



**B. M. S. COLLEGE OF ENGINEERING, BENGALURU**  
Autonomous Institute, Affiliated to VTU

**DEPARTMENT OF CSE**

**2021**

**Lab Report of Database Management System**

**Database Management System -19CS4PCDBM**

**Submitted by**

Name:Yashaswini Shah  
USN:1BM19CS216  
Semester : 4

<b>Experiment</b>	<b>Name of Experiment</b>
1	Insurance Database
2	Banking Enterprise Database
3	Supplier Database
4	Student Faculty Database
5	Airline Flight Database
6	Order Processing Database
7	Book dealer Database
8	Student Enrolment Database
9	Movie Database
10	College Database

## **PROGRAM 1: INSURANCE DATABASE**

Consider the Insurance database given below.

The primary keys are underlined and the data types are specified.

**PERSON (driver-id #: String, name: String, address: String)**

**CAR (Regno: String, model: String, year: int)**

**ACCIDENT (report-number: int, date: date, location: String)**

**OWNS (driver-id #: String, Regno: String)**

**PARTICIPATED (driver-id: String, Regno: String, report-number: int, damage-amount: int)**

---

```
create database insurance;
use insurance;
```

```
create table person(
    driver_id varchar(10),
    name varchar(20),
    address varchar(30),
    primary key(driver_id)
);
```

```
desc person;

create table car(
    reg_num varchar(10),
    model varchar(10),
    year int,
    primary key(reg_num)
);

desc car;

create table accident(
    report_num int,
    accident_date date,
    location varchar(20),
    primary key(report_num)
);
desc accident;

create table owns(
    driver_id varchar(10),
    reg_num varchar(10),
    primary key(driver_id,reg_num),
    foreign key(driver_id) references person(driver_id),
    foreign key(reg_num) references car(reg_num)
);

desc owns;

create table participated(
    driver_id varchar(10),
    reg_num varchar(10),
    report_num int,
    damage_amount int,
    primary key(driver_id,reg_num,report_num),
    foreign key(driver_id) references person(driver_id),
    foreign key(reg_num) references car(reg_num),
    foreign key(report_num) references accident(report_num)
);

desc participated;

insert into person values('A01','Raghu','Electronic City');
insert into person values('A02','Rishab','Orange County');
insert into person values('A03','Rufus','NR Colony');
insert into person values('A04','Jamal','Lawrence Park');
insert into person values('A05','Kevin','Rosedale');

commit;
```

```

select * from person;

insert into car values('KA031111','Accord',2005);
insert into car values('KA041122','MX-5',2019);
insert into car values('KA051133','Indica',2010);
insert into car values('KA061144','Prius',2015);
insert into car values('KA071155','Camry',2020);

commit;
select * from car;

insert into accident values(111,'2020-01-01','NR Road');
insert into accident values(122,'2020-02-02','Dalhousie Road');
insert into accident values(133,'2020-03-03','Henry Road');
insert into accident values(144,'2020-04-04','Beehive Road');
insert into accident values(155,'2020-05-05','Orange Street');
commit;

select * from accident;

insert into owns values ('A01','KA031111');
insert into owns values ('A02','KA041122');
insert into owns values ('A03','KA051133');
insert into owns values ('A04','KA061144');
insert into owns values ('A05','KA071155');
commit;

select * from owns;

insert into participated values ('A01','KA031111',111, 10000);
insert into participated values ('A02','KA041122',122, 20000);
insert into participated values ('A03','KA051133',133, 30000);
insert into participated values ('A04','KA061144',144, 40000);
insert into participated values ('A05','KA071155',155, 50000);
commit;

select * from participated;

```

- i. Create the above tables by properly specifying the primary keys and the foreign keys.**
- ii. Enter at least five tuples for each relation.**

## PERSON TABLE

MySQL Workbench

Yashaswini

File Edit View Query Database Server Tools Scripting Help

labprogram1\* labprogram2 labprogram3\* labprogram4 labprogram5\*

```
6     name varchar(20),
7     address varchar(30),
8     primary key(driver_id)
9   );
10
11
12 • desc person;
13
```

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

Field	Type	Null	Key	Default	Extra
driver_id	varchar(10)	NO	PRI	NULL	
name	varchar(20)	YES		NULL	
address	varchar(30)	YES		NULL	

Results 1-6

MySQL Workbench

Yashaswini

File Edit View Query Database Server Tools Scripting Help

labprogram1\* labprogram2 labprogram3\* labprogram4 labprogram5\*

```
56 • insert into person values('A03','Rufus','NR Colony');
57 • insert into person values('A04','Jamal','Lawrence Park');
58 • insert into person values('A05','Kevin','Rosedale');
59
60 • commit;
61
62 • select * from person;
63
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content: |

driver_id	name	address
A01	Raghu	Electronic City
A02	Rishab	Orange County
A03	Rufus	NR Colony
A04	Jamal	Lawrence Park
A05	Kevin	Rosedale
NULL	NULL	NULL

person 6

Action Output

#	Time	Action	Message
7	10:40:40	desc participated	4 row(s) returned
8	10:40:47	select * from person LIMIT 0, 10	5 row(s) returned

## CAR TABLE

MySQL Workbench - Yashaswi

File Edit View Query Database Server Tools Scripting Help

labprogram1\* labprogram2 labprogram3\* labprogram4 labprogram5\*

```
15     reg_num varchar(10),
16     model varchar(10),
17     year int,
18     primary key(reg_num)
19   );
20
21 • desc car;
22
```

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

Field	Type	Null	Key	Default	Extra
reg_num	varchar(10)	NO	PRI	NULL	
model	varchar(10)	YES		NULL	
year	int	YES		NULL	

Result 17 x Read Only

MySQL Workbench - Yashaswi

File Edit View Query Database Server Tools Scripting Help

labprogram1\* labprogram2 labprogram3\* labprogram4 labprogram5\*

```
64 • insert into car values('KA031111','Accord',2005);
65 • insert into car values('KA041122','MX-5',2019);
66 • insert into car values('KA051133','Indica',2010);
67 • insert into car values('KA061144','Prius',2015);
68 • insert into car values('KA071155','Camry',2020);
69 • commit;
70 • select * from car;
```

Result Grid | Filter Rows: [ ] | Edit: [ ] | Export/Import: [ ] | Wrap Cell Content: [ ]

reg_num	model	year
KA031111	Accord	2005
KA041122	MX-5	2019
KA051133	Indica	2010
KA061144	Prius	2015
KA071155	Camry	2020
NULL	NULL	NULL

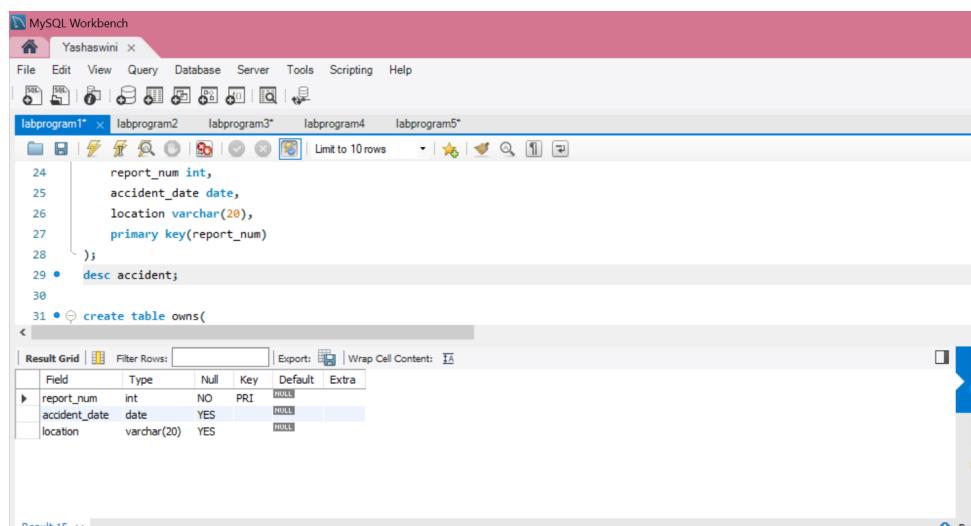
car 8 x

Output:

Action Output

#	Time	Action	Message
10	10:41:21	insert into car values('KA061144','Prius',2015)	1 row(s) affected
11	10:41:24	select * from car LIMIT 0, 10	5 row(s) returned

## ACCIDENT TABLE



The screenshot shows the MySQL Workbench interface with a pink header bar. The title bar says "MySQL Workbench" and "Yashaswini". The menu bar includes File, Edit, View, Query, Database, Server, Tools, Scripting, and Help. Below the menu is a toolbar with various icons. A tab bar at the top has five tabs: "labprogram1\*", "labprogram2", "labprogram3\*", "labprogram4", and "labprogram5\*". The main area contains a code editor with the following SQL script:

```
24     report_num int,  
25     accident_date date,  
26     location varchar(20),  
27     primary key(report_num)  
28   );  
29 •   desc accident;  
30  
31 • ↴ create table owns(  
| Result Grid | Filter Rows: [ ] | Export: [ ] | Wrap Cell Content: [ ] |
```

Below the code editor is a results grid table with the following data:

Field	Type	Null	Key	Default	Extra
report_num	int	NO	PRI	NULL	
accident_date	date	YES		NULL	
location	varchar(20)	YES		NULL	

MySQL Workbench

Yashaswini

File Edit View Query Database Server Tools Scripting Help

labprogram1\* labprogram2 labprogram3\* labprogram4 labprogram5\*

Limit to 10 rows

```

73 • insert into accident values(122,'2020-02-02','Dalhousie Road');
74 • insert into accident values(133,'2020-03-03','Henry Road');
75 • insert into accident values(144,'2020-04-04','Beehive Road');
76 • insert into accident values(155,'2020-05-05','Orange Street');
77 • commit;
78
79 • select * from accident;
80

```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Result Grid | Form Editor | Apply

report_num	accident_date	location
16	2015-03-08	Domlu
111	2020-01-01	NR Road
122	2020-02-02	Dalhousie Road
133	2020-03-03	Henry Road
144	2020-04-04	Beehive Road
155	2020-05-05	Orange Street
1008	1008	1008

accident 9 x

Action Output

#	Time	Action	Message
11	10:41:24	select * from car LIMIT 0, 10	5 row(s) returned
12	10:41:35	select * from accident LIMIT 0, 10	6 row(s) returned

## OWNS TABLE

MySQL Workbench

Yashaswini

File Edit View Query Database Server Tools Scripting Help

labprogram1\* labprogram2 labprogram3\* labprogram4 labprogram5\*

Limit to 10 rows

```

33     reg_num varchar(10),
34     primary key(driver_id,reg_num),
35     foreign key(driver_id) references person(driver_id),
36     foreign key(reg_num) references car(reg_num)
37   );
38
39 • desc owns;
40

```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Result Grid | Form Editor | Apply

Field	Type	Null	Key	Default	Extra
driver_id	varchar(10)	NO	PRI	NULL	
reg_num	varchar(10)	NO	PRI	NULL	

MySQL Workbench

Yashaswini

File Edit View Query Database Server Tools Scripting Help

labprogram1\* labprogram2 labprogram3\* labprogram4 labprogram5\*

```

82 • insert into owns values ('A02','KA041122');
83 • insert into owns values ('A03','KA051133');
84 • insert into owns values ('A04','KA061144');
85 • insert into owns values ('A05','KA071155');
86 • commit;
87
88 • select * from owns;
89 • insert into participated values ('A01','KA031111',111, 10000);

```

Result Grid | Filter Rows: | Edit | Export/Import: | Wrap Cell Content: | Result Grid | Form Editor

driver_id	reg_num
A01	KA031111
A02	KA041122
A03	KA051133
A04	KA061144
A05	KA071155
NULL	NULL

owns 11 x

Action Output

#	Time	Action	Message
14	10:42:02	insert into owns values ('A04','KA061144')	1 row(s) affected
15	10:42:06	select * from owns LIMIT 0, 10	5 row(s) returned

## PARTICIPATED TABLE

MySQL Workbench

Yashaswini

File Edit View Query Database Server Tools Scripting Help

labprogram1\* labprogram2 labprogram3\* labprogram4 labprogram5\*

```

49     foreign key(report_num) references accident(report_num)
50 );
51
52 • desc participated;
53
54 • insert into person values('A01','Raghu','Electronic City');
55 • insert into person values('A02','Rishab','Orange County');
56 • insert into person values('A03','Rufus','NR Colony');

```

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

Field	Type	Null	Key	Default	Extra
driver_id	varchar(10)	NO	PRI	NULL	
reg_num	varchar(10)	NO	PRI	NULL	
report_num	int	NO	PRI	NULL	
damage_amount	int	YES		NULL	

Result 5 x

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

```

90 • insert into participated values ('A02','KA041122',122, 20000);
91 • insert into participated values ('A03','KA051133',133, 30000);
92 • insert into participated values ('A04','KA061144',144, 40000);
93 • insert into participated values ('A05','KA071155',155, 50000);
94 • commit;
95
96 • select * from participated;
97

```

The results grid displays the following data:

driver_id	reg_num	report_num	damage_amount
A01	KA031111	111	10000
A02	KA041122	122	20000
A03	KA051133	133	30000
A04	KA061144	144	40000
A05	KA071155	155	50000
NULL	NULL	NULL	NULL

The output pane shows the following log entries:

- Action Output
 

#	Time	Action	Message
17	10:42:26	insert into participated values (A04,'KA061144',144, 40000)	1 row(s) affected
18	10:42:29	select * from participated LIMIT 0, 10	5 row(s) returned

### iii.Demonstrate how you

#### a.Update the damage amount for the car with a specific Regno in the accident with report number 122 to 25000.

```

update participated set damage_amount=25000 where reg_num='KA041122' and report_num=122 ;
commit;

select * from participated;

```

MySQL Workbench

Yashaswini

File Edit View Query Database Server Tools Scripting Help

labprogram1\* labprogram2 labprogram3\* labprogram4 labprogram5\*

```

94 •    commit;
95
96 •    select * from participated;
97
98 •    update participated set damage_amount=25000 where reg_num='KA041122' and report_num=122 ;
99 •    commit;
100 •   select * from participated;
101

```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content: | Result Grid | Form Editor | Context Help Snippets

driver_id	reg_num	report_num	damage_amount
A01	KA031111	111	10000
A02	KA041122	122	25000
A03	KA051133	133	30000
A04	KA061144	144	40000
A05	KA071155	155	50000
• HULL	HULL	HULL	HULL

participated 19 x

Action Output

#	Time	Action	Message	Duration / Fetch
30	11:01:31	commit	0 row(s) affected	0.000 sec
31	11:01:34	select * from participated LIMIT 0, 10	5 row(s) returned	0.000 sec / 0.000 sec

### b. Add a new accident to the database.

```
insert into accident values(166,'15-03-08','Domlur');
select * from accident;
```

MySQL Workbench

Yashaswini

File Edit View Query Database Server Tools Scripting Help

labprogram1\* labprogram2 labprogram3\* labprogram4 labprogram5\*

```

99 •    commit;
100 •   select * from participated;
101
102 •   insert into accident values(166,'15-03-08','Domlur');
103 •   select * from accident;
104
105
106

```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content: | Result Grid | Form Editor | Context Help Snippets

report_num	accident_date	location
16	2015-03-08	Domlur
111	2020-01-01	NR Road
122	2020-02-02	Dalhouse Road
133	2020-03-03	Henry Road
144	2020-04-04	Beehive Road
155	2020-05-05	Orange Street
166	2015-03-08	Domlur
• HULL	HULL	HULL

accident 20 x

Action Output

#	Time	Action	Message	Duration / Fetch
32	11:03:25	insert into accident values(166,'15-03-08','Domlur')	1 row(s) affected	0.000 sec
33	11:03:42	select * from accident LIMIT 0, 10	7 row(s) returned	0.000 sec / 0.000 sec

### iv. Find the total number of people who owned cars that were involved in accidents in 2008.

```
select count(distinct driver_id) CNT from participated a, accident b where a.report_num=b.report_num  
and b.accident_date like '%08';
```

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

```
101  
102 • insert into accident values(166,'15-03-08','Domlur');  
103 • select * from accident;  
104  
105  
106  
107 • select count(distinct driver_id) CNT from participated a, accident b where a.report_num=b.report_num and b.accident_date like '%08';  
108
```

The result grid shows a single row with the value '0' under the column 'CNT'. The output pane shows two log entries:

#	Time	Action	Message	Duration / Fetch
33	11:03:42	select * from accident LIMIT 0, 10	7 row(s) returned	0.000 sec / 0.000 sec
34	11:05:11	select count(distinct driver_id) CNT from participated a, accident b where a.report_num=b.report_num and... <td>1 row(s) returned</td> <td>0.015 sec / 0.000 sec</td>	1 row(s) returned	0.015 sec / 0.000 sec

#### v. Find the number of accidents in which cars belonging to a specific model were involved.

