(10)	YES		NULL	
subject	varchar(20)	YES		NULL	
marks	int	YES		NULL	
grade	varchar(10)	YES		NULL	
address	varchar(50)	YES		NULL	
course	varchar(20)	YES		NULL	

Activity – 2

1.Add address in student table ,change the datatype size for student name and make studid as primary key

```
ALTER TABLE student ADD(address varchar(50));
ALTER TABLE student MODIFY studname varchar(50);
Alter table student add(primary key(studid));
```

2.Add faculty total mentor details ,make4 fac_id as primary key

```
ALTER TABLE faculty ADD(total_mentor_details varchar(100));
Alter table faculty add(primary key(fac_id));
```

3.add no of students in wise and make dept no as primary key

```
ALTER TABLE department ADD(no_of_students_in_wise integer);
Alter table department add(primary key(dept_no));
```

4.Add faculty name and change the size of act_name and act_id as primary key

```
ALTER TABLE activity ADD(faculty_name varchar(30));
ALTER TABLE activity MODIFY act_name varchar(30);
Alter table activity add(primary key(act_id));
```

Experiment-6

Activity – 3

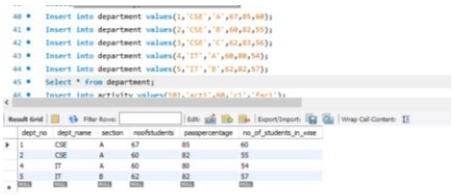
1.Insert 5 instances in each table and display the result

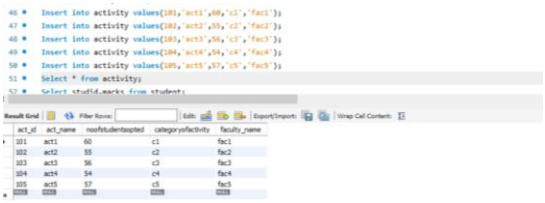
```
Insert into student values(1,'arun','M','DBMS',90,'A','x');
Insert into student values(2,'hari','M','DBMS',80,'A','y');
Insert into student values(3,'poojith','M','DBMS',85,'A','z');
Insert into student values(4,'srest','M','DBMS',60,'B','a');
Insert into student values(5,'nikhil','M','DBMS',75,'B+','b');
Select * from student;
```

studid	studname	gender	subject	marks	grade	address	course
1	arun	M	DBMS	90	A	x	
2	hari	M	DBMS	80	A	y	y
3	poojith	M	DBMS	85	A	z	z
4	srest	M	DBMS	50	B	a	u
5	nikhil	M	DBMS	75	B+	b	v

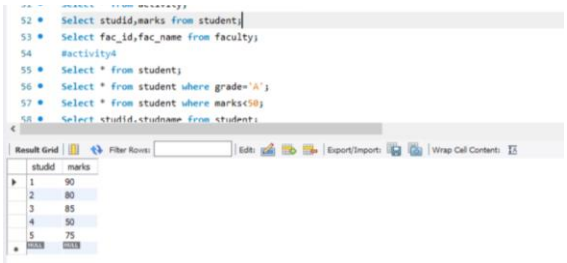
```
Insert into faculty values(1,'aa','x',90,'abc');
Insert into faculty values(2,'bb','y',80,'bby');
Insert into faculty values(3,'cc','Z',80,'ccz');
Insert into faculty values(4,'dd','u',95,'ddu');
Insert into faculty values(5,'ee','v',80,'eev');
Select * from faculty;
Insert into department values(1,'CSE','A',67,85,60);
```

fac_id	fac_name	course	passpercetange	total_mentor_details
1	aa	x	90	abc
2	bb	y	80	bby
3	cc	Z	80	ccz
4	dd	u	95	ddu
5	ee	v	80	eev

