PROGRAM-1

AIM

Familiarization of DDL commands and constraints

Consider the schema

Student(admission_no, stud_name)
Membership(member_id, admission_no)
Book(book_id, book_name, author, book_type)
BookType(type_id, type)
Bookissue(issue_id, issue_date, member_id, book_id)

1. Create the above tables and provide appropriate integrity constraints and insert few records into the table
2. Add the column 'program' to student table
3. Modify the column width of book_name and author of book relation
4. Drop the BookType table

QUERIES :

Create database program;
use program;

1. Create table student(admission_no int primary key, stud_name varchar(20));
   desc student;

Field	Type	Null	key	Default	Extra
admission_no	int	NO	PRI	NULL	
stud_name	varchar(20)	yes		NULL	

Create table membership (member_id int primary key, admission_no int, foreign key (admission_no) references student(admission_no));
desc membership;

Field	Type	Null	key	Default	Extra
member_id	int	NO	PRI	NULL	
admission_no	int	YES	MUL	NULL	

Create table book(book_id int primary key not null, book_name varchar(20), book_author varchar(20), book_type varchar(20));
desc book;

Field	Type	Null	key	Default	Extra
book_id	int	No	PRI	NULL	
book_name	varchar(20)	yes		NULL	
book_author	varchar(20)	yes		NULL	
book_type	varchar(20)	yes		NULL	

Create table booktype (type_id int primary key not null, type varchar(20));
desc booktype;

Field	Type	Null	key	Default	Extra
type_id	int	No	PRI	NULL	
type	varchar(20)	yes		NULL	

Create table bookissue (issueid int primary key, issuedate date, member_id int, foreign key (member_id) references membership (member_id), book_id int, foreign key (book_id) references book (book_id));
desc bookissue;

Field	Type	Null	key	Default	Extra
issueid	int	No	PRI	NULL	
issuedate	date	yes		NULL	
member_id	int	yes	MUL	NULL	
book_id	int	yes	MUL	NULL	

Insert into student values (1, "anu");
Insert into student values (2, "diya");
Insert into student values (3, "kiran");

Insert into membership values (1025, 1);
Insert into membership values (1001, 2);
Insert into membership values (1041, 3);

Insert into book values (23, 'wings of fire', 'APJ Abdul kalam', 'autobiography');
Insert into book values (40, 'Ikigai', 'Hector garcia', 'novel');
Insert into book values (61, 'C programming', 'Balaguru swamy', programming');

Insert into booktype values (35, 'programming');
Insert into booktype values (36, 'autobiography');

Insert into booktype values (37, 'NOVEL');

Insert into bookissue values (365, '2018-07-12', 1001, 23);
insert into bookissue values (980, '2019-05-30', 1002, 61);
insert into bookissue values (752, '2006-10-10', 1003, 40);

2. Alter table Student
   add column programme varchar(20);

3. Alter table book modify book_name varchar(100);
   Alter table book modify book_author varchar(100);

4. Drop table booktype;

(2 to 9), Use the following relations. If needed, make necessary changes the schema.

Department	
Dname	NOT NULL, unique
Dnumber	Primary key
ManagerID	Refers to EmployeeID of employee table
Mgr_start_date	Not NULL

Employee	
Name	Not NULL
EmployeeID	Primary key
BirthDate	
HouseName	
Gender	
Salary	5000 to 25000
SuperEID	Refers to Employee itself
Dnumber	Refers to Dnumber of Department table

DeptLocations	
Dnumber	Dnumber refers to Dnumber of Dept
DLocation	table. also Dno, Dloc are primary key

Project	
Pname	Not null
Pnumber	Primary key
Plocation	
Dnumber	Not null, refers to department table

Works-on	
EmployeeID	Refers to employee table
DependNaPnumber	Refers to project
Hours	Not NULL

Dependent	
EmployeeID	Refers to employee table
DependName	Not null
Sex	
Birthdate	
Relationship	
	Empld, DependName together primary key