```
select count(report_num) CNT from car c,participated p where c.reg_num=p.reg_num and  
model='Accord';
```

The screenshot shows the MySQL Workbench interface. The title bar says "MySQL Workbench" and "Yashaswini". The menu bar includes File, Edit, View, Query, Database, Server, Tools, Scripting, Help. The toolbar has various icons for database management. The query editor window contains the following SQL code:

```
114
115
116
117 • select count(report_num) CNT from car c,participated p where c.reg_num=p.reg_num and model='Accord';
118
119
120
121
122
123
124
```

The result grid shows one row of data:

CNT
1

The results pane below shows "Result 1" with "Output" and "Action Output" tabs.

## PROGRAM 2: BANKING ENTERPRISE DATABASE

Consider the following database for a banking enterprise.

**BRANCH** (branch-name: String, branch-city: String, assets: real)

**ACCOUNTS** (accno: int, branch-name: String, balance: real)

**DEPOSITOR** (customer-name: String, customer-street: String, customer-city: String)

**LOAN** (loan-number: int, branch-name: String, amount: real)

**BORROWER** (customer-name: String, loan-number: int)

---

```
create database Banking;
```

```
use Banking;
```

```
create table Branch(
```

```
branchname varchar(30),
```

```
branchcity varchar(30),
```

```
asests real,
```

```
primary key (branchname)
```

```
);
```

```
desc Branch;
```

```
create table BankAccount(
```

```
accno integer,
```

```
branchname varchar(30),
```

```
balance real,
```

```
primary key (accno),
```

```
foreign key (branchname) references Branch(branchname)
```

```
);
```

```
desc BankAccount;
```

```
create table BankCustomer(
```

```
customername varchar(30),
```

```
customerstreet varchar(30),
```

```
customercity varchar(30),
```

```
primary key(customername)
```

```
);
```

```
desc BankCustomer;
```

```
create table Depositer(
```

```
customername varchar(30),
```

```
accno integer,  
primary key (customername, accno),  
foreign key (customername) references BankCustomer (customername),  
foreign key(accno) references BankAccount(accno)  
);
```

```
desc Depositer;
```

```
create table Loan (  
loannumber int,  
branchname varchar(30),  
amount real,  
primary key (loannumber),  
foreign key (branchname) references Branch(branchname)  
);
```

```
desc Loan;
```

```
insert into Branch values ('SBI_Chamrajpet','Bangalore',50000);  
insert into Branch values ('SBI_ResidencyRoad','Bangalore',10000);  
insert into Branch values ('SBI_ShivajiRoad','Bombay',20000);  
insert into Branch values ('SBI_ParliamentRoad','Delhi',10000);  
insert into Branch values ('SBI_Jantarmantar','Delhi',20000);
```

```
select * from Branch;
```

```
insert into Loan values(2,'SBI_ResidencyRoad',2000);
```

```
insert into Loan values(1,'SBI_Chamrajpet',1000);
insert into Loan values(3,'SBI_ShivajiRoad', 3000);
insert into Loan values(4,'SBI_ParliamentRoad',4000);
insert into Loan values(5,'SBI_Jantarmantar',5000);
```

```
select * from Loan;
```

```
insert into BankAccount values(11,'SBI_Jantarmantar',2000);
insert into BankAccount values(10,'SBI_ResidencyRoad',5000);
insert into BankAccount values(9,'SBI_ParliamentRoad',3000);
insert into BankAccount values(8,'SBI_ResidencyRoad',4000);
insert into BankAccount values(6,'SBI_ShivajiRoad',4000);
insert into BankAccount values(5,'SBI_Jantarmantar',8000);
insert into BankAccount values(4,'SBI_ParliamentRoad',9000);
insert into BankAccount values(3,'SBI_ShivajiRoad',6000);
insert into BankAccount values(2,'SBI_ResidencyRoad',5000);
insert into BankAccount values(1,'SBI_Chamrajpet',2000);
```

```
commit;
```

```
select * from BankAccount;
```

```
insert into Depositer values('Avinash',1);
insert into Depositer values('Dinesh',2);
insert into Depositer values('Nikil',4);
insert into Depositer values('Ravi',5);
insert into Depositer values('Avinash',8);
insert into Depositer values('Nikil',9);
insert into Depositer values('Dinesh',10);
```

```
insert into Depositer values('Nikil',11);
```

```
select * from Depositer;
```

```
insert into BankCustomer values('Avinash','Bull_Temple_Road','Bangalore');
```

```
insert into BankCustomer values('Dinesh','BannerGatta_Road','Bangalore');
```

```
insert into BankCustomer values('Mohan','NationalCollege_Road','Bangalore');
```

```
insert into BankCustomer values('Nikil','Akbar_Road','Delhi');
```

```
insert into BankCustomer values('Ravi','Prithiviraj_Road','Delhi');
```

```
select * from BankCustomer;
```

**i. Create the above tables by properly specifying the primary keys and the foreign keys.**

**ii. Enter at least five tuples for each relation.**

### **TABLE BRANCH**

MySQL Workbench

Yashaswini

File Edit View Query Database Server Tools Scripting Help

labprogram1\* labprogram2\* labprogram3\* labprogram4 labprogram5\*

```

4 • create table Branch(
5     branchname varchar(30),
6     branchcity varchar(30),
7     assets real,
8     primary key (branchname)
9 );
10 • desc Branch;
11

```

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

Field	Type	Null	Key	Default	Extra
branchname	varchar(30)	NO	PRI	HULL	
branchcity	varchar(30)	YES		HULL	
assets	double	YES		HULL	

MySQL Workbench

Yashaswini

File Edit View Query Database Server Tools Scripting Help

labprogram1\* labprogram2\* labprogram3\* labprogram4 labprogram5\*

```

50
51 • insert into Branch values ('SBI_Chamrajpet','Bangalore',50000);
52 • insert into Branch values ('SBI_ResidencyRoad','Bangalore',10000);
53 • insert into Branch values ('SBI_ShivajiRoad','Bombay',20000);
54 • insert into Branch values ('SBI_ParliamentRoad','Delhi',10000);
55 • insert into Branch values ('SBI_Jantarmantar','Delhi',20000);
56
57 • select * from Branch;

```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content: |

branchname	branchcity	assets
SBI_Chamrajpet	Bangalore	50000
SBI_Jantarmantar	Delhi	20000
SBI_ParliamentRoad	Delhi	10000
SBI_ResidencyRoad	Bangalore	10000
SBI_ShivajiRoad	Bombay	20000
*	HULL	HULL

Branch 6 x

Output:

Action Output

#	Time	Action	Message	Duration / Fetch
46	11:22:32	desc Loan	3 row(s) returned	0.000 sec / 0.000 sec
47	11:23:17	select * from Branch LIMIT 0, 10	5 row(s) returned	0.031 sec / 0.000 sec

Type here to search

## TABLE BANK ACCOUNT

MySQL Workbench

Yashaswini

File Edit View Query Database Server Tools Scripting Help

labprogram1\* labprogram2\* labprogram3\* labprogram4 labprogram5\*

12 • create table BankAccount( accno integer, branchname varchar(30), balance real, primary key (accno), foreign key (branchname) references Branch(branchname) );

13  
14  
15  
16  
17  
18  
19  
20 • desc BankAccount;

21

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

Field	Type	Null	Key	Default	Extra
accno	int	NO	PRI	HULL	
branchname	varchar(30)	YES	MUL	HULL	
balance	double	YES		HULL	

MySQL Workbench

Yashaswini

File Edit View Query Database Server Tools Scripting Help

labprogram1\* labprogram2\* labprogram3\* labprogram4 labprogram5\*

68 • insert into BankAccount values(11,'SBI\_Jantarmantar',2000);  
69 • insert into BankAccount values(10,'SBI\_ResidencyRoad',5000);  
70 • insert into BankAccount values(9,'SBI\_ParliamentRoad',3000);  
71 • insert into BankAccount values(8,'SBI\_ResidencyRoad',4000);  
72 • insert into BankAccount values(6,'SBI\_ShivajiRoad',4000);  
73 • insert into BankAccount values(5,'SBI\_Jantarmantar',8000);  
74 • insert into BankAccount values(4,'SBI\_ParliamentRoad',9000);  
75 • insert into BankAccount values(3,'SBI\_ShivajiRoad',6000);  
76 • insert into BankAccount values(2,'SBI\_ResidencyRoad',5000);  
77 • insert into BankAccount values(1,'SBI\_Chamrajpet',2000);

Result Grid | Filter Rows: [ ] Edit: [ ] Export/Import: [ ] Wrap Cell Content: [ ] Fetch rows: [ ]

accno	branchname	balance
1	SBI_Chamrajpet	2000
2	SBI_ResidencyRoad	5000
3	SBI_ShivajiRoad	6000
4	SBI_ParliamentRoad	9000
5	SBI_Jantarmantar	8000
6	SBI_ShivajiRoad	4000
8	SBI_ResidencyRoad	4000
9	SBI_ParliamentRoad	3000
10	SBI_ResidencyRoad	5000
11	SBI_Jantarmantar	2000
*	HULL	HULL

BankAccount 8

Context Help Snippets

Output

Type here to search

11:25 AM 5/23/2021 6

## TABLE BANK CUSTOMER

MySQL Workbench

Yashaswini

File Edit View Query Database Server Tools Scripting Help

labprogram1\* labprogram2\* labprogram3\* labprogram4 labprogram5\*

```

21
22 • create table BankCustomer(
23     customername varchar(30),
24     customerstreet varchar(30),
25     customercity varchar(30),
26     primary key(customername)
27 );
28
29 • desc BankCustomer;
30

```

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

Field	Type	Null	Key	Default	Extra
customername	varchar(30)	NO	PRI	NULL	
customerstreet	varchar(30)	YES		NULL	
customercity	varchar(30)	YES		NULL	

MySQL Workbench

Yashaswini

File Edit View Query Database Server Tools Scripting Help

labprogram1\* labprogram2\* labprogram3\* labprogram4 labprogram5\*

```

92
93
94 • insert into BankCustomer values('Avinash','Bull_Temple_Road','Bangalore');
95 • insert into BankCustomer values('Dinesh','Bannergatta_Road','Bangalore');
96 • insert into BankCustomer values('Mohan','NationalCollege_Road','Bangalore');
97 • insert into BankCustomer values('Nikil','Akbar_Road','Delhi');
98 • insert into BankCustomer values('Ravi','Prithviraj_Road','Delhi');
99
100 • select *from BankCustomer;
101

```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content: |

customername	customerstreet	customercity
Avinash	Bull_Temple_Road	Bangalore
Dinesh	Bannergatta_Road	Bangalore
Mohan	NationalCollege_Road	Bangalore
Nikil	Akbar_Road	Delhi
Ravi	Prithviraj_Road	Delhi
NULL	NULL	NULL

SQLAdditions... | Jump to

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

BankCustomer 10 x

Output : Type here to search

11:26 AM 5/23/2021 6

## TABLE DEPOSITOR

MySQL Workbench

Yashaswini

File Edit View Query Database Server Tools Scripting Help

labprogram1\* labprogram2\* labprogram3\* labprogram4 labprogram5\*

31 • create table Depositer(  
32     customername varchar(30),  
33     accno integer,  
34     primary key (customername, accno),  
35     foreign key (customername) references BankCustomer (customername),  
36     foreign key(accno) references BankAccount(accno)  
37 );  
38  
39 • desc Depositer;  
40

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

Field	Type	Null	Key	Default	Extra
customername	varchar(30)	NO	PRI	HULL	
accno	int	NO	PRI	HULL	

MySQL Workbench

Yashaswini

File Edit View Query Database Server Tools Scripting Help

labprogram1\* labprogram2\* labprogram3\* labprogram4 labprogram5\*

82 • insert into Depositer values('Avinash',1);  
83 • insert into Depositer values('Dinesh',2);  
84 • insert into Depositer values('Nikil',4);  
85 • insert into Depositer values('Ravi',5);  
86 • insert into Depositer values('Avinash',8);  
87 • insert into Depositer values('Nikil',9);  
88 • insert into Depositer values('Dinesh',10);  
89 • insert into Depositer values('Nikil',11);  
90  
91 • select \* from Depositer;

Result Grid | Filter Rows: [ ] Edit: [ ] Export/Import: [ ] Wrap Cell Content: [ ]

customername	accno
Avinash	1
Dinesh	2
Nikil	4
Ravi	5
Avinash	8
Nikil	9
Dinesh	10
Nikil	11
*	HULL
*	HULL

Depositer 9 x Apply Revert Context Help Snippets

Output :

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

## TABLE LOAN

MySQL Workbench

Yashaswini

File Edit View Query Database Server Tools Scripting Help

labprogram1\* labprogram2\* labprogram3\* labprogram4 labprogram5\*

```
41 • Ⓜ create table Loan (
42     loannumber int,
43     branchname varchar(30),
44     amount real,
45     primary key (loannumber),
46     foreign key (branchname) references Branch(branchname)
47 );
48
49 • desc Loans;
```

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

Field	Type	Null	Key	Default	Extra
loannumber	int	NO	PRI	NULL	
branchname	varchar(30)	YES	MUL	NULL	
amount	double	YES		NULL	

MySQL Workbench

Yashaswini

File Edit View Query Database Server Tools Scripting Help

labprogram1\* labprogram2\* labprogram3\* labprogram4 labprogram5\*

```
60 • insert into Loan values(2,'SBI_ResidencyRoad',2000);
61 • insert into Loan values(1,'SBI_Chamrajpet',1000);
62 • insert into Loan values(3,'SBI_ShivajiRoad', 3000);
63 • insert into Loan values(4,'SBI_ParlimentRoad',4000);
64 • insert into Loan values(5,'SBI_Jantarmantar',5000);
65
66 • select * from Loan;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content: |

loannumber	branchname	amount
1	SBI_Chamrajpet	1000
2	SBI_ResidencyRoad	2000
3	SBI_ShivajiRoad	3000
4	SBI_ParlimentRoad	4000
5	SBI_Jantarmantar	5000
*	NULL	NULL

Loan 7 x

Output:

Action Output

#	Time	Action	Message	Duration / Fetch
47	11:23:17	select * from Branch LIMIT 0, 10	5 row(s) returned	0.031 sec / 0.000 sec
48	11:24:43	select * from Loan LIMIT 0, 10	5 row(s) returned	0.000 sec / 0.000 sec

SQLAdditions... | Jump to | Context Help | Snippets

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

### iii. Find all the customers who have at least two accounts at the Main branch.

Select C.customername from BankCustomer C

where exists(

select D.customername, count(D.customername)

from depositer D, BankAccount BA

where

D.accno = BA.accno AND

C.customername = D.customername AND

BA.branchname= 'SBI\_ResidencyRoad'

group by D.customername

having count(D.customername)>=2);

```
102 • Select C.customername from BankCustomer C
103   where exists(
104     select D.customername, count(D.customername)
105       from depositer D, BankAccount BA
106      where
107        D.accno = BA.accno AND
108        C.customername = D.customername AND
109        BA.branchname= 'SBI_ResidencyRoad'
110      group by D.customername
111      having count(D.customername)>=2);
112
```

customername
Dinesh
NULL

BankCustomer 1 ×

Output :

#	Time	Action	Message	Duration / Fetch
2	09:38:37	use Banking	0 row(s) affected	0.016 sec
3	09:38:51	Select C.customername from BankCustomer C where exists( select D.customername, count(D.customername) from depositer D, BankAccount BA where D.accno = BA.accno AND C.customername = D.customername AND BA.branchname= 'SBI_ResidencyRoad' group by D.customername having count(D.customername)>=2);	1 row(s) returned	0.078 sec / 0.000 sec

Type here to search

### iv. Find all the customers who have an account at all the branches located in a specific city. v.

select customer\_name from depositor d,

accounts a where d.accno=a.accno and branch\_name in

(select branch\_name as branch\_name from branch where branch\_city='Delhi')

group by customer\_name having count(\*)=(select count(branch\_name)

as branch\_name from branch where branch\_city='Delhi');

The screenshot shows the MySQL Workbench interface. In the top-left, there's a tab bar with 'labprogram1\*', 'labprogram2\*', 'labprogram3', 'labprogram4.', and 'SQL File 7\*'. The 'SQL File 7\*' tab is active, displaying the following SQL code:

```

72
73 • select customer_name from depositor d,
74 accounts a where d.accno=a.accno and branch_name in
75 (select branch_name as branch_name from branch where branch_city='Delhi')
76 group by customer_name having count(*)=(select count(branch_name)
77 as branch_name from branch where branch_city='Delhi');
78
79

```

Below the code is a 'Result Grid' showing a single row:

customer_name
customer1

To the right of the results is a sidebar titled 'SQLAdditions' with a note: "Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help." Below the sidebar are buttons for 'Result Grid' and 'Form Editor'.