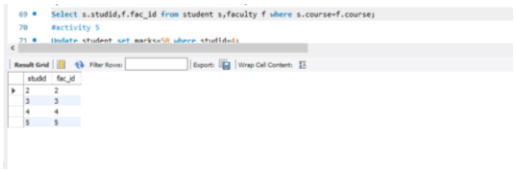




2.Display student no,marks from student table



3.Display faculty no,name from faculty table

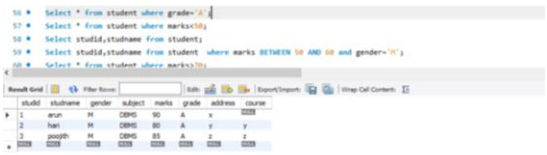


Experiment-7

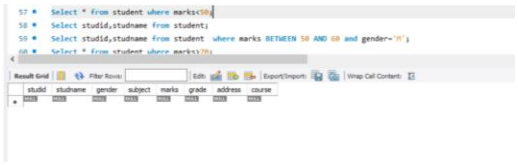
Activity – 4

1.Display 1 to 5 students details

2.Display who got grade A

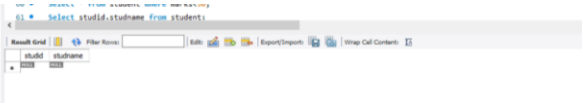


3. Display whose marks is less than 50

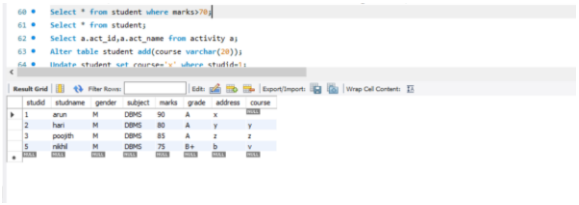


4.Display student id and name

5.Display the student id and name whose mark is 50 to 60and female

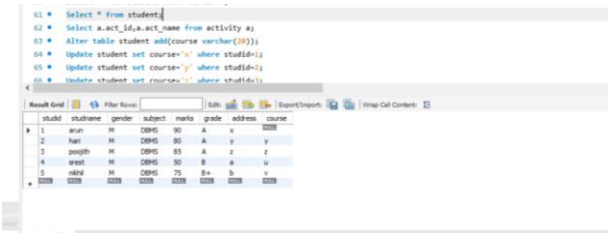


6.Display the list of students who gets greater than 70



7.Delete the failure students

Display complete table



8.Display activity id,name using object

```
Select studname,act_name from student,activity;
Select studname,marks,fac_name from student,faculty;
```

studname	act_name
nikhil	act1
sreet	act1
poojith	act1
hari	act1
arun	act1
nikhil	act2
sreet	act2
poojith	act2
hari	act2
arun	act2

9.Add course to student table then insert values .Display student id,faculty id using course name condition with object

```
Alter
Update student set course='x' where studid=1;
Update student set course='y' where studid=2;
Update student set course='z' where studid=3;
Update student set course='u' where studid=4;
Update student set course='v' where studid=5;
Select s.studid,f.fac_id from student s,faculty f where s.course=f.course;
```

studid	fac_id
2	2
3	3
4	4
5	5

Activity – 5

1.change mark to 50 whose id is 4

```
Update student set marks=50 where studid=4;
```

2.change name whose id is 3

```
Update student set studname='poojith' where studid=3;
```

3. change activity name whose id between 5 to 7

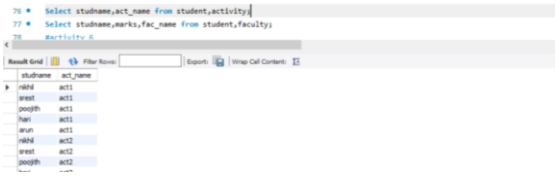
4.change department section to A whosee id is less than 5

```
Update activity set act_name='abc' where act_id BETWEEN 5 AND 7;
Update department set section='A' where dept_no<5;
```

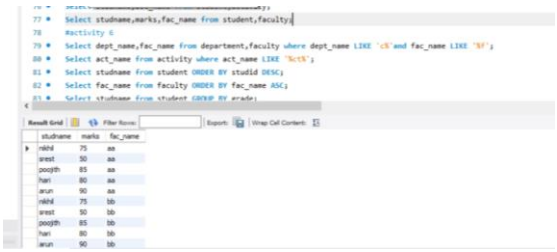
5.delete row who has id = 3

```
Delete from department where dept_no= 3;
```

6.select student name and activity name from student and activity table



7.select student name and mark from student and faculty name from faculty

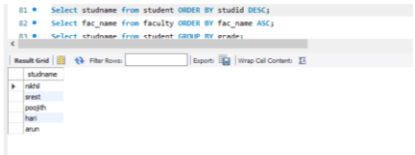


Activity – 6

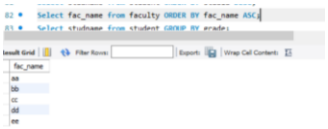
1.select department starts from ‘c’ and faculty name ends with “