QUERIES:

```
Create database example;
Use example;
```

→ Create table department
(dname varchar(20) not null unique,
dnumber int primary key,
managerid int,
mngr_start_date date not null);

desc department;

Field	Type	Null	key	Default	Extra
dname	varchar(20)	No	UNI	NULL	
dnumber	int	No	PRI	NULL	
managerid	int	yes		NULL	
mngr_start_date	date	no		NULL	

→Create table employee (ename varchar(20), eid int primary key, housename varchar(20), salary int, superid varchar(20), dnumber int, foreign key (dnumber) references department (dnumber), birth_date date);

desc employee;

Field	Type	Null	Key	Default	Extra
ename	varchar(20)	Yes		NULL	
eid	int	No	PRI	NULL	
housename	varchar(20)	Yes		NULL	
gender	varchar(20)	Yes		NULL	
salary	int	Yes		NULL	
superid	varchar(20)	Yes		NULL	
dnumber	int	Yes	MUL	NULL	
birth_date	date	Yes		NULL	

→Create table dlocation (dnumber int, foriegn key (dnumber) references department (dnumber), dlocation varchar(20) not null, primary key (dnumber, dlocation));
desc dlocation;

Field	Type	Null	key	Default	Extra
dnumber	int	No	PRI	NULL	
dlocation	varchar(20)	No	PRI	NULL	

→ Create table project (pname varchar(20) not null, pnumber int primary key, plocation varchar(20), dnumber int, foreign key (dnumber) references department (dnumber));
desc project;

Field	Type	Null	key	Default	Extra
pname	varchar(20)	No		NULL	
pnumber	int	No	PRI	NULL	
plocation	varchar(20)	yes		NULL	
dnumber	int	yes	MUL	NULL	

→ Create table works_on (eid int, foreign key (eid) references employee (eid), pnumber int, foreign key (pnumber) references project (pnumber), hour time not null);
desc works-on;

Field	Type	Null	key	Default	extra
eid	int	yes	mul	null	
pnumber	int	yes	mul	null	
hours	time	no		~~not~~ null	

→ Create table dependent (eid int, foriegn key (eid) references employee (eid), depend_name varchar(20) not null, sex varchar(20), b_date date, relationship varchar(20), primary key (eid, depend_name));
desc dependent;

Field	Type	Null	Key	Default	Extra
eid	int	No	PRI	NULL	
depend_name	varchar(20)	No	PRI	NULL	
sex	varchar(20)	yes		NULL	
b_date	date	yes		NULL	
relationship	varchar(20)	yes		NULL	

Insert into department values ('physics' 12, '2012-04-15', 100);
Insert into department values ('teshrig', 13, '2020-10-11', 102);
Insert into department values ('automahon,' 5,' 2019-11-10', 101);

1.

dname	dnumber	managerid	mngr_start_date
physics	12	100	2012-04-15
testing	13	102	2020-10-11
automation	5	101	2019-11-10
computer	2	101	2010-03-12

2.

ename	eid	housename	gender	salary	superid	dnumber	birth_date
Yunho	101	KQ villa	male	19950	101	2	1999-03-23
Yesi	102	villa	female	12000	102	12	1994-07-11
Joon	103	HB house	male	5600	103	13	1995-06-12
bam	104	star	male	7000	104	2	1997-04-06
lisa	105	yh	female	11000	105	13	1997-03-04

3.

ename	eid	housename	gender	salary	superid	dnumber	birth_date
Yunho	101	KQ villa	male	15000	101	2	1999-03-23
Yesi	102	Villa	female	15000	102	12	1994-07-11
Lisa	105	Yh	female	15000	105	13	1997-03-04

PROGRAM-2

AIM

Execution of DML commands

1. Insert a single record into department table
2. Insert more than a record into employee table using a single insert command
3. Update the employee table to set the salary of all employees to Rs 15000/- who are getting a salary > 10000
4. Move a project 'P1' of department no D1 to another department D2
5. Move Delete only those who are working on a particular project say 'P1'.