At the bottom, the 'Output' pane shows two log entries:

#	Time	Action	Message	Duration / Fetch
98	21:14:02	delete from accounts where branch_name=banglore'	0 row(s) affected	0.000 sec
99	21:16:09	select customer name from depositor d, accounts a where d accno=a.accno and branch name in (selec...	1 row(s) returned	0.000 sec / 0.000 sec

**v.Demonstrate how you delete all account tuples at every branch located in a specific city.**

**(BOMBAY)**

```

delete from BankAccount
where branchname IN (
select branchname
from Branch
where branchcity='Bombay'
);
select * from BankAccount;

```

The screenshot shows the MySQL Workbench interface. In the top-left corner, it says "MySQL Workbench" and "Yashaswini". The menu bar includes File, Edit, View, Query, Database, Server, Tools, Scripting, and Help. Below the menu is a toolbar with various icons. The main area contains a query editor with the following SQL code:

```

129
130
131
132 • delete from BankAccount
133   where branchname IN (
134     select branchname
135     from Branch
136     where branchcity='Bombay'
137   );
138 • select * from BankAccounts;

```

Below the code is a "Result Grid" showing the results of the query. The columns are "acco", "branchname", and "balance". The data is as follows:

acco	branchname	balance
1	SBI_Chamajpet	2000
2	SBI_ResidencyRoad	5000
4	SBI_ParliamentRoad	9000
5	SBI_Jantarmantar	8000
8	SBI_ResidencyRoad	4000
9	SBI_ParliamentRoad	3000
10	SBI_ResidencyRoad	5000
11	SBI_Jantarmantar	2000
*	HOLE	HOLE

On the right side of the interface, there is a "SQL Additions" panel with a message: "Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help." Below this is a "Result Grid" button.

## PROGRAM 3: SUPPLIER DATABASE

Consider the following schema:

**SUPPLIERS** (sid: integer, sname: string, address: string)

**PARTS** (pid: integer, pname: string, color: string)

**CATALOG** (sid: integer, pid: integer, cost: real)

The Catalog relation lists the prices charged for parts by Suppliers.

---

create database Supplier;

use Supplier;

create table SUPPLIERS (

sid integer

```
primary key,  
sname varchar(20),  
city varchar(20)  
);
```

```
desc SUPPLIERS;
```

```
create table PARTS(  
pid integer primary key,  
pname varchar(20),  
color varchar(10)  
);
```

```
desc PARTS;
```

```
create table CATALOG (  
sid integer,  
pid integer,  
foreign key(sid) references SUPPLIERS(sid),  
foreign key(pid) references PARTS(pid),  
cost float(6),  
primary key(sid,pid)  
);
```

```
desc CATALOG;
```

```
insert into suppliers value (10001,'Acme Widget','Bangalore');  
insert into suppliers value (10002,'Johns','Kolkata');
```

```
insert into suppliers value (10003,'Vimal','Mumbai');  
insert into suppliers value (10004,'Reliance','Delhi');  
insert into suppliers value(10005,'Mahindra','Mumbai');
```

```
select * from SUPPLIERS;
```

```
commit;
```

```
insert into PARTS values(20001,'Book','Red');  
insert into PARTS values(20002,'Pen','Red');  
insert into PARTS values(20003,'Pencil','Green');  
insert into PARTS values(20004,'Mobile','Green');  
insert into PARTS values(20005,'Charger','Black');
```

```
select * from PARTS;
```

```
commit;
```

```
insert into CATALOG values(10001,'20001','10');  
insert into CATALOG values(10001,'20002','10');  
insert into CATALOG values(10001,'20003','30');  
insert into CATALOG values(10001,'20004','10');  
insert into CATALOG values(10001,'20005','10');
```

```
insert into CATALOG values(10002,'20001','10');  
insert into CATALOG values(10002,'20002','20');
```

```
insert into CATALOG values(10003,'20003','30');
```

```
insert into CATALOG values(10004,'20003','40');
```

```
select * from CATALOG;
```

```
commit;
```

## 1) SUPPLIERS

The screenshot shows the MySQL Workbench interface with the following details:

- Title Bar:** MySQL Workbench - Yashaswini
- Menu Bar:** File, Edit, View, Query, Database, Server, Tools, Scripting, Help
- Toolbar:** Includes icons for Home, New, Open, Save, Print, Copy, Paste, Find, Replace, and others.
- Query Editor:** Shows two queries:
  - Line 4: `create table SUPPLIERS (`
  - Line 5: `sid integer`
  - Line 6: `primary key,`
  - Line 7: `sname varchar(20),`
  - Line 8: `city varchar(20)`
  - Line 9: `);`
  - Line 10: An empty line.
  - Line 11: `desc SUPPLIERS;`
- Result Grid:** Displays the structure of the SUPPLIERS table.

Field	Type	Null	Key	Default	Extra
sid	int	NO	PRI	NULL	
sname	varchar(20)	YES		NULL	
city	varchar(20)	YES		NULL	
- Status Bar:** Result 1

MySQL Workbench

Yashaswini

File Edit View Query Database Server Tools Scripting Help

labprogram1 labprogram2 labprogram3\* labprogram4 labprogram5\*

```

33 • insert into suppliers value (10001,'Acme Widget','Bangalore');
34 • insert into suppliers value (10002,'Johns','Kolkata');
35 • insert into suppliers value (10003,'Vimal','Mumbai');
36 • insert into suppliers value (10004,'Reliance','Delhi');
37 • insert into suppliers value(10005,'Mahindra','Mumbai');
38
39 • select * from SUPPLIERS;

```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content: | SQLAdditions: | Jump to |

sid	sname	city
10001	Acme Widget	Bangalore
10002	Johns	Kolkata
10003	Vimal	Mumbai
10004	Reliance	Delhi
10005	Mahindra	Mumbai
NULL	NULL	NULL

SUPPLIERS 2 SUPPLIERS 3 x

Action Output

#	Time	Action	Message	Duration / Fetch
1	21:52:51	use Supplier	0 row(s) affected	0.000 sec
2	21:52:55	desc SUPPLIERS	3 row(s) returned	0.000 sec / 0.000 sec
3	21:54:21	select * from SUPPLIERS LIMIT 0, 10	5 row(s) returned	0.031 sec / 0.000 sec
4	21:54:21	select * from SUPPLIERS LIMIT 0, 10	5 row(s) returned	0.000 sec / 0.000 sec

Output

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

## 2) PARTS TABLE

MySQL Workbench

Yashaswini

File Edit View Query Database Server Tools Scripting Help

labprogram1 labprogram2 labprogram3\* labprogram4 labprogram5\*

```

13 • create table PARTS(
14     pid integer primary key,
15     pname varchar(20),
16     color varchar(10)
17 );
18
19 • desc PARTS;

```

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

Field	Type	Null	Key	Default	Extra
pid	int	NO	PRI	NULL	
pname	varchar(20)	YES		NULL	
color	varchar(10)	YES		NULL	

MySQL Workbench

Yashaswini

File Edit View Query Database Server Tools Scripting Help

labprogram1 labprogram2 labprogram3\* labprogram4 labprogram5\*

43 • insert into PARTS values(20001,'Book','Red');  
44 • insert into PARTS values(20002,'Pen','Red');  
45 • insert into PARTS values(20003,'Pencil','Green');  
46 • insert into PARTS values(20004,'Mobile','Green');  
47 • insert into PARTS values(20005,'Charger','Black');  
48  
49 • select \* from PARTS;

Result Grid | Filter Rows: [ ] | Edit: [ ] | Export/Import: [ ] | Wrap Cell Content: [ ]

pid	pname	color
20001	Book	Red
20002	Pen	Red
20003	Pencil	Green
20004	Mobile	Green
20005	Charger	Black
*	NULL	NULL

PARTS 6 x

Output

Action Output

#	Time	Action	Message	Duration / Fetch
5	21:55:42	desc PARTS	3 row(s) returned	0.000 sec / 0.000 sec
6	21:56:19	select * from PARTS LIMIT 0, 10	5 row(s) returned	0.031 sec / 0.000 sec
7	21:56:19	select * from PARTS LIMIT 0, 10	5 row(s) returned	0.000 sec / 0.000 sec

SQLAdditions: [ ] | Jump to [ ]

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

### 3) CATALOG TABLE

MySQL Workbench

Yashaswini

File Edit View Query Database Server Tools Scripting Help

labprogram1 labprogram2 labprogram3\* labprogram4 labprogram5\*

```

21 • create table CATALOG (
22     sid integer,
23     pid integer,
24     foreign key(sid) references SUPPLIERS(sid),
25     foreign key(pid) references PARTS(pid),
26     cost float(6),
27     primary key(sid,pid)
28 );
29

```

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

Field	Type	Null	Key	Default	Extra
sid	int	NO	PRI	HULL	
pid	int	NO	PRI	HULL	
cost	float	YES		HULL	

MySQL Workbench

Yashaswini

File Edit View Query Database Server Tools Scripting Help

labprogram1 labprogram2 labprogram3\* labprogram4 labprogram5\*

```

63
64 • insert into CATALOG values('10004','20003','40');
65
66 • select * from CATALOG;
67

```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content: |

sid	pid	cost
10001	20001	10
10001	20002	10
10001	20003	30
10001	20004	10
10001	20005	10
10002	20001	10
10002	20002	20
10003	20003	30
10004	20003	40
*	HULL	HULL

CATALOG 8 CATALOG 9 x

Output

Action Output

#	Time	Action	Message	Duration / Fetch
8	21:58:44	desc CATALOG	3 row(s) returned	0.000 sec / 0.000 sec
9	22:00:10	select * from CATALOG LIMIT 0, 10	9 row(s) returned	0.016 sec / 0.000 sec
10	22:00:10	select * from CATALOG LIMIT 0, 10	9 row(s) returned	0.000 sec / 0.000 sec

SQLAdditions... | Jump to | Context Help Snippets

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

**Write the following queries in SQL:**

i. Find the pnames of parts for which there is some supplier.

insert into parts values(5,'tiles','blue');

```

select p.pname from parts p where p.pid in
(select pid from catalog c group by c.pid having count(c.sid)>0);
insert into catalog values(1,5,140);
select p.pname from parts p where p.pid in
(select pid from catalog c group by c.pid having count(c.sid)>0);
delete from catalog where pid=5;
delete from parts where pid=5;

```

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

```

70      -- i. Find the pnames of parts for which there is some supplier.
71
72 •   insert into parts values(5,'tiles','blue');
73 •   select p.pname from parts p where p.pid in
74     (select pid from catalog c group by c.pid having count(c.sid)>0);
75 •   insert into catalog values(1,5,140);
76 •   select p.pname from parts p where p.pid in
77     (select pid from catalog c group by c.pid having count(c.sid)>0);
78 •   delete from catalog where pid=5;
79 •   delete from parts where pid=5;

```

The results grid shows the following data:

pname
Book
Pen
Pencil
Mobile
Charger

The output pane shows the following log entries:

- Action Output
 

#	Time	Action
22	22:40:55	insert into parts values(5,tiles,'blue')
23	22:40:58	select p.pname from parts p where p.pid in (select pid from catalog c group by c.pid having count(c.sid)>0)

 Message: 1 row(s) affected  
 5 row(s) returned

## ii. Find the snames of suppliers who supply every part.

```

select s.sname from suppliers s where s.sid in
(select c.sid from catalog c
group by c.sid having count(distinct (c.pid))=(select count(p.pid) from parts p));

```

MySQL Workbench

Yashaswini

File Edit View Query Database Server Tools Scripting Help

labprogram1 labprogram2 labprogram3\* labprogram4 labprogram5\*

```

79 • delete from parts where pid=5;
80
81
82 -- ii. Find the snames of suppliers who supply every part.
83
84 • select s.sname from suppliers s where s.sid in
85   (select c.sid from catalog c
86   group by c.sid having count(distinct (c.pid))=(select count(p.pid) from parts p));
87
88

```

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

Result Grid

Form Editor

Read Only Context Help Snippets

Action Output

#	Time	Action	Message	Duration / Fetch
25	22:41:51	select s.sname from suppliers s where s.sid in (select c.sid from catalog c group by c.sid having count(dis...)	0 row(s) returned	0.016 sec / 0.000 sec
26	22:41:54	select s.sname from suppliers s where s.sid in (select c.sid from catalog c group by c.sid having count(dis...)	0 row(s) returned	0.000 sec / 0.000 sec

suppliers 18 x

Output

Action Output

#	Time	Action	Message	Duration / Fetch
25	22:41:51	select s.sname from suppliers s where s.sid in (select c.sid from catalog c group by c.sid having count(dis...)	0 row(s) returned	0.016 sec / 0.000 sec
26	22:41:54	select s.sname from suppliers s where s.sid in (select c.sid from catalog c group by c.sid having count(dis...)	0 row(s) returned	0.000 sec / 0.000 sec

### iii. Find the snames of suppliers who supply every red part.

select s.sname from suppliers s where s.sid in

(select ca.sid from catalog ca,

parts p where ca.pid=p.pid and p.color='red'

group by ca.sid having count(ca.pid)=(select count(\*) from parts p where p.color='red'));

MySQL Workbench

Yashaswini

File Edit View Query Database Server Tools Scripting Help

labprogram1 labprogram2 labprogram3\* labprogram4 labprogram5\*

```

88
89 -- iii. Find the snames of suppliers who supply every red part.
90
91 • select s.sname from suppliers s where s.sid in
92   (select ca.sid from catalog ca,
93    parts p where ca.pid=p.pid and p.color='red'
94   group by ca.sid having count(ca.pid)=(select count(*) from parts p where p.color='red')));
95
96
97

```

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

Result Grid

Form Editor

Read Only Context Help Snippets

Action Output

#	Time	Action	Message	Duration / Fetch
28	22:42:23	select s.sname from suppliers s where s.sid in (select ca.sid from catalog ca, parts p where ca.pid=p.pid ...	2 row(s) returned	0.000 sec / 0.000 sec
29	22:42:37	select s.sname from suppliers s where s.sid in (select ca.sid from catalog ca, parts p where ca.pid=p.pid ...	2 row(s) returned	0.000 sec / 0.000 sec

suppliers 21 x

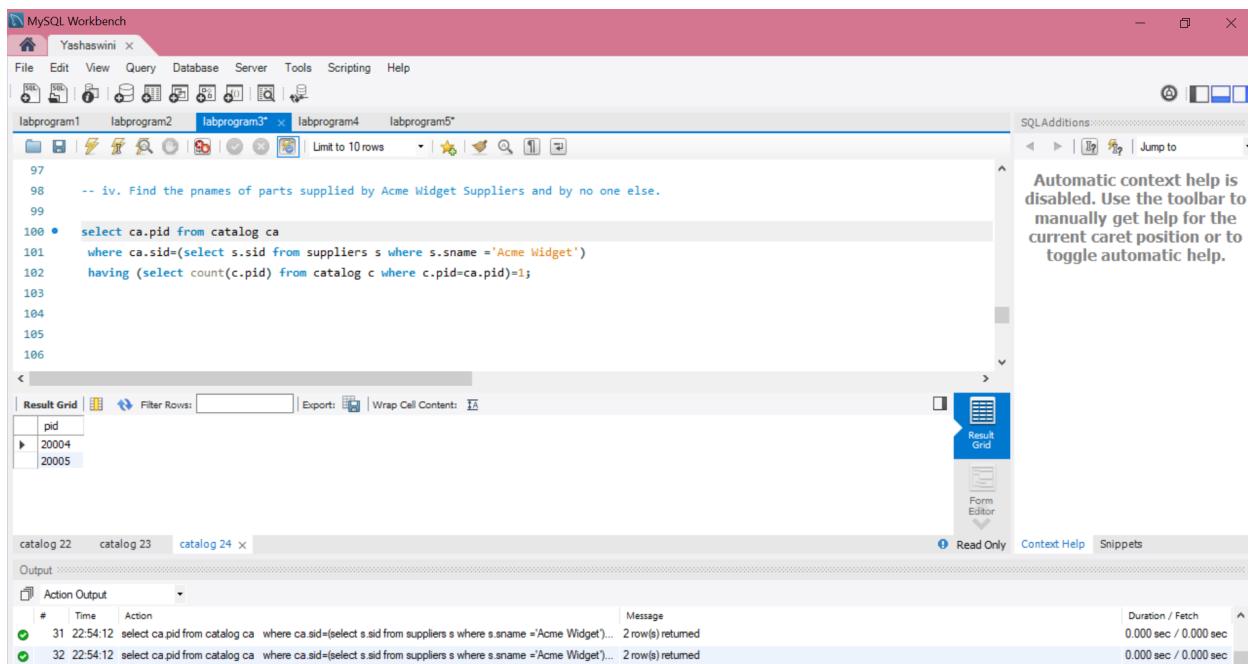
Output

Action Output

#	Time	Action	Message	Duration / Fetch
28	22:42:23	select s.sname from suppliers s where s.sid in (select ca.sid from catalog ca, parts p where ca.pid=p.pid ...	2 row(s) returned	0.000 sec / 0.000 sec
29	22:42:37	select s.sname from suppliers s where s.sid in (select ca.sid from catalog ca, parts p where ca.pid=p.pid ...	2 row(s) returned	0.000 sec / 0.000 sec

**iv. Find the pnames of parts supplied by Acme Widget Suppliers and by no one else.**

```
select ca.pid from catalog ca
where ca.sid=(select s.sid from suppliers s where s.sname ='Acme Widget')
having (select count(c.pid) from catalog c where c.pid=ca.pid)=1;
```



The screenshot shows the MySQL Workbench interface with the following details:

- Toolbar:** Standard MySQL Workbench toolbar with icons for file operations, database navigation, and search.
- Query Editor:** Shows the SQL query:

```
97
98 -- iv. Find the pnames of parts supplied by Acme Widget Suppliers and by no one else.
99
100 • select ca.pid from catalog ca
101   where ca.sid=(select s.sid from suppliers s where s.sname ='Acme Widget')
102   having (select count(c.pid) from catalog c where c.pid=ca.pid)=1;
103
104
105
106
```
- Result Grid:** Displays the results of the query:

pid
20004
20005
- Output Window:** Shows the log of actions taken:

#	Time	Action	Message	Duration / Fetch
31	22:54:12	select ca.pid from catalog ca where ca.sid=(select s.sid from suppliers s where s.sname ='Acme Widget')... 2 row(s) returned		0.000 sec / 0.000 sec
32	22:54:12	select ca.pid from catalog ca where ca.sid=(select s.sid from suppliers s where s.sname ='Acme Widget')... 2 row(s) returned		0.000 sec / 0.000 sec
- Help Panel:** A tooltip message: "Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help."