2.select activity having characters between ‘ck’

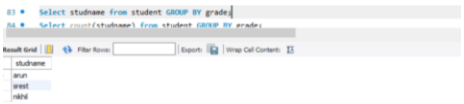
3.display students list descending order of student id



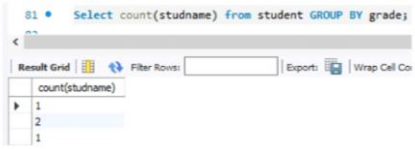
4.display faculty name ascending order



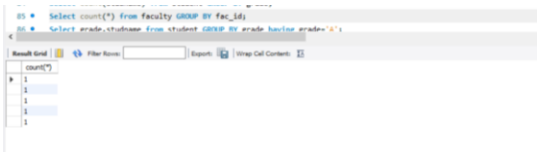
5.display students list based on grade



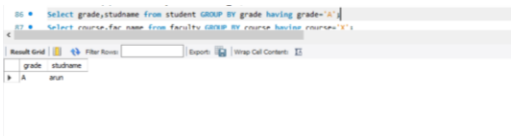
6.display students having grade a using group by



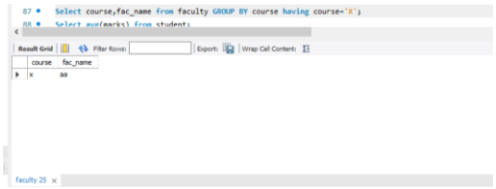
7.group by faculty id and display



8.display the students list whose grade is A using having



9.display the faculty list who are teaching subjest pps



10.Apply aggregate functions in students marks-min,max,sum,count,avg

```
88 •  Select avg(marks) from student;
89 •  Select max(marks) from student;
```

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

avg(marks)
76.0000

```
89 •  Select max(marks) from student;
90 •  Select min(marks) from student;
```

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

max(marks)
90

```
91 •  Select sum(marks) from student;
92 •  Select count(marks) from student;
```

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

sum(marks)
380

```
92 •  Select count(marks) from student;
93
```

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

count(marks)
5

Experiment-8

1.create a tables for sailors,boats and reserves

```
1 •  CREATE SCHEMA sailors_boats;
2 •  CREATE TABLE boat(bid integer,bname varchar(30),color varchar(20));
3 •  CREATE TABLE sailor(sid integer,sname varchar(30),rating integer,age real);
4 •  CREATE TABLE reserves(sid integer,bid integer,DAY DATE);
5 •  desc sailor;
6 •  desc boat;
```

Field	Type	Null	Key	Default	Extra
sid	int	YES		NULL	
sname	varchar(30)	YES		NULL	
rating	int	YES		NULL	
age	double	YES		NULL	

Field	Type	Null	Key	Default	Extra
bid	int	YES		NULL	
bname	varchar(30)	YES		NULL	
color	varchar(20)	YES		NULL	

Field	Type	Null	Key	Default	Extra
sid	int	YES		NULL	
bid	int	YES		NULL	
DAY	date	YES		NULL	

2.Insert 5 values in each table

Sailor table:

```
8 •  Insert into sailor values(22,'Dustin',7,45.0);
9 •  Insert into sailor values(29,'Brutus',1,33.0);
10 •  Insert into sailor values(31,'Lubber',8,55.5);
11 •  Insert into sailor values(32,'Andy',8,25.5);
12 •  Insert into sailor values(58,'Rusty',10,35.0);
```

Boat table:

```
Insert into boat values(101,'Interlake','blue');
Insert into boat values(102,'Interlake','red');
Insert into boat values(103,'Clipper','green');
Insert into boat values(104,'Marine','red');
```

Reserve table:

```
Insert into reserves values(22,101,'98/10/10');
Insert into reserves values(22,102,'98/10/10');
Insert into reserves values(31,102,'98/10/11');
Insert into reserves values(64,101,'98/5/9');
Insert into reserves values(74,103,'98/8/9');
```

3.Display all records

```
Select * from sailor;
```

sid	sname	rating	age
22	Dustin	7	45
29	Brutus	1	33
31	Lubber	8	55.5
32	Andy	8	25.5
58	Rusty	10	35

```
Select * from boat;
```

Result Grid | Filter Rows:

bid	bname	color
101	Interlake	blue
102	Interlake	red
103	Clipper	green
104	Marine	red

```
Select * from reserves;
```