QUERIES

1. Insert into department values ('computer', 2, '2010-03-12', 101);
   Select * from department;

2. Insert into employee values
   ('Yunho', 101, 'KQ villa', 'Male', 19950, 101, 2, '1999-03-2023'),
   ('Yesi', 102, 'villa', 'female', 12000, 102, 12, '1994-07-11'),
   ('joon', 103, 'HB house', 'male', 5600, 103, 13, '1995-06-12');
   ('bam', 104, 'star', 'male', 7000, 104, 2, '1997-04-06'),
   ('lisa', 105, 'yh', 'female', 11000, 105, 13, '1997-03-04');
   Select * from employee;

pname	pnumber	plocation	dnumber
P1	1	Mumbai	13
P1	3	chennai	13

eid	pnumber	hours
101	3	06:00:00
102	5	07:30:00
105	2	04:00:00
104	1	02:30:00
103	3	07:00:00
102	3	02:00:00

eid	pnumber	hours
101	3	06:00:00
102	5	07:30:00
104	1	02:30:00
103	3	07:00:00
102	3	02:00:00

3. Update employee set salary=15000 where salary>10000;
   Select * from employee where salary=15000;

4. Insert into project values
   ('P1',1,'Mumbai',2),('p2',2,'kochi',13),
   ('p1',3,'chennai',2),('p3',4,'chennai,12),
   ('p4',5,'Delhi',13);
   Select * from project;
   → Update project set dnumber=13 where
5. pname='p1';
   Select * from project where pname='p1';

5. Insert into works_on values(101,3,'06:00:00');
   Insert into works_on values(102,5,'07:30:00');
   Insert into works_on values(105,2,'04:00:00');
   Insert into works_on values(104,1,'02:30:00');
   Insert into works_on values(103,3,'07:00:00');
   Insert into works_on values(102,3,'02:00:00');
   Select * from works_on;
   → delete from works_on where pnumber=2;
   Select * from works_on;

1.

ename	eid	housename	gender	salary	superid	dnumber	birth_date
Joon	103	HB house	male	103	5600	13	1995-06-12
bam	104	star	male	104	7000	2	1997-04-06
Yunho	101	kq villa	male	101	15000	2	1999-03-23
Yeri	102	villa	female	102	15000	12	1994-07-11
Lisa	105	Yh	female	105	15000	13	1997-03-04

2.

ename	eid	housename	gender	salary	superid	dnumber	birth_date
Yunho	101	kq villa	male	15000	101	2	1999-03-23
bam	104	star	male	7000	104	2	1997-04-06

3.

ename	birth_date
Joon	1995-06-12
Lisa	1997-03-04

PROGRAM-3

AIM

Retrieving data using select commands

1. List the records in the employee table order by salary in asending / descending order
2. Display only those employees whose dnumber is 30
3. Retrieve the name and birthdate of Employee working in a particular in a particular department
4. For every project located in 'cochin', list the project number, the controlling departmentno and the department manager's name, Housename and birth date
5. List the employees who work in more than one project

QUERIES :

1. Select * from employee order by salary asc;

2. Select * from employee where dnumber=2; //30

3. Select ename, birth_date from employee where dnumber=13;

4. Select plocation, dname as 'manager name', housename, birth_date, pnumber from department as d, project as p, employee where plocation='chennai' and p.dnumber = d.dnumber and d.managerid=eid;

5.

eid
102

5. Select the eid from works-on group by eid having count(eid)>1;

1.

sum(salary)	max(salary)	min(salary)	avg(salary)
57600	15000	5600	11520.0000

2.

count(pnumber)	dnumber
1	12
4	13

3

count(eid)	dnumber
2	2
1	12
2	13

4

max(salary)
15000

PROGRAM - 4

AIM

Aggregate functions

1. Find the sum of salaries of all employees, the maximum salary, minimum and average salary
2. Count the no. of projects handled in each department
3. Count the number of employees working in each department
4. Find the department number and maximum salary of those departments where minimum salary greater than 10000.