**v. Find the sids of suppliers who charge more for some part than the average cost of that part (averaged over all the suppliers who supply that part).**

```
select distinct c.sid, c.pid from catalog c where c.cost > (select avg(ca.cost) from catalog ca where ca.pid=c.pid);
```

MySQL Workbench

Yashaswini

File Edit View Query Database Server Tools Scripting Help

labprogram1 labprogram2 labprogram3\* labprogram4 labprogram5\*

```

106
107    -- v. Find the sids of suppliers who charge more for some part than the average cost of that part (averaged over
108    -- all the suppliers who supply that part).
109
110 • select distinct c.sid,
111   c.pid from catalog c where c.cost > (select avg(ca.cost)
112   from catalog ca where ca.pid=c.pid);
113
114
115

```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content: | Result Grid | Form Editor | Context Help | Snippets

sid	pid
10002	20002
10004	20003
10005	20003

catalog 25 x

Output

Action Output

#	Time	Action	Message	Duration / Fetch
32	22:54:12	select ca.pid from catalog ca where ca.sid=(select s.sid from suppliers s where s.sname ='Acme Widget')...	2 row(s) returned	0.000 sec / 0.000 sec
33	22:54:36	select distinct c.sid, c.pid from catalog c where c.cost > (select avg(ca.cost) from catalog ca where ca.pi...	2 row(s) returned	0.000 sec / 0.000 sec

SQLAdditions: | Jump to |

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

## vi. For each part, find the sname of the supplier who charges the most for that part.

select s.sname from suppliers s where s.sid in

(select c.sid from catalog c where c.cost=(select max(cost) from catalog ca where ca.pid=c.pid));

MySQL Workbench

Yashaswini

File Edit View Query Database Server Tools Scripting Help

labprogram1 labprogram2 labprogram3\* labprogram4 labprogram5\*

```

115
116    -- vi. For each part, find the sname of the supplier who charges the most for that part.
117
118 • select s.sname from suppliers s where s.sid in
119   (select c.sid from catalog c where c.cost=(select max(cost) from catalog ca where ca.pid=c.pid));
120
121
122
123
124

```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Result Grid | Form Editor | Context Help | Snippets

sname
Acme Widget
Johns
Reliance

suppliers 26 x

Output

Action Output

#	Time	Action	Message	Duration / Fetch
33	22:54:36	select distinct c.sid, c.pid from catalog c where c.cost > (select avg(ca.cost) from catalog ca where ca.pi...	2 row(s) returned	0.000 sec / 0.000 sec
34	22:55:06	select s.sname from suppliers s where s.sid in (select c.sid from catalog c where c.cost=(select max(cost) ...	3 row(s) returned	0.000 sec / 0.000 sec

SQLAdditions: | Jump to |

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

**vii. Find the sids of suppliers who supply only red parts.**

```
select s.sname from suppliers s where
s.sid in(select c.sid from catalog c where c.sid not in (select distinct(ca.sid) from catalog ca,parts p
where ca.pid=p.pid and p.color!='red'));
insert into catalog values(5,1,140);
select s.sname from suppliers s where s.sid in(select c.sid from catalog c where c.sid not in (select distinct(ca.sid) from catalog ca,parts p where ca.pid=p.pid and p.color!='red'));
delete from catalog where sid=5;
```

The screenshot shows the MySQL Workbench interface with the following details:

- Title Bar:** MySQL Workbench - Yashaswini
- Toolbar:** Standard MySQL Workbench toolbar with icons for file operations, database management, and query execution.
- Query Editor:** Contains the SQL code provided above. Lines 127 through 130 are highlighted in blue, indicating they were executed.
- Result Grid:** Shows the output of the query `select sname from suppliers s where s.sid in(select c.sid from catalog c where c.sid not in (select distinct(ca.sid) from catalog ca,parts p where ca.pid=p.pid and p.color!='red'))`.
- Action Output:** Displays log entries for the executed statements:
  - Statement 39: `insert into catalog values(5,1,140)` - Error: Error Code: 1452. Cannot add or update a child row: a foreign key constraint fails ('supplier','catalog', CO...`
  - Statement 40: `select sname from suppliers s where s.sid in(select c.sid from catalog c where c.sid not in (select distinct(...` - Result: 1 row(s) returned

Name: Yashaswini Shah

USN:1BM19CS216

## **PROGRAM 4:STUDENT FACULTY DATABASE**

**Consider the following database for student enrolment for course:**

**STUDENT (snum: integer, sname: string, major: string, level: string, age: integer)**

**CLASS (name: string, meets at: time, room: string, fid: integer)**

**ENROLLED (snum: integer, cname: string)**

**FACULTY (fid: integer, fname: string, deptid: integer)**

The meaning of these relations is straightforward; for example, Enrolled has one record per student-class pair

such that the student is enrolled in the class. Level is a two character code with 4

different values (example: Junior: JR etc)

Write the following queries in SQL.

No duplicates should be printed in any of the answers.

---

```
create database student_faculty;
```

```
use student_faculty;
```

```
create table student(
```

```
    snum INT,
```

```
    sname VARCHAR(10),
```

```
    major VARCHAR(2),
```

```
    lvl VARCHAR(2),
```

```
    age INT,
```

```
    primary key(snum));
```

```
desc student;
```

```
create table faculty(
```

```
    fid INT,
```

```
    fname VARCHAR(20),
```

```
    deptid INT,
```

```
    PRIMARY KEY(fid));
```

```
desc faculty;
```

```
CREATE TABLE class(
    cname VARCHAR(20),
    metts_at TIMESTAMP,
    room VARCHAR(10),
    fid INT,
    PRIMARY KEY(cname),
    FOREIGN KEY(fid) REFERENCES faculty(fid));
```

```
desc class;
```

```
CREATE TABLE enrolled(
    snum INT,
    cname VARCHAR(20),
    PRIMARY KEY(snum,cname),
    FOREIGN KEY(snum) REFERENCES student(snum),
    FOREIGN KEY(cname) REFERENCES class(cname));
```

```
desc enrolled;
```

```
INSERT INTO STUDENT VALUES(1, 'John', 'CS', 'Sr', 19);
INSERT INTO STUDENT VALUES(2, 'Smith', 'CS', 'Jr', 20);
INSERT INTO STUDENT VALUES(3 , 'Jacob', 'CV', 'Sr', 20);
INSERT INTO STUDENT VALUES(4, 'Tom ', 'CS', 'Jr', 20);
INSERT INTO STUDENT VALUES(5, 'Rahul', 'CS', 'Jr', 20);
INSERT INTO STUDENT VALUES(6, 'Rita', 'CS', 'Sr', 21);
commit;
```

```
select * from student;
```

```
INSERT INTO FACULTY VALUES(11, 'Harish', 1000);
INSERT INTO FACULTY VALUES(12, 'MV', 1000);
INSERT INTO FACULTY VALUES(13 , 'Mira', 1001);
INSERT INTO FACULTY VALUES(14, 'Shiva', 1002);
INSERT INTO FACULTY VALUES(15, 'Nupur', 1000);
commit;
```

```
select * from faculty;
```

```
insert into class values('class1', '12/11/15 10:15:16', 'R1', 14);
insert into class values('class10', '12/11/15 10:15:16', 'R128', 14);
insert into class values('class2', '12/11/15 10:15:20', 'R2', 12);
insert into class values('class3', '12/11/15 10:15:25', 'R3', 11);
insert into class values('class4', '12/11/15 20:15:20', 'R4', 14);
insert into class values('class5', '12/11/15 20:15:20', 'R3', 15);
insert into class values('class6', '12/11/15 13:20:20', 'R2', 14);
insert into class values('class7', '12/11/15 10:10:10', 'R3', 14);
commit;
```

```
select * from class;
```

```
insert into enrolled values(1, 'class1');
insert into enrolled values(2, 'class1');
insert into enrolled values(3, 'class3');
insert into enrolled values(4, 'class3');
insert into enrolled values(5, 'class4');
insert into enrolled values(1, 'class5');
insert into enrolled values(2, 'class5');
insert into enrolled values(3, 'class5');
insert into enrolled values(4, 'class5');
```

```
insert into enrolled values(5, 'class5');  
commit;
```

```
select * from enrolled;
```

## STUDENT TABLE

The screenshot shows the MySQL Workbench interface with the 'Yashaswini' database selected. The 'Query' tab is active, displaying the following SQL code:

```
56 • insert into faculty values(11,'Harish',1000);  
57 • insert into faculty values(12,'MV',1000);  
58 • insert into faculty values(13,'Mira',1001);  
59 • insert into faculty values(14,'Shiva',1002);  
60 • insert into faculty values(15,'Nupur',1000);  
61  
62 • commit;  
63 • select * from faculty;  
64
```

The results grid shows the following data for the 'faculty' table:

fid	fname	deptid
11	Harish	1000
12	MV	1000
13	Mira	1001
14	Shiva	1002
15	Nupur	1000
NULL	NULL	NULL

The 'Output' pane at the bottom shows the execution details:

Action Output	#	Time	Action	Message	Duration / Fetch
47	23.04.00		select * from faculty LIMIT 0, 10	5 row(s) returned	0.031 sec / 0.000 sec
48	23.04.14		select * from faculty LIMIT 0, 10	5 rows returned	0.000 sec / 0.000 sec

## CLASS TABLE

MySQL Workbench - Yashaswini

File Edit View Query Database Server Tools Scripting Help

labprogram1 labprogram2 labprogram3\* labprogram4\* labprogram5\*

```

68 • insert into class values('class1','12/11/15 10:15:16','R1',14);
69 • insert into class values('class2','12/11/15 10:15:20','R2',12);
70 • insert into class values('class10','12/11/15 10:15:16','R128',14);
71 • insert into class values('class3','12/11/15 10:15:25','R3',12);
72 • insert into class values('class4','12/11/15 20:15:20','R4',14);
73 • insert into class values('class5','12/11/15 20:15:20','R3',15);
74 • insert into class values('class6','12/11/15 13:20:20','R2',14);
75 • insert into class values('class7','12/11/15 10:10:10','R3',14);
76
77 • select * from class;
    
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content: | Context Help Snippets

cname	metts_at	room	fid
class1	2012-11-15 10:15:16	R1	14
class10	2012-11-15 10:15:16	R128	14
class2	2012-11-15 10:15:20	R2	12
class3	2012-11-15 10:15:25	R3	12
class4	2012-11-15 20:15:20	R4	14
class5	2012-11-15 20:15:20	R3	15
class6	2012-11-15 13:20:20	R2	14
class7	2012-11-15 10:10:10	R3	14
class8		HULL	HULL
class9		HULL	HULL
class10		HULL	HULL

Output:

Action Output

#	Time	Action	Message	Duration / Fetch
1	22-Nov-15	insert into class values('class7','12/11/15 10:10:10','R2',14)	1 row(s) affected	0.000 sec

SQLAdditions... | Jump to | Context Help

## ENROLLED TABLE

MySQL Workbench - Yashaswini

File Edit View Query Database Server Tools Scripting Help

labprogram1 labprogram2 labprogram3\* labprogram4\* labprogram5\*

```

81 • insert into enrolled values(1,'class1');
82 • insert into enrolled values(2,'class1');
83 • insert into enrolled values(3,'class3');
84 • insert into enrolled values(4,'class3');
85 • insert into enrolled values(5,'class4');
86 • insert into enrolled values(1,'class5');
87 • insert into enrolled values(2,'class5');
88 • insert into enrolled values(3,'class5');
89 • insert into enrolled values(4,'class5');
90 • insert into enrolled values(5,'class5');
91
    
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content: | Fetch rows: | Context Help Snippets

snum	cname
1	class1
2	class1
3	class3
4	class3
5	class4
1	class5
2	class5
3	class5
4	class5
5	class5

enrolled 10 | Output: | Context Help

SQLAdditions... | Jump to | Context Help

## FACULTY TABLE

MySQL Workbench

Yashaswini

File Edit View Query Database Server Tools Scripting Help

labprogram1\* labprogram2 labprogram3\* labprogram4\* labprogram5\*

56  
57 • insert into faculty values(11,'Harish',1000);  
58 • insert into faculty values(12,'MV',1000);  
59 • insert into faculty values(13,'Mira',1001);  
60 • insert into faculty values(14,'Shiva',1002);  
61 • insert into faculty values(15,'Nupur',1000);  
62  
63 • commit;  
64  
65 • select \* from faculty;  
66  
67

Result Grid | Filter Rows: [ ] | Edit: [ ] | Export/Import: [ ] | Wrap Cell Content: [ ]

fid	fname	deptid
11	Harish	1000
12	MV	1000
13	Mira	1001
14	Shiva	1002
15	Nupur	1000

faculty 14 faculty 15

Output: [ ] Apply Revert

i. Find the names of all Juniors (level = JR) who are enrolled in a class taught by

SELECT DISTINCT S.Sname

FROM Student S, Class C, Enrolled E, Faculty F  
WHERE S.snum = E.snum AND E cname = C cname AND C fid = F fid AND  
F fname = 'Harish' AND S lvl = 'Jr';

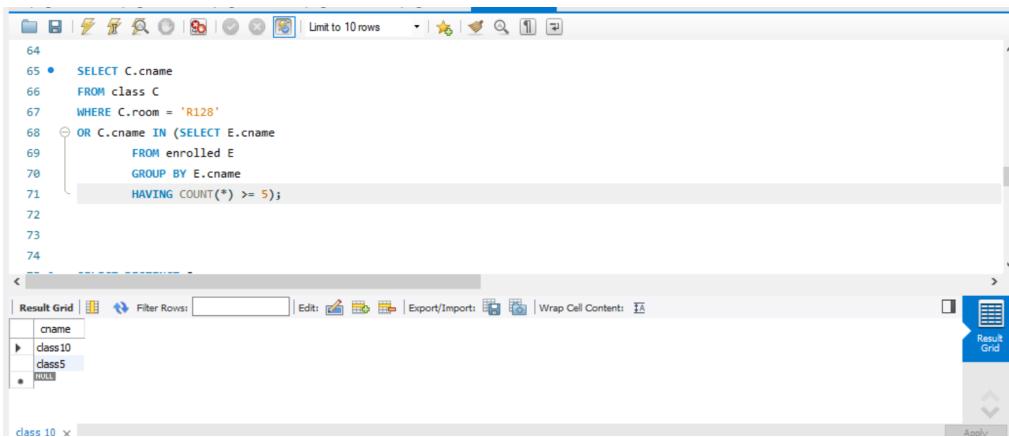
54  
55  
56 • SELECT DISTINCT S.Sname  
FROM Student S, Class C, Enrolled E, Faculty F  
WHERE S.snum = E.snum AND E cname = C cname AND C fid = F fid AND  
F fname = 'Harish' AND S lvl = 'Jr';  
60  
61  
62  
63  
64