Result Grid | Filter Rows:

sid	bid	DAY
22	101	1998-05-09
22	102	1998-10-11
31	102	1998-10-11
64	101	1998-05-09

4.find the names and ages of the sailors

```
Select sname,age from sailor;
```

Result Grid | Filter Rows:

sname	age
Dustin	45
Brutus	33
Lubber	55.5
Andy	25.5
Rusty	35

5.find all the sailors with a rating above 8

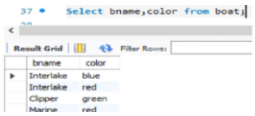
```
Select sid from sailor where rating>=8;
```

Result Grid | Filter Rows: | Export:

sid
31
32
58

6.find all sailors name with rating above 7 & age above 25

7.Display all the names and colors of the boat



8.Find all the boats with red color

9.find the names of sailors who have reserved boat number 101

10.find the sids of sailor who have reserved pink boat

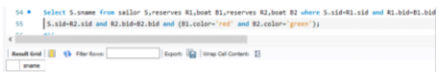
11.find names of sailors who have reserved red boat

12.find the colors of boat reserved by same name(provide any name in the table)

13.find the names of the sailors who have atleast one boat

14.find the names of the sailor who have reserved two different boats

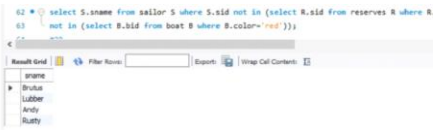
15.find the names of the sailors who have reserved a red or a green boat



16.find the names of sailors who have reserved both a red and a green boat

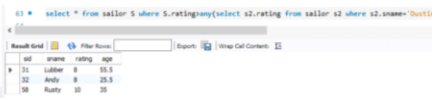
17.find the names of sailors who have reserved boat 3

19.find the names of sailors who have not reserved a red boat(nq)



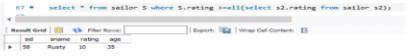
20.find the names of sailors who have reserved boat number 103(exists)

21.find the sailors whose rating is better than some sailor called name

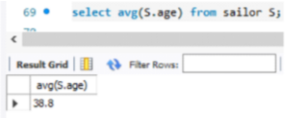


22.find sailors whose rating is better than every sailor called name

23.find the sailors with the highest rating.



24.find the average age of all sailors



25.find the average age of sailors with a rating of 10.

```
71 •  select avg(S.age) from sailor S where S.rating=10;
```

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

avg(S.age)
35

26.count the number of sailors

```
73 •  select count(*) from sailor S;
```

Result Grid | Filter Rows:

count(*)
5

27.count the number of different sailor ratings

```
73 •  select count(*) from sailor S;
```

Result Grid | Filter Rows:

count(*)
5

28.find the name and age of the oldest sailor

```
77 •  select S.sname, S.age from sailor S where (select max(S2.age) from sailor S2)=S
```

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

sname	age
Lubber	55.5

29.find the name of sailors who are older than the oldest sailor with a rating of 10.

```
79 •  select S.sname from sailor S where S.age> (select max(S2.age) from sailor S2 where S2.rating=
```

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

sname
Dustin
Lubber

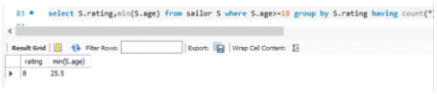
30.find the age of the youngest sailor for each rating level.

```
81 •  select S.rating,min(S.age) from sailor S group by S.rati
```

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

rating	min(S.age)
7	45
1	33
8	25.5
10	35

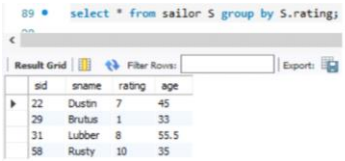
31.find the age of the youngest sailor who is eligible to vote(I.e.,atleast 18 years old)for each rating level with atleast two such sailors.



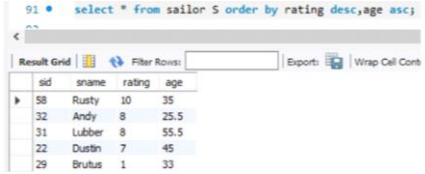
32.for each red boat,find the number of reservation for this boat.

33.find all sailors name according to names.

34.find all sailors details according to rating



35.find all sailors details according to rating (highest first),if ratings are same then according to age.