QUERIES

1. Select max(salary), sum(salary), min(salary), avg(salary) from employee;

2. Select count(pnumber), dnumber from project group by dnumber;

3. Select count(eid), dnumber from employee group by dnumber;

4. Select max(salary) from employee group by dnumber having min(salary) >1000;

1.

ename	eid	housename	gender	salary	superid	dnumber	birth_date
anu	110	Land villa	Female	20950	101	2	1988-03-23

2

ename	eid	housename	gender	salary	superid	dnumber	birth_date
anu	110	Land villa	female	20950	101	2	1988-03-23

PROGRAM-5

AIM

String functions

1. Retrieve all employees whose name begins with 'A'.
2. Find all employees who were born during 1980's

QUERIES

-> Insert into employee values
('anu', 110, 'landvilla', 'Female', 20950, 101, 2, '1988-03-23');

1. Select * from employee
where ename like 'a%';

2. Select * from employee where birth_date
between '1980-01-01' and '1989-12-31';

ename	eid	housename	gender	salary	superid	dnumber	birth_date
yeri	102	villa	female	15000	102	12	1994-07-11
bam	104	star	male	7000	104	2	1997-04-06
lisa	105	yh	female	15000	105	13	1997-03-04
anu	110	land villa	female	20950	101	2	1998-03-23

ename	service
Yunho	12
Yeri	10
Joon	2
Bam	12
Lisa	2

PROGRAM-6

AIM

Date functions

1. List all employees who age lies between 25-45 years
2. Calculate the service period of all managers

QUERIES

1. Select * from employee where birth_date between date_sub(curdate(), interval 45 year) and date_sub(curdate(), interval 25 year);

2. Select employee.ename, timestamp diff(year, department.mngr_start_date, CURDATE()) as service from employee, department where employee.eid = department.managerid;

1.

pnumber
3

PROGRAM-7

AIM

Union, intersection, set difference

1. Make list of all project numbers for projects that involve an employee whose name is "Raju" either as a worker or as a manager of the department that controls project

QUERIES

→ Update employee set ename='raju' where eid=101;

1. Select pnumber from works_on where eid in (select eid from employee where ename='raju');

1.

eid	depend_name	sex	b_date	relationship
20	anu	Female	1991-07-03	client
14	raj	male	1994-06-11	client
11	maya	Female	1990-03-04	client
104	bam	male	1997-04-03	client
102	yeri	Female	1994-07-11	client

2.

depend_name
bam
yeri

PROGRAM - 8

AIM

Nested queries and join operation

1. Retrieve the name of each employee who has a dependent with the same name and is the same sex as employee.
2. Retrieve the names of employee who have no dependents
3. List the names of all managers who have atleast one dependant

QUERIES

-> Insert into dependent values(20,'anu','female', '1991-07-03','client');
Insert into dependent values(14,'raj','male', 1984-06-11','client');
Insert into dependent values(11,'maya','female', 1990-03-04','client');
Insert into dependent values(104,'bam','male', 1997-04-03','client');
Insert into dependent values(102,'yeri','female', 1994-07-11','client');
Select * from dependent;

1. Select depend_name from dependent join employee.ename = dependent.depend_name and dependent.eid = employee.eid and dependent.sex = employee.gender;

2.

ename
raju
joon
lisa
anu

3.

ename
Bam
Yeri

2. Select ename from employee
where employee.eid not in (select dependent.eid from dependent);

3. Select employee.ename from employee where employee.eid in (select dependent.eid from dependent, department where dependent.eid = department.manager_id);

1.

dnumber	max(salary)	min(salary)	avg(salary)
2	20950	7000	14316.6667
12	15000	15000	15000.0000
13	15000	5600	10300.0000

2.

ename	pname	hours
raju	P1	06:00:00
yeri	P4	07:30:00
bam	P1	02:30:00
joon	P1	07:00:00
yeri	P1	02:00:00