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

Sname
Tom

**ii. Find the names of all classes that either meet in room R128 or have five or more Students enrolled.**

```
SELECT C cname
FROM class C
WHERE C.room = 'R128'
OR C cname IN (SELECT E cname
                FROM enrolled E
                GROUP BY E cname
                HAVING COUNT(*) >= 5);
```



The screenshot shows a database query editor window. The top pane displays the SQL code with line numbers 64 through 74. Lines 65-71 represent the main query, while lines 68-71 show the subquery being expanded. The bottom pane shows the 'Result Grid' with one row of data:

cname
class10

**iii. Find the names of all students who are enrolled in two classes that meet at the same time.**

```
SELECT DISTINCT S.sname
FROM student S
WHERE S.snum IN (SELECT E1.snum
                  FROM enrolled E1, enrolled E2, class C1, class C2
```

```

WHERE E1.snum = E2.snum AND E1 cname <> E2 cname
AND E1 cname = C1 cname
AND E2 cname = C2 cname AND C1 metts_at = C2 metts_at;

```

```

72
73
74
75 • SELECT DISTINCT S.sname
76   FROM student S
77   WHERE S.snum IN (SELECT E1.snum
78     FROM enrolled E1, enrolled E2, class C1, class C2
79     WHERE E1.snum = E2.snum AND E1.cname <> E2.cname
80     AND E1.cname = C1.cname
81     AND E2.cname = C2.cname AND C1.metts_at = C2.metts_at);
82

```

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

sname
Rahul

**iv. Find the names of faculty members who teach in every room in which some class is taught.**

```

SELECT f.fname,f.fid
FROM faculty f
WHERE f.fid in ( SELECT fid FROM class
GROUP BY fid HAVING COUNT(*)=(SELECT COUNT(DISTINCT room) FROM class ) );

```

```

82
83
84
85 • SELECT f.fname,f.fid
86   FROM faculty f
87   WHERE f.fid in ( SELECT fid FROM class
88     GROUP BY fid HAVING COUNT(*)=(SELECT COUNT(DISTINCT room) FROM class ) );
89
90
91
92

```

Result Grid | Filter Rows: [ ] Edit: [ ] Export/Import: [ ] Wrap Cell Content: [ ]

fname	fid
Shiva	14
*	HULL

v. Find the names of faculty members for whom the combined enrollment of the courses that they teach is less than five.

```
SELECT DISTINCT F.fname  
FROM faculty F  
WHERE 5 > (SELECT COUNT(E.snum)  
FROM class C, enrolled E  
WHERE C cname = E cname  
AND C fid = F fid);
```

fname
Harish
MV
Mira
Shiva

vi. Find the names of students who are not enrolled in any class.

```
SELECT DISTINCT S.sname  
FROM Student S  
WHERE S.snum NOT IN (SELECT E.snum  
FROM Enrolled E );
```

```

99
100
101
102 • SELECT DISTINCT S.sname
103   FROM Student S
104   WHERE S.snum NOT IN (SELECT E.snum
105     FROM Enrolled E );
106
107
108
109

```

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

sname
Rita

Student 14 x

Output: Action Output

vii. For each age value that appears in Students, find the level value that appears most often. For example, if there are more FR level students aged 18 than SR, JR, or SO students aged 18, you should print the pair (18, FR).

select distinct(st.age),st.lvl from student st

where st.lvl = (select s.lvl from student s

where s.age=st.age group by s.lvl order by count(s.lvl) desc limit 1);

```

134
135
136
137 • select distinct(st.age),st.lvl from student st
138   where st.lvl = (select s.lvl from student s
139     where s.age=st.age group by s.lvl order by count(s.lvl) desc limit 1);
140
141
142
143
144

```

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

age	lvl
19	Sr
20	Jr
21	Sr

Result 1 x

Output: Action Output

---

## **PROGRAM 5: AIRLINE FLIGHT DATABASE**

Consider the following database that keeps track of airline flight information:

**FLIGHTS(flno: integer, from: string, to: string, distance: integer, departs: time, arrives: time, price: integer)**  
**AIRCRAFT(aid: integer, aname: string, cruisingrange: integer)**  
**CERTIFIED(eid: integer, aid: integer)**  
**EMPLOYEES(eid: integer, ename: string, salary: integer)**

---

```
create database AirlineFlight;
use AirlineFlight;
```

```
CREATE TABLE FLIGHTS
(FLNO INTEGER PRIMARY KEY,
FFROM VARCHAR(15) ,
TTO VARCHAR(15) ,
DISTANCE INTEGER,
DEPARTS TIMESTAMP,
ARRIVES TIMESTAMP,
PRICE INTEGER );
```

```
DESC FLIGHTS;
```

```
CREATE TABLE AIRCRAFT
(AID INTEGER PRIMARY KEY,
ANAME VARCHAR(10),
CRUISINGRANGE INTEGER);
```

```
DESC AIRCRAFT;
```

```
CREATE TABLE EMPLOYEES  
(EID INTEGER PRIMARY KEY,  
ENAME VARCHAR(15),  
SALARY INTEGER );
```

```
DESC EMPLOYEES;
```

```
CREATE TABLE CERTIFIED  
(EID INTEGER NOT NULL,  
AID INTEGER NOT NULL,  
PRIMARY KEY (EID, AID),  
FOREIGN KEY (EID) REFERENCES EMPLOYEES (EID),  
FOREIGN KEY (AID) REFERENCES AIRCRAFT (AID));
```

```
DESC CERTIFIED;
```

```
COMMIT;
```

```
insert into aircraft values(101,'747',3000);  
insert into aircraft values(102,'Boeing',900);  
insert into aircraft values(103,'647',800);  
insert into aircraft values(104,'Dreamliner',10000);  
insert into aircraft values(105,'Boeing',3500);  
insert into aircraft values(106,'707',1500);  
insert into aircraft values(107,'Dream', 120000);
```

```
select * from aircraft;
```

```
insert into employees values(701,'A',50000);  
insert into employees values(702,'B',100000);  
insert into employees values(703,'C',150000);  
insert into employees values(704,'D',90000);  
insert into employees values(705,'E',40000);  
insert into employees values(706,'F',60000);  
insert into employees values(707,'G',90000);
```

```
select * from employees;

insert into certified values(701,101);
insert into certified values(701,102);
insert into certified values(701,106);
insert into certified values(701,105);
insert into certified values(702,104);
insert into certified values(703,104);
insert into certified values(704,104);
insert into certified values(702,107);
insert into certified values(703,107);
insert into certified values(704,107);
insert into certified values(702,101);
insert into certified values(703,105);
insert into certified values(704,105);
insert into certified values(705,103);
```

```
select * from certified;
```

```
insert into flights values(101,'Bangalore','Delhi',2500,TIMESTAMP '2005-05-13 07:15:31',TIMESTAMP
'2005-05-13 17:15:31',5000);
insert      into      flights      values(102,'Bangalore','Lucknow',3000,TIMESTAMP      '2005-05-13
07:15:31',TIMESTAMP '2005-05-13 11:15:31',6000);
insert      into      flights      values(103,'Lucknow','Delhi',500,TIMESTAMP '2005-05-13 12:15:31',TIMESTAMP '
2005-05-13 17:15:31',3000);
insert      into      flights      values(107,'Bangalore','Frankfurt',8000,TIMESTAMP      '2005-05-13
07:15:31',TIMESTAMP '2005-05-13 22:15:31',60000);
insert      into      flights      values(104,'Bangalore','Frankfurt',8500,TIMESTAMP      '2005-05-13
07:15:31',TIMESTAMP '2005-05-13 23:15:31',75000);
insert      into      flights      values(105,'Kolkata','Delhi',3400,TIMESTAMP '2005-05-13 07:15:31',TIMESTAMP
'2005-05-13 09:15:31',7000);
```

```
select * from Flights;
```

**Note that the Employees relation describes pilots and other kinds of employees as well;  
Every pilot is certified for some aircraft, and only pilots are certified to fly.**

**Write each of the following queries in SQL.**

- i. **Find the names of aircraFor each pilot who is certified for more than three aircrafts, find the eid and the maximum cruisingrange of the aircraft for which she or he is certified.**

```
SELECT C.eid, MAX(A.cruisingrange)
FROM Certified C, Aircraft A
WHERE C.aid = A.aid
GROUP BY C.eid
HAVING COUNT(*) > 3;
```

The screenshot shows the MySQL Workbench interface. The title bar says "MySQL Workbench" and "Yashaswini". The menu bar includes File, Edit, View, Query, Database, Server, Tools, Scripting, Help. The toolbar has various icons for database management. The query editor tab is "labprogram5\*". The code in the editor is:

```
95  
96  
97  
98 •  SELECT C.eid, MAX(A.cruisingrange)  
99   FROM Certified C, Aircraft A  
100  WHERE C.aid = A.aid  
101  GROUP BY C.eid  
102  HAVING COUNT(*) > 3;  
103  
104  
105  
106  
107
```

The results grid shows one row:

eid	MAX(A.cruisingrange)
701	3500

Below the results grid, there is a "Result 15" tab and an "Output" tab.

- ii. Find the names of pilots whose salary is less than the price of the cheapest route from Bengaluru to Frankfurt.

```
SELECT DISTINCT E.ename  
FROM Employees E  
WHERE E.salary <( SELECT MIN(F.price)  
FROM Flights F  
WHERE F.ffrom = 'Bangalore' AND F.tto = 'Frankfurt' );
```

The screenshot shows the MySQL Workbench interface. The title bar says "MySQL Workbench" and "Yashaswini". The menu bar includes File, Edit, View, Query, Database, Server, Tools, Scripting, and Help. Below the menu is a toolbar with various icons. The query editor tab is "labprogram5\*". The code in the editor is:

```

184
185 •  SELECT DISTINCT E.ename
186   FROM Employees E
187   WHERE E.salary < ( SELECT MIN(F.price)
188     FROM Flights F
189     WHERE F.ffrom = 'Bangalore' AND F.tto = 'Frankfurt' );
110
111
112
113
114
115
116

```

The results grid shows the output of the query:

ename
A
E

Below the results grid, it says "Employees 16" and has a "Read Only" status indicator.

- iii. **For all aircraft with cruisingrange over 1000 Kms, find the name of the aircraft and the average salary of all pilots certified for this aircraft.**

```

SELECT Temp.name, Temp.AvgSalary
FROM ( SELECT A.aid, A.aname AS name, AVG (E.salary) AS AvgSalary
      FROM Aircraft A, Certified C, Employees E
      WHERE A.aid = C.aid AND C.eid = E.eid AND A.cruisingrange > 1000
      GROUP BY A.aid, A.aname ) Temp;

```

MySQL Workbench

Yashaswini

File Edit View Query Database Server Tools Scripting Help

labprogram1\* labprogram2 labprogram3\* labprogram4\* labprogram5\*

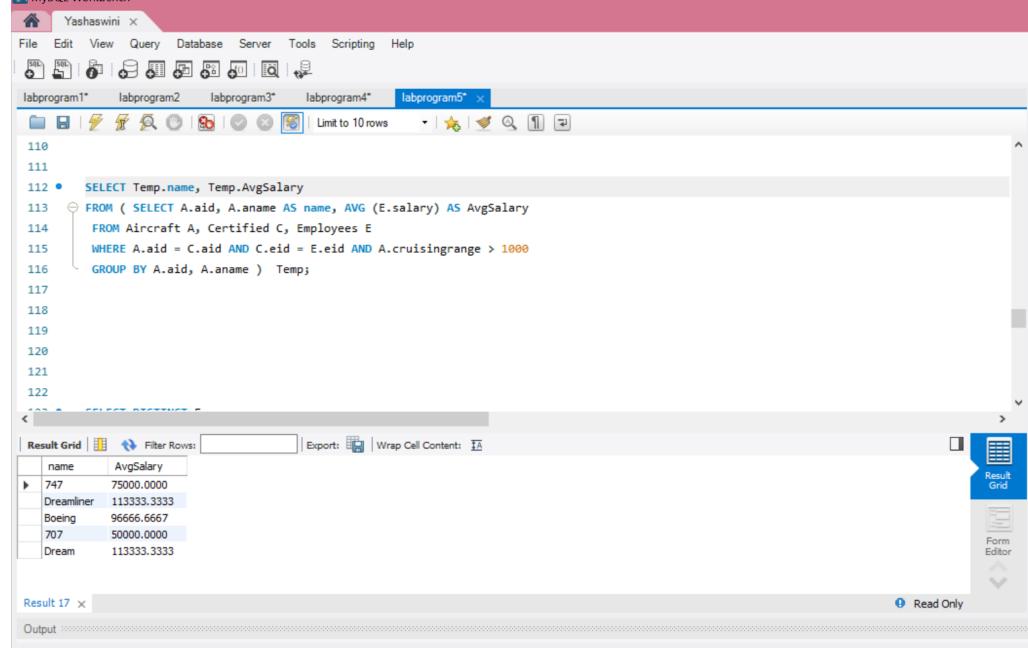
```
110
111
112 •  SELECT Temp.name, Temp.AvgSalary
113   FROM ( SELECT A.aid, A.aname AS name, AVG (E.salary) AS AvgSalary
114     FROM Aircraft A, Certified C, Employees E
115    WHERE A.aid = C.aid AND C.eid = E.eid AND A.cruisingrange > 1000
116   GROUP BY A.aid, A.aname ) Temp;
117
118
119
120
121
122
```

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

name	AvgSalary
747	75000.0000
Dreamliner	113333.3333
Boeing	96666.6667
707	50000.0000
Dream	113333.3333

Result 17 x Read Only

Output



iv. Find the names of pilots certified for some Boeing aircraft.

```
SELECT DISTINCT E.ename
FROM Employees E, Certified C, Aircraft A
WHERE E.eid = C.eid AND C.aid = A.aid AND A.aname LIKE 'Boeing%';
```

MySQL Workbench

Yashaswini

File Edit View Query Database Server Tools Scripting Help

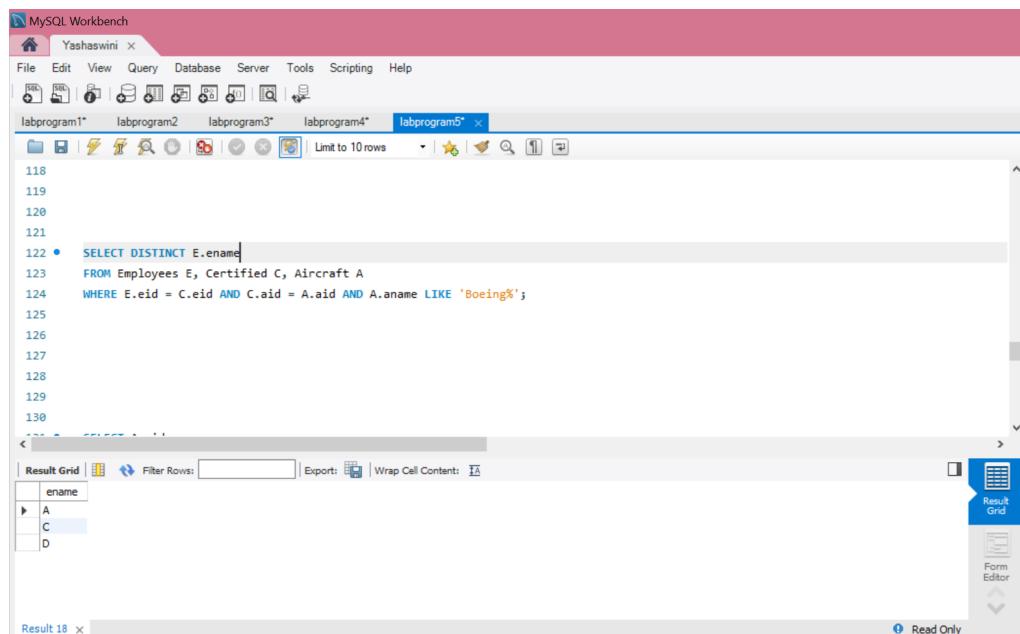
labprogram1\* labprogram2 labprogram3\* labprogram4\* labprogram5\*

```
118
119
120
121
122 •  SELECT DISTINCT E.ename
123   FROM Employees E, Certified C, Aircraft A
124  WHERE E.eid = C.eid AND C.aid = A.aid AND A.aname LIKE 'Boeing%';
125
126
127
128
129
130
```

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

ename
A
C
D

Result 18 x Read Only



- v. Find the aids of all aircraft that can be used on routes from Bengaluru to Frankfurt.

```
SELECT A.aid  
FROM Aircraft A  
WHERE A.cruisingrange >( SELECT MIN(F.distance)  
                           FROM Flights F  
                           WHERE F.ffrom = 'Bangalore' AND F.tto = 'Frankfurt' );
```