Experiment-9

Views

1.create a table student attributes stdno,name,dept

2.Display

```
create table students(studno integer,stud_name varchar(20),dept varchar(20));
desc students;
```

Field	Type	Null	Key	Default	Extra
studno	int	YES		NULL	
stud_name	varchar(20)	YES		NULL	
dept	varchar(20)	YES		NULL	

3.create view and select

```
create view stud_view as select studno,stud_name,dept from students;
select * from stud_view;
```

studno	stud_name	dept

4.Insert values

```
Insert into stud_view values(549,"pooja","CSE");
Insert into stud_view values(512,"pooja","CSE");
Insert into stud_view values(550,"Akshaya","CSE");
alter view stud_view as select studno,stud_name,dept from students where studno=512;
```

studno	stud_name	dept
512	Pooja	CSE

5.create view with check option

Insert values and alter tables

```
alter view stud_view as select studno,stud_name,dept from students where studno=512;
create view student_view as select studno,stud_name,dept from students with check option;
Insert into student_view values(553,"harila","CSE");
alter view student_view as select studno,stud_name,dept from students where studno=553;
drop view student_view;
delete from stud_view where studno=549;
```

Experiment-10

Triggers

1.create a table with attribute sname and another table with newname

```
Create table test1(sname varchar(20));
Create table test_audit1(newname varchar(20));
```

2.create a trigger to insert second table before inserting the first table

```
Create trigger before_test_update
    before insert on test1
    for each row insert into test_audit1
    set newname = new.sname;
Insert into test1 Values ("Harila");
select * from test1;
select * from test_audit1;
```

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

newname
Harila

- Create a table for account details with attributes account number and an amount

```
Create trigger before_insert_
    before insert on acc_details
    for each row
    begin
    if (new.amount < 0) then
        set new.amount = 0;
    elseif (new.amount > 100) then
        set new.amount = 100;
    end if;
    end //
delimiter ;
```

```
Insert into acc_details values (100, -14.98);
Select * from acc_details;
```

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

accno	amount
100	0.00

```
Insert into acc_details values (200, 500.00);
Select * from acc_details;
```

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

accno	amount
100	0.00
200	100.00

```
Insert into acc_details values (300, -14.98);
Select * from acc_details;
```

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

accno	amount
100	0.00
200	100.00
300	0.00

Experiment-11

Procedures

1.Create a table with attributes students and marks

Insert values into tables

```
1  Create table proc (id int, marks int);
2  Insert into proc values (1, 30), (2, 40), (3, 50), (5, 60), (6, 70);
3  select * from proc;
```

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

id	marks
1	30
2	40
3	50
5	60
6	70

- Create a procedure to get the merit students(marks>50)
- Create a procedure to get marks of a given id(using in)

```
52  delimiter //
53  create procedure get_merit_student()
54  begin
55      select * from proc where marks > 50;
56      select count(id) as Total_students from proc;
57  end //
58  delimiter ;
59  call get_merit_student();
```

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

Total_students
5

Result Grid | Fil

id	marks
5	60
6	70

```
61  delimiter //
62  create procedure get_marks(IN stdid int)
63  begin
64      select marks from proc where (id = stdid);
65  end //
66  delimiter ;
67  call get_marks(6);
```

Experiment-12

Cursors

- Create a table with attributes students and marks
- Insert values into tables

```
CREATE TABLE curs(id int, marks int);
INSERT INTO curs (id,marks)values(1,30),(2,40),(3,50),(5,60),
DELIMITER \\
```

- Create a procedure and fetch the marks of given id using a cursor

```
DELIMITER \\
CREATE PROCEDURE cur_marks(id1 int)
BEGIN
    DECLARE m1 int;
    DECLARE cur1 CURSOR FOR SELECT marks FROM curs WHERE id =
    OPEN cur1;
    FETCH cur1 INTO m1;
    SELECT m1;
    CLOSE cur1;
END \\
DELIMITER ;
CALL cur_marks(2);
```

- Create a procedure and fetch the highest marks using a cursor

```
DELIMITER \\
CREATE PROCEDURE cur_mark1()
BEGIN
    DECLARE m1 int;
    DECLARE cur1 CURSOR FOR SELECT max(marks) FROM cu
    OPEN cur1;
    FETCH cur1 INTO m1;
    SELECT m1;
    CLOSE cur1;
END \\
DELIMITER ;
CALL cur_mark1();
```

Result Grid

	m1
▶	70