PROGRAM-9

AIM

Database views

1. Create a view to display the department no, minimum salary, maximum salary and average salary in each department
2. Create a view to display the employee name, project name and hours worked by her/him
3. Update the project name of above view (Q1) from pname='Productx' to pname='Producty'.
4. Drop the above specified view
5. Create a view work-info to display the employee id, employee name, project no. of all employees whose working hour is greater than 0

QUERIES

1. Create view deptview as
select dnumber, max(salary), min(salary), avg(salary) from employee group by dnumber;
Select * from deptview;

2. Create view empview as select employee.ename, project.pname, works-on.hours from works-on, employee, project where works-on.eid = employee.eid and works-on.pnumber = project.pnumber;
Select * from empview;

3.

ename	pname	hours
raju	P1	06:00:00
yesi	productY	07:30:00
bam	P1	02:30:00
joon	P1	07:00:00
yesi	P1	02:00:00

5.

ename	eid	pnumber
raju	101	3
yesi	102	5
bam	104	1
joon	103	3
yesi	102	3

3. Update empview set pname = 'productY'
where pname = 'P4';

4. Drop view empview;

5. Create view workson as select employee.ename, employee.eid, works_on.pnumber from employee, works_on where employee.eid = works_on.eid and works_on.hours > '00:00:00';
Select * from workson;

PROGRAM-10

AIM

Stored procedures and functions

1. Create a procedure to that generate all the prime numbers below the given number and count the no of prime numbers generated

Query :

```
delimiter $$
Create procedure prime2(IN n int, OUT result varchar(20))
Begin
declare j,i, flag, int;
set j:=2;
set result:=' ';
while(j<n) do
set i:=2;
set flag:=0;
while(i<=j) do
if(j%i=0) then
set flag:= flag+1;
end if;
set i:= i+1;
end while;
if(flag=i) then
set result:=concat(result,j,',');
end if;
set j:=j+1;
end while;
```

OUTPUT

substr(@result, 1, length(@result - 1)
2, 3, 5, 7, 11, 13, 17, 19,

```
End
$$

-> call prime2(20, @result);
  Select substr(@result, 1, length(@result) - 1);
  $$
```

2. Consider the employee table (emp-id, ename, basic, dept) and insert 10 records to the table.
Write a procedure to update salary of an employee accepting emp-id and rate as parameters. Also fetch the names and salaries of the five highest-paid employees with their department.

Query :

```
create table employee (eid int, ename varchar(10),
basic int, dept varchar(10), primary key (eid));
desc employee;
```

Field	Type	Null	key	Default	Extra
eid	int	No	PRI	Null	
ename	varchar(10)	yes		Null	
basic	int	yes		Null	
dept	varchar(10)	yes		Null	

```
Insert into employee values (100, 'amal', 10000, 'cs').
Insert into employee values (101, 'ajay', 15000, 'phy');
Insert into employee values (102, 'sujith', 10000, 'cs');
Insert into employee values (103, 'pranav', 10000, 'testing');
Insert into employee values (104, 'aswin', 15000, 'analysis');
Insert into employee values (105, 'anu', 25000, 'analysis');
Insert into employee values (106, 'daya', 20000, 'design');
Insert into employee values (107, 'niya', 30000, 'design');
Insert into employee values (108, 'nithin', 20000, 'testing');
Insert into employee values (109, 'priya', 25000, 'cs');
Select * from employee;
```

OUTPUT

eid	ename	basic	dept
100	amal	10000	cs
101	ajay	15000	phy
102	sujith	10000	cs
103	pranav	10000	testing
104	aswin	15000	analysis
105	anu	25000	analysis
106	daya	20000	design
107	riya	30000	design
108	nithin	20000	testing
109	priya	25000	cs

1.

ename	basic	dept
riya	30000	design
anu	25000	analysis
priya	25000	cs
ajay	21000	phy
daya	20000	design

2

ename	basic	dept
riya	30000	design
ajay	27000	phy
anu	25000	analysis
priya	25000	cs
daya	20000	design

```
delimiter $$
create proudure salary (in eid int(10), rate int)
begin
update employee set employee.basic = employee.basic +
rate where employee.eid = eid;
select ename, basic, dept from employee order by
basic desc limit 5;
end
$$
```