The screenshot shows the MySQL Workbench interface with the query editor tab open. The query is the same as the one above. The results pane shows a single column named 'aid' with two rows: 104 and 107.

aid
104
107

- vi. A customer wants to travel from Madison to New York with no more than two changes of flight. List the choice of departure times from Madison if the customer wants to arrive in New York by 6 p.m.

```
SELECT F.departs
```

```
FROM Flights F
```

```
WHERE F.flno IN ( ( SELECT F0.flno FROM Flights F0 WHERE F0.ffrom = 'Bangalore' AND F0.tto = 'Delhi' AND extract(hour from F0.arrives) < 18 )
```

```
UNION
```

```

( SELECT F0.flno FROM Flights F0, Flights F1 WHERE F0.ffrom = 'Bangalore' AND F0.tto <> 'Delhi'
AND F0.tto = F1.ffrom AND F1.tto = 'Delhi' AND F1.deperts > F0.arrives AND extract(hour from
F1.arrives) < 18)

UNION

( SELECT F0.flno FROM Flights F0, Flights F1, Flights F2 WHERE F0.ffrom = 'Bangalore'
AND F0.tto = F1.ffrom AND F1.tto = F2.ffrom
AND F2.tto = 'Delhi' AND F0.tto <> 'Delhi'
AND F1.tto <> 'Delhi' AND F1.deperts > F0.arrives
AND F2.deperts > F1.arrives AND extract(hour from F2.arrives) < 18));

```

The screenshot shows the MySQL Workbench interface. The query editor window displays the complex SQL query provided above. The results grid below shows a single column named 'departs' with two rows of data: '2005-05-13 07:15:31' and '2005-05-13 07:15:31'. The bottom status bar indicates 'Result 25' and 'Result 26'.

```

135 •   SELECT F.deperts
136   FROM Flights F
137   WHERE F.flno IN ( ( SELECT F0.flno
138     FROM Flights F0
139     WHERE F0.ffrom = 'Bangalore' AND F0.tto = 'Delhi'
140     AND extract(hour from F0.arrives) < 18 )
141   UNION
142   ( SELECT F0.flno
143     FROM Flights F0, Flights F1
144     WHERE F0.ffrom = 'Bangalore' AND F0.tto <> 'Delhi'
145     AND F0.tto = F1.ffrom AND F1.tto = 'Delhi'
146     AND F1.deperts > F0.arrives
147     AND extract(hour from F1.arrives) < 18 )
148   UNION
149   ( SELECT F0.flno

```

**Print the name and salary of every non-pilot whose salary is more than the average salary for pilots.**

```

SELECT E.ename, E.salary
FROM Employees E
WHERE E.eid NOT IN ( SELECT DISTINCT C.eid
FROM Certified C )
AND E.salary >( SELECT AVG (E1.salary)
FROM Employees E1
WHERE E1.eid IN
( SELECT DISTINCT C1.eid
FROM Certified C1 ) );

```

The screenshot shows the MySQL Workbench interface with the following details:

- Title Bar:** MySQL Workbench - Yashaswini
- Toolbar:** Standard MySQL Workbench toolbar with icons for file operations, database management, and search.
- Script Editor:** The main area displays the SQL query with line numbers 160 through 174. Lines 163-171 represent the main query, while lines 162 and 172-174 are comments.
- Result Grid:** A table titled "Result Grid" showing the output of the query. It has two columns: "ename" and "salary". The single row shows "G" in the "ename" column and "90000" in the "salary" column.
- Status Bar:** Shows "Employees 30" and "Read Only".

## **PROGRAM 6: ORDER DATABASE**

**SALESMAN (*Salesman\_id, Name, City, Commission*)**

**CUSTOMER (*Customer\_id, Cust\_Name, City, Grade, Salesman\_id*)**

**ORDERS (*Ord\_No, Purchase\_Amt, Ord\_Date, Customer\_id, Salesman\_id*)**

**Write SQL queries to**

- 1. Count the customers with grades above Bangalore's average.**
- 2. Find the name and numbers of all salesmen who had more than one customer.**
- 3. List all salesmen and indicate those who have and don't have customers in their cities (Use UNION operation.)**
- 4. Create a view that finds the salesman who has the customer with the highest order of a day.**

```
CREATE DATABASE ORDERDATABASE;
USE ORDERDATABASE;
CREATE TABLE SALESMAN(
SALESMANID INT,
SNAME VARCHAR(30),
CITY VARCHAR(30),
COMMISSION VARCHAR(30),
PRIMARY KEY (SALESMANID));
CREATE TABLE CUSTOMER(
CUSTOMERID INT,
SALESMANID INT,
CNAME VARCHAR(30),
CITY VARCHAR(20),
GRADE INT,
PRIMARY KEY (CUSTOMERID),
FOREIGN KEY (SALESMANID) REFERENCES SALESMAN(SALESMANID));
CREATE TABLE ORDERS(
ORDERNO INT,
CUSTOMERID INT,
SALESMANID INT,
PURCHASEAMT INT,
ORDERDATE DATE,
PRIMARY KEY (ORDERNO),
FOREIGN KEY (SALESMANID) REFERENCES SALESMAN(SALESMANID),
FOREIGN KEY (CUSTOMERID) REFERENCES CUSTOMER(CUSTOMERID));

INSERT INTO SALESMAN VALUES(1000,'JOHN','BANGALORE','25%');
```

```
INSERT INTO SALESMAN VALUES(2000,'RAVI','BANGALORE','20%');
INSERT INTO SALESMAN VALUES(3000,'KUMAR','MYSORE','15%');
INSERT INTO SALESMAN VALUES(4000,'SMITH','DELHI','30%');
INSERT INTO SALESMAN VALUES(5000,'HARSHA','HYDERABAD','15%');

INSERT INTO CUSTOMER VALUES(10,1000,'PREETHI','BANGALORE',100);
INSERT INTO CUSTOMER VALUES(11,2000,'VIVEK','MANGALORE',300);
INSERT INTO CUSTOMER VALUES(12,2000,'BHASKAR','CHENNAI',400);
INSERT INTO CUSTOMER VALUES(13,3000,'CHETAN','BANGALORE',200);
INSERT INTO CUSTOMER VALUES(14,2000,'MAMTHA','BANGALORE',400);

INSERT INTO ORDERS VALUES(50,10,1000,5000,'2017-05-04');
INSERT INTO ORDERS VALUES(51,10,2000,450,'2017-01-20');
INSERT INTO ORDERS VALUES(52,13,2000,1000,'2017-02-24');
INSERT INTO ORDERS VALUES(53,14,3000,3500,'2017-04-13');
INSERT INTO ORDERS VALUES(54,12,2000,550,'2017-03-09');
```

-- 1. Count the customers with grades above Bangalore's average.

```
SELECT GRADE, COUNT(*)
FROM CUSTOMER
GROUP BY GRADE
HAVING GRADE >
(SELECT AVG(GRADE)
 FROM CUSTOMER
 WHERE CITY = 'BANGALORE');
```

```

-- 1. Count the customers with grades above Bangalore's average.

SELECT GRADE, COUNT(*)
FROM CUSTOMER
GROUP BY GRADE
HAVING GRADE >
    (SELECT AVG(GRADE)
     FROM CUSTOMER
     WHERE CITY = 'BANGALORE');

```

GRADE	COUNT(*)
300	1
400	2

Action Output

#	Time	Action	Message	Duration / Fetch
21	10:45:02	INSERT INTO ORDERS VALUES(64,12,2000,550,'2017-03-09')	1 row(s) affected	0.000 sec
22	10:45:07	SELECT GRADE,COUNT(*) FROM CUSTOMER GROUP BY GRADE HAVING G...	2 row(s) returned	0.000 sec / 0.000 sec

-- 2. Find the name and numbers of all salesmen who had more than one customer.

```

SELECT SALESMANID, SNAME
FROM SALESMAN A
WHERE 1 < (SELECT COUNT(*)
FROM CUSTOMER
WHERE SALESMANID=A.SALESMANID);

```

```

54
55 -- 2.Find the name and numbers of all salesmen who had more than one customer.
56
57 • SELECT SALESMANID, SNAME
58   FROM SALESMAN A
59   WHERE 1 < (SELECT COUNT(*)
60     FROM CUSTOMER
61     WHERE SALESMANID=A.SALESMANID);
62
63
64 -- 3. List all salesmen and indicate those who have and don't have customers in their cities (Use UNION operation.)

```

SALESMANID	SNAME
2000	RAVI
NULL	NULL

SALESMAN 2 x

Action Output

#	Time	Action	Message	Duration / Fetch
22	10:45:07	SELECT GRADE,COUNT(*) FROM CUSTOMER GROUP BY GRADE HAVING G...	2 row(s) returned	0.000 sec / 0.000 sec
23	10:45:59	SELECT SALESMANID, SNAME FROM SALESMAN A WHERE 1 < (SELECT CO...	1 row(s) returned	0.000 sec / 0.000 sec

**-- 3. List all salesmen and indicate those who have and don't have customers in their cities (Use UNION operation.)**

```

SELECT SALESMAN.SALESMANID,SNAME, CNAME, COMMISSION
FROM SALESMAN, CUSTOMER
WHERE SALESMAN.CITY = CUSTOMER.CITY
UNION
SELECT SALESMANID, SNAME, 'NO MATCH', COMMISSION FROM SALESMAN
    WHERE NOT CITY = ANY
    (SELECT CITY
     FROM CUSTOMER)
ORDER BY 2 DESC;

```

The screenshot shows the MySQL Workbench interface. The top menu bar includes File, Edit, View, Query, Database, Server, Tools, Scripting, and Help. The title bar says "MySQL Workbench" and "Yashaswini". The left sidebar has a "Navigator" section with "Schemas" expanded, showing databases like airlineflight, banking, banking\_enterprise, insurance, and orderdatabase. Below it are "Tables", "Views", "Stored Procedures", and "Functions". Other sections include "Administration" and "Schemas". The main area contains a query editor with the following SQL code:

```

-- 3. List all salesmen and indicate those who have and don't have customers in their cities (Use UNION op ^

64 -- 3. List all salesmen and indicate those who have and don't have customers in their cities (Use UNION op ^
65
66 • SELECT SALESMAN.SALESMANID,SNAME, CNAME, COMMISSION
67 FROM SALESMAN, CUSTOMER
68 WHERE SALESMAN.CITY = CUSTOMER.CITY
69 UNION
70 SELECT SALESMANID, SNAME, 'NO MATCH', COMMISSION FROM SALESMAN
71     WHERE NOT CITY = ANY
72         (SELECT CITY
73             FROM CUSTOMER)
74     ORDER BY 2 DESC;

```

The results grid shows the following data:

SALESMANID	SNAME	CNAME	COMMISSION
4000	SMITH	NO MATCH	30%
2000	RAVI	PREETHI	20%
2000	RAVI	CHESTAN	20%
2000	RAVI	MAMTHA	20%
3000	KUMAR	NO MATCH	15%
1000	JOHN	PREETHE	25%
1000	JOHN	CHESTAN	25%
1000	JOHN	MAMTHA	25%
5000	HARSHA	NO MATCH	15%

-- 4. Create a view that finds the salesman who has the customer with the highest order of a day.

```

CREATE VIEW VIEWSALESMAN AS
SELECT B.ORDERDATE, A.SALESMANID, A.SNAME
FROM SALESMAN A, ORDERS B
WHERE A.SALESMANID = B.SALESMANID
AND B.PURCHASEAMT=(SELECT MAX (PURCHASEAMT)
FROM ORDERS C
WHERE C.ORDERDATE = B.ORDERDATE);

```

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

```

69 UNION
70   SELECT SALESMANID, SNAME, 'NO MATCH', COMMISSION FROM SALESMAN
71     WHERE NOT CITY = ANY
72       (SELECT CITY
73        FROM CUSTOMER)
74      ORDER BY 2 DESC;
75
76 -- 4. Create a view that finds the salesman who has the customer with the highest order of a day.
77
78 • CREATE VIEW VIEWSALESMAN AS
79   SELECT B.ORDERDATE, A.SALESMANID, A.SNAME
80   FROM SALESMAN A, ORDERS B
81   WHERE A.SALESMANID = B.SALESMANID
82   AND B.PURCHASEAMT=(SELECT MAX (PURCHASEAMT)
83   FROM ORDERS C
84   WHERE C.ORDERDATE = B.ORDERDATE);

```

The output pane shows the results of the last two statements:

Action	Time	Action	Message	Duration / Fetch
23	10:45:59	SELECT SALESMANID, SNAME FROM SALESMAN A WHERE 1 < (SELECT CO...	1 row(s) returned	0.000 sec / 0.000 sec
24	10:46:18	SELECT SALESMAN.SALESMANID,SNAME,CNAME,COMMISSION FROM SAL...	9 row(s) returned	0.015 sec / 0.000 sec

## PROGRAM NUMBER 7: BOOK DEALER

**AUTHOR(author-id: int, name: String, city: String, country: String)**

**PUBLISHER(publisher-id: int, name: String, city: String, country: String)**

**CATALOG (book-id: int, title: String, author-id: int, publisher-id: int, category-id: int, year: int, price: int)**

**CATEGORY(category-id: int, description: String)**

**ORDER-DETAILS(order-no: int, book-id: int, quantity: int)**

- Create the above tables by properly specifying the primary keys and the foreign keys.
  - Enter at least five tuples for each relation.
  - Give the details of the authors who have 2 or more books in the catalog and the price of the books in the catalog and the year of publication is after 2000.
  - Find the author of the book which has maximum sales.
  - Demonstrate how you increase the price of books published by a specific publisher by 10%.
- create database book\_dealer;

```

use book_dealer;
create table author
author_id int,
name varchar(30),

```

```

city varchar(20),
country varchar(20),
constraint auth_id primary key (author_id)
);
create table publisher(
publisher_id int,
name varchar(30),
city varchar (30),
country varchar(30),
constraint pub_id primary key (publisher_id)
);

create table category(
category_id int,
description varchar(30),
constraint primary key (category_id)
);
create table catalog(
book_id int ,
title varchar(30),
author_id int,
publisher_id int ,
category_id int,
year int,
price int,
constraint cat_id primary key(book_id),
constraint cat_auth foreign key (author_id) references author(author_id) on delete cascade on
update cascade,
constraint cat_pub foreign key (publisher_id) references publisher(publisher_id) on delete
cascade on update cascade ,
constraint cat_log foreign key (category_id) references category(category_id) on delete cascade
on update cascade
);
create table order_details (
order_no int,
book_id int,
quantity int,
constraint ord_no primary key (order_no),
constraint ord_id foreign key (book_id) references catalog(book_id) on delete cascade on
update cascade
);

```

**-- ii. Inserting tuples**

```
insert into author values(1001,'TERAS CHAN','CA','USA');
insert into author values(1002,'STEVENS','ZOMBI','UGANDA');
insert into author values(1003,'M MANO','CAIR','CANADA');
insert into author values(1004,'KARTHIK B.P','NEW YORK','USA');
insert into author values(1005,'WILLIAM STALLINGS','LAS VEGAS','USA');
```

```
insert into publisher values(1,'PEARSON','NEW YORK','USA');
insert into publisher values(2,'EEE','NEW SOUTH VALES','USA');
insert into publisher values(3,'PHI','DELHI','INDIA');
insert into publisher values(4,'WILLEY','BERLIN','GERMANY');
insert into publisher values(5,'MGH','NEW YORK','USA');
```

```
insert into category values(1001,'COMPUTER SCIENCE');
insert into category values(1002,'ALGORITHM DESIGN');
insert into category values(1003,'ELECTRONICS');
insert into category values(1004,'PROGRAMMING');
insert into category values(1005,'OPERATING SYSTEMS');
```

```
insert into catalog values(11,'Unix System Prg',1001,1,1001,2000,251);
insert into catalog values(12,'Digital Signals',1002,2,1003,2001,425);
insert into catalog values(13,'Logic Design',1003,3,1002,1999,225);
insert into catalog values(14,'Server Prg',1004,4,1004,2001,333);
insert into catalog values(15,'Linux OS',1005,5,1005,2003,326);
insert into catalog values(16,'C++ Bible',1005,5,1001,2000,526);
insert into catalog values(17,'COBOL Handbook',1005,4,1001,2000,658);
```

```
insert into order_details values(1,11,5);
insert into order_details values(2,12,8);
insert into order_details values(3,13,15);
insert into order_details values(4,14,22);
insert into order_details values(5,15,3);
insert into order_details values(6,17,10);
```

-- iii. Give the details of the authors who have 2 or more books in the catalog and the price of the books in the catalog and the year of publication is after cataloger 2000.select \* from author a where a.author\_id in (select c.author\_id from catalog c where c.author\_id=a.author\_id and year>2000 having count(\*)>=2);

```
insert into catalog values(18,'CP Handbook',1005,4,1001,2003,658);
select * from author a where a.author_id in (select c.author_id from catalog c where c.author_id=a.author_id and year>2000 having count(*)>=2);
```

MySQL Workbench

Yashaswini

File Edit View Query Database Server Tools Scripting Help

Schemas

author 4

No object selected

Output:

author_id	name	city	country
1005	WILLIAM STALLINGS	LAS VEGAS	USA
*	HULL	HULL	HULL

Result Grid

SQL Additions

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

-- iv. Find the author of the book which has maximum sales.

```
select a.name from author a where a.author_id = (select c.author_id from catalog c where
c.book_id = (select o.book_id from order_details o group by o.book_id order by sum(quantity)
desc limit 1));
```

MySQL Workbench

Yashaswini

File Edit View Query Database Server Tools Scripting Help

Schemas

author 5

No object selected

Output:

name
KARTHIK.B.P.

Result Grid

SQL Additions

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

-- v. Demonstrate how you increase the price of books published by a specific publisher by 10%.

```
select c.book_id,(1.10*price) as price from catalog c where c.publisher_id in (select p.publisher_id from publisher p where P.name="MGH");
```

The screenshot shows the MySQL Workbench interface. In the central pane, a SQL editor window displays the following code:

```
-- v. Demonstrate how you increase the price of books published by a specific publisher by 10%.
select c.book_id,(1.10*price) as price from catalog c where c.publisher_id in
(select p.publisher_id from publisher p where P.name="MGH");
```

The Result Grid shows the output of the query:

book_id	price
15	358.60
16	578.60

The Output pane shows the execution log:

Action	Time	Action	Message	Duration / Fetch
Action Output	#	Time	Action	
155	00:13:18	select a name from author a where a.author_id =(select c.author_id from catalog c ...	1 row(s) returned	0.016 sec / 0.000 sec
156	00:13:40	select c.book_id,(1.10*price) as price from catalog c where c.publisher_id in (select... 2 row(s) returned		0.016 sec / 0.000 sec

## PROGRAM 8: STUDENT ENROLLMENT DATABASE

Consider the following database of student enrollment in courses and books adopted for each course.

STUDENT (regno: String, name: String, major: String, bdate: date)

COURSE (course #: int, cname: String, dept: String)

ENROLL (regno: String, cname: String, sem: int, marks: int)

BOOK\_ADOPTION (course #: int, sem: int, book-ISBN: int)

TEXT(book-ISBN:int, book-title:String, publisher:String, author:String)

i. Create the above tables by properly specifying the primary keys and the foreign keys.

ii. Enter at least five tuples for each relation.

iii. Demonstrate how you add a new text book to the database and make this book be adopted by some department.

iv. Produce a list of textbooks (include Course #, Book-ISBN, Book-title) in the alphabetical order for courses offered by the 'CS' department that use more than two books.

v. List any department that has all its adopted books published by a specific publisher

```
create database student_enrollment;
```

```

use student_enrollment;

-- i) creating table
create table student(
regno varchar(10) ,
name varchar(30),
major varchar(10),
bdate date,
constraint stu_reg primary key (regno)
);
create table course (
course int ,
cname varchar(30),
dept varchar (30),
constraint cou_cou primary key (course)
);
create table enroll (
regno varchar(10) ,
cname varchar (30),
sem int ,
marks int,
constraint en_reg foreign key (regno) references student(regno)
);
create table text (
book_isbn int,
book_title varchar(30),
publisher varchar(30),
author varchar (30),
constraint book_book primary key (book_isbn)
);
create table book_adoption (
course int,
sem int,
book_isbn int,
constraint book_cou foreign key (course)
references course(course) on delete cascade on update cascade ,
constraint book_book foreign key (book_isbn)
references text(book_isbn) on delete cascade on update cascade
);
-- ii. Inserting tuples
insert into student values("CS01", "PRANAV", "DS", "1986-03-12");
insert into student values("IS02", "PRATEEK", "USP", "1987-12-23");
insert into student values("EC03", "SAURAB", "SNS", "1985-04-17");

```

```
insert into student values("CS03", "ARKA", "DBMS", "1987-01-01");
insert into student values("TC05", "PRANSHU", "EC", "1986-10-06");
insert into course values(11,"DS","CS");
insert into course values(22,"USP","IS");
insert into course values(33,"SNS","EC");
insert into course values(44,"DBMS","CS");
insert into course values(55,"EC","TC");
insert into enroll values("CS01", 11, 4, 85);
insert into enroll values("IS02", 22, 6, 80);
insert into enroll values("EC03", 33, 2, 80);
insert into enroll values("CS03", 44, 6, 75);
insert into enroll values("TC05", 11, 4, 85);
insert into text values(1, "DS AND C", "PRINCETON", "PADMA REDDY");
insert into text values(2, "FUNDAMENTALS OF DS", "SPRINGER", "GODSE");
insert into text values(3, "FUNDAMENTALS OF DBMS", "SPRINGER", "NAVATHE");
insert into text values(4, "SQL", "PRINCETON", "FOLEY");
insert into text values(5, "ELECTRONIC CIRCUITS", "TMH", "ELMASRI");
insert into book_adoption values(11, 4 ,1);
insert into book_adoption values(11, 4 ,2);
insert into book_adoption values(44, 6 ,3);
insert into book_adoption values(44, 6 ,4);
insert into book_adoption values(55, 2 ,5);
```

-- iii. Demonstrate how you add a new text book to the database and make this book be adopted by some department.

```
insert into text values(6, "RELATIONAL ALGEBRA", "ABS", "HUGH DARWIN");
insert into book_adoption values(22, 6 ,6);
select * from book_adoption;
```

The screenshot shows the MySQL Workbench interface. In the top navigation bar, the database 'Yashaswini' is selected. The main area displays a query editor with the following SQL code:

```

-- iii. Demonstrate how you add a new text book to the database and make this book be adopted by some department
insert into text values(6, "RELATIONAL ALGEBRA", "ABS", "HUGH DARWIN");
insert into book_adoption values(22, 6 ,6);
select * from book_adoption;

```

Below the code, the results are shown in a table:

course	sem	book_isbn
11	4	1
11	4	2
44	6	3
44	6	4
55	2	5
22	6	6
22	6	6

The 'Output' tab shows the execution log:

#	Time	Action	Message	Duration / Fetch
164	00:26:53	insert into book_adoption values(22, 6 ,6)	1 row(s) affected	0.016 sec
165	00:26:54	select * from book_adoption LIMIT 0, 10	7 row(s) returned	0.000 sec / 0.000 sec

-- iv. Produce a list of text books (include Course #, Book-ISBN, Book-title) in the alphabetical order for courses offered by the 'CS' department that use more than two books.

```

select b.book_isbn from book_adoption b where b.course in (select c.course from course c
where c.dept="CS") and (select count(*) from book_adoption be where be.course=b.course)>=2
;
insert into course values(66,"OS","CS");
insert into text values(7, "OPERATING SYSTEM", "MC GRAW","ABRAHAM
SILBERSCHATZ");
insert into book_adoption values(66, 6 ,7);
select * from book_adoption;

```

MySQL Workbench - Yashaswini

File Edit View Query Database Server Tools Scripting Help

Navigator: labprogram7 SQL File 6\* labprogram6\* labprogram9\* labprogram8\* SQL File 7\* SQL File 8\* SQL File 9\*

SCHEMAS: Filter objects

- airlineflight
- banking
- banking\_enterprise
- books
- college
- insurance
- orderdatabase
- Tables
- Views
- Stored Procedures

Administration Schemas

No object selected

```

78 -- iv. Produce a list of text books (include Course #, Book-ISBN, Book-title) in the alphabetical order fo
79
80 •      select b.book_isbn from book_adoption b where b.course in (select c.course from course c where c.dept=
81 •      insert into course values(66,"OS","CS"));
82 •      insert into text values(7, "OPERATING SYSTEM", "MC GRAW", "ABRAHAM SILBERSCHATZ");
83 •      insert into book_adoption values(66, 6 ,?);
84 •      select * from book_adoption;
85

```

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

course	sem	book_isbn
11	4	1
11	4	2
44	6	3
44	6	4
55	2	5
22	6	6
22	6	6

book\_adoption 7 x

Output:

#	Time	Action	Message	Duration / Fetch
165	00:26:54	select * from book_adoption LIMIT 0, 10	7 row(s) returned	0.000 sec / 0.000 sec
166	00:27:30	select * from book_adoption LIMIT 0, 10	7 row(s) returned	0.000 sec / 0.000 sec

Object Info Session

Type here to search

-- v. List any department that has all its adopted books published by a specific publisher.

```
select distinct c.dept from course c where not exists (select t.book_isbn from text t where t.publisher="SPRINGER") not in (select b.book_isbn from book_adoption b where b.course in (select ce.course from course ce where ce.dept=c.dept));
```

MySQL Workbench - Yashaswini

File Edit View Query Database Server Tools Scripting Help

Navigator: labprogram7 SQL File 6\* labprogram6\* labprogram9\* labprogram8\* SQL File 7\* SQL File 8\* SQL File 9\*

SCHEMAS: Filter objects

- airlineflight
- banking
- banking\_enterprise
- books
- college
- insurance
- orderdatabase
- Tables
- Views
- Stored Procedures

Administration Schemas

No object selected

```

85
86 -- v. List any department that has all its adopted books published by a specific publisher.
87
88 •      select distinct c.dept from course c where not exists
89 (select t.book_isbn from text t where t.publisher="SPRINGER")
90 not in (select b.book_isbn from book_adoption b where b.course in
91 (select ce.course from course ce where ce.dept=c.dept));
92

```

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

dept
CS

course 8 x

Output:

#	Time	Action	Message	Duration / Fetch
166	00:27:30	select * from book_adoption LIMIT 0, 10	7 row(s) returned	0.000 sec / 0.000 sec
167	00:28:19	select distinct c.dept from course c where not exists (select t.book_isbn from text ... 1 row(s) returned		0.000 sec / 0.000 sec

Object Info Session

---

USN:1BM19CS216

## PROGRAM 9 : MOVIE DATABASE

Consider the schema for Movie Database:

ACTOR(Act\_id, Act\_Name, Act\_Gender)  
DIRECTOR(Dir\_id, Dir\_Name, Dir\_Phone)  
MOVIES(Mov\_id, Mov\_Title, Mov\_Year, Mov\_Lang, Dir\_id)  
MOVIE\_CAST(Act\_id, Mov\_id, Role)  
RATING(Mov\_id, Rev\_Stars)

Write SQL queries to

- i. List the titles of all movies directed by ‘Hitchcock’.
- ii. Find the movie names where one or more actors acted in two or more movies.
- iii. List all actors who acted in a movie before 2000 and also in a movie after 2015 (use JOIN operation).
- iv. Find the title of movies and number of stars for each movie that has at least one rating and find the highest number of stars that movie received. Sort the result by movie title.
- v. Update rating of all movies directed by ‘Steven Spielberg’ to 5.

```
create database movies;
use movies;
```

```
create table Actor(
act_id int,
act_name varchar(30),
act_gender varchar(1),
constraint act_id primary key (act_id)
);
create table Director(
dir_id int,
dir_name varchar(30),
dir_phone varchar(30),
constraint dir_id primary key (dir_id)
);
create table Movies(
mov_id int,
mov_title varchar(30),
mov_year varchar(4),
mov_lang varchar(12),
dir_id int ,
```

```
constraint mov_id primary key (mov_id),
constraint mov_id foreign key (dir_id) references director(dir_id)
);
create table movie_cast (
act_id int,
mov_id int,
role varchar(10),
constraint mov_pr primary key(act_id ,mov_id),
constraint mov_act foreign key (act_id) references actor(act_id),
constraint mov_mov foreign key (mov_id) references movies(mov_id)
);
create table rating (
mov_id int ,
rev_stars varchar(25),
constraint rat primary key (mov_id),
constraint rat_mov foreign key (mov_id) references movies(mov_id));
```

```
insert into actor values (301,'anushka','f');
insert into actor values (302,'prabhas','m');
insert into actor values (303,'punith','m');
insert into actor values (304,'jeremy','m');
```

```
INSERT INTO DIRECTOR VALUES (60,"RAJAMOULI", 8751611001);
INSERT INTO DIRECTOR VALUES (61,"HITCHCOCK", 7766138911);
INSERT INTO DIRECTOR VALUES (62,"FARAN", 9986776531);
INSERT INTO DIRECTOR VALUES (63,"STEVEN SPIELBERG", 8989776530);
```

```
INSERT INTO MOVIES VALUES (1001,"BAHUBALI-2", 2017, "TELAGU", 60);
INSERT INTO MOVIES VALUES (1002,"BAHUBALI-1", 2015, "TELAGU", 60);
INSERT INTO MOVIES VALUES (1003,"AKASH", 2008, "KANNADA", 61);
INSERT INTO MOVIES VALUES (1004,"WAR HORSE", 2011, "ENGLISH", 63);
```

```
INSERT INTO MOVIE_CAST VALUES (301, 1002, "HEROINE");
INSERT INTO MOVIE_CAST VALUES (301, 1001, "HEROINE");
INSERT INTO MOVIE_CAST VALUES (303, 1003, "HERO");
INSERT INTO MOVIE_CAST VALUES (303, 1002, "GUEST");
INSERT INTO MOVIE_CAST VALUES (304, 1004, "HERO");
INSERT INTO MOVIE_CAST VALUES (301, 1005, "HEROINE");
INSERT INTO MOVIE_CAST VALUES (303, 1005, "HERO");
```

```
INSERT INTO RATING VALUES (1001, 4);
INSERT INTO RATING VALUES (1002, 2);
INSERT INTO RATING VALUES (1003, 5);
INSERT INTO RATING VALUES (1004, 4);
```

-- i. List the titles of all movies directed by "Hitchcock".

```
select m.mov_title from movies m where dir_id = (select dir_id from director where dir_name = "HITCHCOCK");
```

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

```
66
67
68 -- i. List the titles of all movies directed by "Hitchcock".
69 • select m.mov_title from movies m where dir_id =
70   (select dir_id from director where dir_name = "HITCHCOCK");
71
72
73
```

The Result Grid shows the output:

mov_title
AKASH

The Output pane shows the execution details:

Action Output
# Time Action 184 00:40:01 use movies 185 00:40:08 select m.mov_title from movies m where dir_id = (select dir_id from director where dir_name = "HITCHCOCK"); 1 row(s) returned

-- ii. Find the movie names where one or more actors acted in two or more movies.

```
select mo.mov_title from movies mo where mo.mov_id in (select c.mov_id from movie_cast c where c.act_id in (select m.act_id from movie_cast m group by m.act_id having count(*)>=2));
```

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

```
72
73 -- ii. Find the movie names where one or more actors acted in two or more movies.
74
75 • select mo.mov_title from movies mo where mo.mov_id in
76   (select c.mov_id from movie_cast c where c.act_id in
77     (select m.act_id from movie_cast m group by m.act_id having count(*)>=2));
78
79
```

The Result Grid shows the output:

mov_title
BAHUBALI-2
BAHUBALI-1
AKASH

The Output pane shows the execution details:

Action Output
# Time Action 185 00:40:08 select m.mov_title from movies m where dir_id = (select dir_id from director where dir_name = "HITCHCOCK"); 1 row(s) returned

-- iii. List all actors who acted in a movie before 2000 and also in a movie after 2015 (use JOIN operation).

```
select distinct(m.act_id) from movie_cast m join movie_cast mo on m.act_id=mo.act_id and m.mov_id!=mo.mov_id where m.mov_id in (select mo.mov_id from movies mo where mo.mov_year>=2015) and mo.mov_id in (select mo.mov_id from movies mo where mo.mov_year<2000);
```

-- iv. Find the title of movies and number of stars for each movie that has at least one rating and find the highest number of stars that movie received. Sort the result by movie title.

```
select m.mov_title,r.rev_stars from movies m join rating r where m.mov_id=r.mov_id group by m.mov_title having min(r.rev_stars)>=1 order by r.rev_stars desc ;
```

The screenshot shows the MySQL Workbench interface with the following details:

- File Bar:** File, Edit, View, Query, Database, Server, Tools, Scripting, Help.
- Navigator:** Shows Schemas (airlineflight, banking, banking\_enterprise, books, college, insurance, orderdatabase) and Tables under orderdatabase.
- SQL Editor:** Contains two queries:
  - Query 87: A comment: "-- iv. Find the title of movies and number of stars for each movie that has at least one rating and find the highest number of stars that movie received. Sort the result by movie title."
  - Query 90: The actual query: 

```
select m.mov_title,r.rev_stars from movies m join rating r where m.mov_id=r.mov_id group by m.mov_title having min(r.rev_stars)>=1 order by r.rev_stars desc ;
```
- Result Grid:** Displays the results of the second query:

mov_title	rev_stars
AKASH	5
WAR HORSE	5
BAHUBALI-2	4
BAHUBALI-1	2
- Output Window:** Shows the execution log:

#	Time	Action	Message	Duration / Fetch
188	00:41:39	select distinct(m.act_id) from movie_cast m join movie_cast mo on m.act_id=mo.act_id and m.mov_id!=mo.mov_id where m.mov_id in (select mo.mov_id from movies mo where mo.mov_year>=2015) and mo.mov_id in (select mo.mov_id from movies mo where mo.mov_year<2000);	0 row(s) returned	0.016 sec / 0.000 sec
189	00:42:31	select m.mov_title,r.rev_stars from movies m join rating r where m.mov_id=r.mov_id group by m.mov_title having min(r.rev_stars)>=1 order by r.rev_stars desc ;	4 row(s) returned	0.015 sec / 0.000 sec

-- v. Update rating of all movies directed by "Steven Spielberg' to 5.

```
update rating set rev_stars = 5 where mov_id in (select m.mov_id from movies m where m.dir_id in (select dir_id from director where dir_name="STEVEN SPIELBERG" ));
```

```
select * from rating;
```

The screenshot shows the MySQL Workbench interface. In the top navigation bar, the database 'Yashaswini' is selected. The main area contains a SQL editor window with the following code:

```

94
95 -- v. Update rating of all movies directed by "Steven Spielberg" to 5.
96
97 update rating set rev_stars = 5 where mov_id in
98 (select m.mov_id from movies m where m.dir_id in
99 (select dir_id from director where dir_name="STEVEN SPIELBERG "));
100
101 select * from rating;

```

The results grid shows the following data:

mov_id	rev_stars
1001	4
1002	2
1003	5
1004	5
*	NULL

The output pane shows the execution log:

Action	Time	Message	Duration / Fetch
select * from rating LIMIT 0, 10	00:44:21	4 row(s) returned	0.000 sec / 0.000 sec
select * from rating LIMIT 0, 10	00:44:38	4 row(s) returned	0.000 sec / 0.000 sec

## PROGRAM 10:COLLEGE DATABASE

Consider the schema for College Database:

STUDENT(USN, SName, Address, Phone, Gender)

SEMSEC(SSID, Sem, Sec) CLASS(USN, SSID)

SUBJECT(Subcode, Title, Sem, Credits)

IAMARKS(USN, Subcode, SSID, Test1, Test2, Test3, FinalIA) Write SQL queries to

i. List all the student details studying in fourth semester 'C' section.

ii. Compute the total number of male and female students in each semester and in each section.

iii. Create a view of Test1 marks of student USN '1BI15CS101' in all subjects.

iv. Calculate the FinalIA (average of best two test marks) and update the corresponding table for all students.

v. Categorize students based on the following criterion: If FinalIA = 17 to 20 then CAT = 'Outstanding' If FinalIA = 12 to 16 then CAT = 'Average' If FinalIA < 12 then CAT = 'Weak' Give these details only for 8th semester A, B, and C section students.

```

create database college;
use college;
create table student (
usn varchar(10),
sname varchar(10),

```

```

address varchar (30),
phone varchar(10),
gender varchar(1),
constraint stu_usn primary key (usn)
);
create table semsec (
    ssid varchar(5),
    sem varchar(2),
    sec varchar(1),
    constraint sem_ssid primary key (ssid)
);
create table class(
    usn varchar(10),
    ssid varchar(5),
    constraint class_usn primary key (usn),
    constraint class_usn foreign key (usn) references student (usn) on delete cascade on
update cascade,
    constraint class_ssid foreign key (ssid) references semsec(ssid) on delete cascade on
update cascade
);
create table subjects(
    subcode varchar(10),
    title varchar (20),
    sem int,
    credits int,
    constraint sub_sub primary key (subcode)
);
create table iamarks(
    usn varchar(10),
    subcode varchar(10),
    ssid varchar(5),
    test1 int,
    test2 int,
    test3 int,
    finalia int,
    constraint ia primary key (usn,subcode,ssid),
    constraint ia_usn foreign key (usn) references student(usn) on delete cascade on
update cascade ,
    constraint ia_subcode foreign key (subcode) references subjects(subcode) on delete
cascade on update cascade,
    constraint ia_ssid foreign key (ssid) references semsec(ssid) on delete cascade on
update cascade
);

```

```
INSERT INTO STUDENT VALUES ('1BM19CS020','AKSHAY','BELAGAVI',
8877881122,'M');
INSERT INTO STUDENT VALUES ('1BM19CS062','SANDHYA','BENGALURU',
7722829912,'F');
INSERT INTO STUDENT VALUES ('1BM19CS091','TEESHA','BENGALURU',
7712312312,'F');
INSERT INTO STUDENT VALUES ('1BM19CS066','SUPRIYA','MANGALURU',
8877881122,'F');
INSERT INTO STUDENT VALUES ('1BM20CS010','ABHAY','BENGALURU',
9900211201,'M');
INSERT INTO STUDENT VALUES ('1BM20CS032','BHASKAR','BENGALURU',
9923211099,'M');
INSERT INTO STUDENT VALUES ('1BM20CS025','ASMI','BENGALURU',
7894737377,'F');
INSERT INTO STUDENT VALUES ('1BM20CS011','AJAY','TUMKUR', 9845091941,'M');
```

```
INSERT INTO SEMSEC VALUES ('CSE4A', 4,'A');
INSERT INTO SEMSEC VALUES ('CSE4B', 4,'B');
INSERT INTO SEMSEC VALUES ('CSE4C', 4,'C');
INSERT INTO SEMSEC VALUES ('CSE6A', 6,'A');
INSERT INTO SEMSEC VALUES ('CSE6B', 6,'B');
INSERT INTO SEMSEC VALUES ('CSE6C', 6,'C');
INSERT INTO SEMSEC VALUES ('CSE8A', 8,'A');
INSERT INTO SEMSEC VALUES ('CSE8B', 8,'B');
INSERT INTO SEMSEC VALUES ('CSE8C', 8,'C');
```

```
INSERT INTO CLASS VALUES ('1BM19CS020','CSE8A');
INSERT INTO CLASS VALUES ('1BM19CS062','CSE4A');
INSERT INTO CLASS VALUES ('1BM19CS066','CSE8B');
INSERT INTO CLASS VALUES ('1BM19CS091','CSE8C');
INSERT INTO CLASS VALUES ('1BM20CS010','CSE4A');
INSERT INTO CLASS VALUES ('1BM20CS025','CSE4B');
INSERT INTO CLASS VALUES ('1BM20CS032','CSE4C');
```

```
INSERT INTO SUBJECTS VALUES ('10CS81','ACA', 8, 4);
INSERT INTO SUBJECTS VALUES ('10CS82','SSM', 8, 4);
INSERT INTO SUBJECTS VALUES ('10CS83','NM', 8, 4);
```

```
INSERT INTO IAMARKS (USN, SUBCODE, SSID, TEST1, TEST2, TEST3) VALUES
('1BM19CS091','10CS81','CSE8C', 15, 16, 18);
INSERT INTO IAMARKS (USN, SUBCODE, SSID, TEST1, TEST2, TEST3) VALUES
('1BM19CS091','10CS82','CSE8C', 12, 19, 14);
```

```
INSERT INTO IAMARKS (USN, SUBCODE, SSID, TEST1, TEST2, TEST3) VALUES  
('1BM19CS091','10CS83','CSE8C', 19, 15, 20);
```

-- i. List all the student details studying in fourth semester 'C' section.

```
select * from student s where s.usn  
in (select c.usn from class c where c ssid in  
(select ssid from semsec where sem='4' and sec='C'));
```

The screenshot shows the MySQL Workbench interface with the following details:

- Navigator:** Shows the database schema with a tree view of tables, views, stored procedures, and functions under the 'college' schema.
- SQL Editor:** Contains the SQL query for listing students in the 'C' section of the fourth semester. The output shows one row returned:

usn	sname	address	phone	gender
1BM20CS032	BHASKAR	BENGALURU	9923211099	M

- Output Window:** Displays the execution log with two entries:

Action	Time	Message	Duration / Fetch
use college	00:56:57	0 row(s) affected	0.000 sec
select * from student s where s.usn in (select c.usn from class c where c ssid in ...)	00:57:06	1 row(s) returned	0.016 sec / 0.000 sec

-- ii. Compute the total number of male and female students in each semester and in each section

```
select se.sem,se.sec,gender,count(gender) as 'number of students' from class c natural  
join student s natural join semsec se group by sem,sec,gender;
```

The screenshot shows the MySQL Workbench interface. In the top navigation bar, the user is connected to 'Yashaswini'. The main area displays a query editor with the following SQL code:

```

86
87
88 -- ii. Compute the total number of male and female students in each semester and in each section
89 select se.sem,se.sec,gender,count(gender) as
90 'number of students' from class c natural join student s
91 natural join semsec se group by sem,sec,gender;
92
93

```

The results of this query are shown in a 'Result Grid' table:

sem	sec	gender	number of students
4	A	F	1
4	A	M	1
4	B	F	1
4	C	M	1
8	A	M	1
8	B	F	1
8	C	F	1

Below the results, the 'Output' pane shows two log entries:

#	Time	Action	Message	Duration / Fetch
194	00:57:06	select * from student s where s.usn in (select c.usn from class c where c.usn in ...)	1 row(s) returned	0.016 sec / 0.000 sec
195	01:01:21	select se.sem,se.sec,gender,count(gender) as	'number of students' from class c ...	7 row(s) returned

-- iii. Create a view of Test1 marks of student USN '1BI15CS101' in all subjects.

create view marks(test1) as select i.test1 from iamarks i where i.usn='1BM19CS091';

-- iv.calculate finalia marks and update the respective column for all student;

-- select ia.usn,test1,test2,test3,(case

-- when test1>test3 and test2>test3 then (test1+test2)/2

-- when test2>test1 and test3>test1 then (test2+test3)/2

-- else (test1+test3)/2 end) as finalia from iamarks ia;

update iamarks set finalia= case when test1>test2 and test2>test3 then (test1+test2)/2

when test2>test3 and test3>test1 then (test2+test3)/2

else (test1+test3)/2 end;

select \* from iamarks;

The screenshot shows the MySQL Workbench interface. In the top navigation bar, the user is connected to 'Yashaswini'. The 'Query' tab is selected. Below the toolbar, there are tabs for 'labprogram7', 'SQL File 6\*', 'labprogram6\*', 'labprogram9\*', 'labprogram8\*', 'SQL File 7\*' (which is currently active), 'ENROLL', and 'MOVIE'. The 'Navigator' pane on the left lists various schemas: 'airlineflight', 'banking', 'banking\_enterprise', 'books', 'college' (selected), 'Views', 'Stored Procedures', 'Functions', and 'insurance'. The 'Information' pane below it shows the 'Schema: college'. The main area contains a SQL editor with the following code:

```

105      --      else (test1+test3)/2 end) as finalia from iamarks ia;
106
107  update iamarks set finalia= case when test1>test2 and test2>test3 then (test1+test2)/2
108      when test2>test3 and test3>test1 then (test2+test3)/2
109      else (test1+test3)/2 end;
110
111  select * from iamarks;
112

```

Below the SQL editor is a 'Result Grid' showing the data from the 'iamarks' table:

USN	subcode	ssid	test1	test2	test3	finalia
1BM19CS091	10CS81	CSEBC	15	16	18	NULL
1BM19CS091	10CS82	CSEBC	12	19	14	NULL
1BM19CS091	10CS83	CSEBC	19	15	20	NULL
NULL	NULL	NULL	NULL	NULL	NULL	NULL

The 'Output' pane at the bottom shows the execution log:

- Action Output: # 204 01:03:53 update iamarks set finalia= case when test1>test2 and test2>test3 then (test1+test2)/2 ... Duration / Fetch 0.000 sec / 0.000 sec
- Action Output: # 205 01:03:56 select \* from iamarks LIMIT 0, 10 3 row(s) returned Duration / Fetch 0.000 sec / 0.000 sec

-- v. Categorize students based on the following criterion:

- If FinalIA = 17 to 20 then CAT = 'Outstanding'
- If FinalIA = 12 to 16 then CAT = 'Average'
- If FinalIA < 12 then CAT = 'Weak'
- Give these details only for 8th semester A, B, and C section students.

```

select s.usn,s.sname,s.address,s.phone,s.gender,
(case
when ia.finalia between 17 and 20 then 'outstanding'
when ia.finalia between 12 and 16 then 'average'
else 'weak'
end) as cat from iamarks ia,student s where s.usn=ia.usn and ia.ssid in (select ss.ssid
from semsec ss where sem='8');
select test1,test2,test3,finalia,subcode from iamarks group by subcode;
select s.usn,s.sname,s.address,s.phone,s.gender from student s,iamarks i where i.finalia
>45;
create view subject as (select test1,test2,test3,finalia,subcode from iamarks group by subcode);
select avg(test1),avg(test2),avg(test3) from iamarks group by subcode;

```

MySQL Workbench

Yashaswini

File Edit View Query Database Server Tools Scripting Help

Navigator: labprogram7 SQL File 6\* labprogram6\* labprogram9\* labprogram8\* SQL File 7\* ENROLL MOVIE

Schemas: college

```

121      when ia.finalia between 17 and 20 then 'outstanding'
122      when ia.finalia between 12 and 16 then 'average'
123      else 'weak'
124  end) as cat from iamarks ia,student s where s.usn=ia.usn and ia.ssid in (select ss ssid from semsec ss
125  select test1,test2,test3,finalia,subcode from iamarks group by subcode;
126  select s.usn,s.sname,s.address,s.phone,s.gender from student s,iamarks i where i.finalia >45;
127  create view subject as (select test1,test2,test3,finalia,subcode from iamarks group by subcode);
128  select avg(test1),avg(test2),avg(test3) from iamarks group by subcode;

```

Result Grid:

	avg(test1)	avg(test2)	avg(test3)
1	15.0000	16.0000	18.0000
2	12.0000	19.0000	14.0000
3	19.0000	15.0000	20.0000

Result 11 x

Output:

Action Output:

#	Time	Action	Message	Duration / Fetch
207	01:04:32	create view subject as (select test1,test2,test3,finalia,subcode from iamarks group by subcode);	0 row(s) affected	0.031 sec
208	01:04:35	select avg(test1),avg(test2),avg(test3) from iamarks group by subcode LIMIT 0, 10	3 row(s) returned	0.000 sec / 0.000 sec

Object Info Session

MySQL Workbench

Yashaswini

File Edit View Query Database Server Tools Scripting Help

Navigator: labprogram7 SQL File 6\* labprogram6\* labprogram9\* labprogram8\* SQL File 7\* ENROLL MOVIE

Schemas: college

```

121      when ia.finalia between 17 and 20 then 'outstanding'
122      Execute the statement under the keyboard cursor 'average'
123      when ia.finalia between 12 and 16 then 'average'
124      else 'weak'
125  end) as cat from iamarks ia,student s where s.usn=ia.usn and ia.ssid in (select ss ssid from semsec ss
126  select test1,test2,test3,finalia,subcode from iamarks group by subcode;
127  select s.usn,s.sname,s.address,s.phone,s.gender from student s,iamarks i where i.finalia >45;
128  create view subject as (select test1,test2,test3,finalia,subcode from iamarks group by subcode);
129  select avg(test1),avg(test2),avg(test3) from iamarks group by subcode;

```

Result Grid:

usn	sname	address	phone	gender	cat
IBM19CS091	TEESHA	BENGALURU	7712312312	F	weak
IBM19CS091	TEESHA	BENGALURU	7712312312	F	weak
IBM19CS091	TEESHA	BENGALURU	7712312312	F	weak

Result 12 x

Output:

Action Output:

#	Time	Action	Message	Duration / Fetch
208	01:04:35	select avg(test1),avg(test2),avg(test3) from iamarks group by subcode LIMIT 0, 10	3 row(s) returned	0.000 sec / 0.000 sec
209	01:05:06	select s.usn,s.sname,s.address,s.phone,s.gender,(case when ia.finalia between 17 and 20 then 'outstanding' when ia.finalia between 12 and 16 then 'average' else 'weak' end) as cat from iamarks ia,student s where s.usn=ia.usn and ia.ssid in (select ss ssid from semsec ss select test1,test2,test3,finalia,subcode from iamarks group by subcode);	3 row(s) returned	0.000 sec / 0.000 sec

Object Info Session

**1BM19CS216**

**THANK YOU**