1→ `call salary(100,500);`
`$$`

2→ `call salary(101, 6000);`
`$$`

5. Consider the relations
Customer (cust_id, cust_name, address)
Order (ord_no, cust_id, ord_date, ship_date, status, comments)
Field status take values like "delivered", "pending", "shipped", "cancelled". Insert few records and create a stored procedure to return the count of orders delivered, pending, shipped and cancelled

Query:
Create table customer (cid int, cname varchar(10), address varchar(10), primary key (cid));
desc customer;

Field	Type	Null	key	Default	Extra
cid	int	NO	PRI	Null	
cname	varchar(10)	yes		Null	
address	varchar(10)	yes		Null	

Insert into customer values (100, 'anu', 'calicut');
Insert into customer values (101, 'riya', 'kochi');
Insert into customer values (102, 'karan', 'chennai');
Insert into customer values (103, 'Jinu', 'Banglore');
Select * from customer;

DUTPUT

Customer

cid	cname	address
100	anu	calicut
101	riya	kochi
102	karan	chennai
103	Jinu	Banglore

Corder

cono	cid	codate	cshipdate	status	comments
200	101	2020-05-01	2020-06-01	delivered	excellent
201	101	2021-04-10	2021-04-12	pending	average
202	102	2021-06-11	2021-06-13	cancelled	poor
203	103	2022-10-11	2022-10-12	shipped	above average
204	100	2022-11-12	2022-11-14	shipped	above average
205	102	2023-11-12	2023-11-14	pending	average
206	101	2020-03-17	2020-03-20	cancelled	poor

```
Create table corder (cono int, cid int, codate date,
cshipdate date, status varchar(20), comments varchar
(20), foreign key (cid) references customer (cid),
primary key (cono));
desc corder;
```

Field	Type	Null	key	Default	Extra
cono	int	No	PRI	NULL	
cid	int	yes	MUL	NULL	
codate	date	yes		NULL	
cshipdate	date	yes		NULL	
status	varchar(20)	yes		NULL	
comments	varchar(20)	yes		NULL	

```
Insert into corder values (200, 100, '2020-05-01', '2020-06-01',
'delivered', 'excellent');
Insert into corder values (201, '101', '2021-04-10', '2021-04-12',
'pending', 'average');
Insert into corder values (202, 102, '2021-06-11', '2021-06-
13', 'cancelled', 'poor');
Insert into corder values (203, '103', '2022-10-12', '2022-10-13',
'shipped', 'above average');
Insert into corder values (204, 100, '2022-11-12', '2022-11-14',
'shipped', 'above average');
Insert into corder values (205, 102, '2023-11-12', '2023-11-
14', 'pending', 'average');
Insert into corder values (206, 101, '2020-03-17', 2020-03-
20', 'cancelled', 'poor');
```

count('pending')
2

count('cancelled')
2

count('shipped')
2

```
Select * from corder;

DELIMITER $$
Create proedure cust_order()
Begin
select count('delivered') from corder where status=
'delivered';
Select count('pending') from corder where status=
'cancelled';
Select count('shipped') from corder where status=
'shipped';
END $$
DELIMITER;

→Call cust_order();
```

OUTPUT

factorial(4)
24

4. Create a function to find the factorial of a number passed as parameter

Query :

```
delimiter $$
Create function factorial (n INT)
Returns INT(11)
Begin
Declare factorial INT;
Set factorial = n;
IF n<=0 THEN
return 1;
end if;
bucle : LOOP
Set n = n-1 ;
If n<1 Then
Leave bucle;
end if;
Set factorial = factorial * n;
end loop bucle;
Return factorial;
END
$$

-> Select factorial(4);
$$
```

5. Write a function to check a number is perfect, abundant or deficient

Query:

```
delimiter $$
Create function checknum2(n int)
returns text
begin
declare s int;
declare i int;
Set s=0;
Set i=1;
lb: while i<n do
if n%i=0 then
Set s=s+i;
end if;
Set i=i+1;
end while lb;
if s=n then
return 'Perfect';
elseif s>n then
return 'abundant';
else
return 'deficient';
end if;
end;
$$
```

OUTPUT

checknum2(6)
Perfect

checknum2(8)
Deficient

checknum2(12)
Abundant

→ Select checknum2(6);
$$

→ Select checknum2(8);
$$

→ Select checknum2(12);
$$

PROGRAM-11

AIM

Triggers

1- Create a table student (id, name, dob) and insert few records. Create trigger to prevent updating and deletion from the table

Query:

```
Create table student (sid int, sname varchar(10),
dob date, primary key (sid));
describe student;
```

Field	Type	null	key	Default	Extra
sid	int	No	PRI	NULL	
sname	varchar(10)	yes		NULL	
dob	date	yes		NULL	

```
Insert into student values(1,'ab','2000/05/01');
Insert into student values (2,'abi','2000/06/01');
Select * from student;

delimiter $$
Create trigger stt
before delete on student for each row begin
SIGNAL SQLSTATE '02000'
set message_text = 'cant update/delete';
end;
$$
```

OUTPUT

1.

Field	Type	Null	key	Default	Extra
sid	int	NO	PRI	NULL	
sname	varchar(10)	yes		NULL	
dob	date	yes		NULL	

2.

sid	sname	dob
1	ab	2000-05-01
2	abi	2000-06-01

3. Error 1643 (02000): can't update/delete

1 → describe student;
¢¢

2 → Select * from student;
¢¢

3 → Delete from student where sid=2;

2. Consider the schema
Product (prod-id, prod_name, price, quantity-available)
Sale (sale_id, prod_id, quantity)
Create a trigger to update the quantity in stock after each sale

Query:
Create table product (pid, varchar(10), pname varchar(10), price int, q-avail int, primary key (pid));
desc product;

Field	Type	Null	key	Default	Extra
pid	varchar(10)	NO	PRI	Null	
pname	varchar(10)	yes		Null	
price	int	yes		Null	
q-avail	int	yes		Null	

Insert into product values('p1','orange',100,100);
Insert into product values('p2','apple',100,100);
Insert into product values('p3','papaya',100,100);
Insert into product values('p4','grape','100',100);
Insert into product values('p5','water',100,100);
Select * from product;

Create table sale (sid varchar(10), pid varchar(10), foreign key (pid) references product (pid), quantity int, primary key (sid));
describe sale;

OUTPUT

1.

sid	pid	quantity
S1	P1	100
S2	P1	100
S3	P3	100
S4	P4	100
S5	P5	100

2

pid	pname	price	q-avail
P1	orange	100	100
P2	apple	100	100
P3	papaya	100	100
P4	grape	100	100
P5	water	100	100

3

sid	pid	quantity
S1	P1	100
S2	P1	100
S3	P3	100
S4	P4	100
S5	P5	100
S6	P2	5

Field	Type	Null	key	Default	Extra
Sid	varchar(10)	No	PRI	NULL	
pid	varchar(10)	yes	MUL	NULL	
quantity	int	yes		NULL	

```
Insert into sale values ('S1','P1',100);
Insert into sale values ('S2','P1',100);
Insert into sale values ('S3','P3',100);
Insert into sale values ('S4','P4',100);
Insert into sale values ('S5','P5',100);

delimiter $$
Create trigger stri
after insert on sale for each row
begin
 update product set q-avail = q-avail - new.quantity
 where pid = new.pid;
 end;.
$$
```

1 → `select * from sale;`
`$$`

2 → `select * from product;`
`$$`

4

pid	pname	price	q-avail
P1	orange	100	100
P2	apple	100	95
P3	papaya	100	100
P4	grape	100	100
P5	water	100	100

→ Insert into sale values ('s6', 'p2', 5);
//

3 → Select * from sale;
//

4 → Select * from product;
//