

Simon Abhijet Biswas

CSE370: Database Systems

Brac University

Lab 01: Setting Up and Connecting to the MySQL Server

Lab 02: SQL Update, Delete & Basic Select Queries

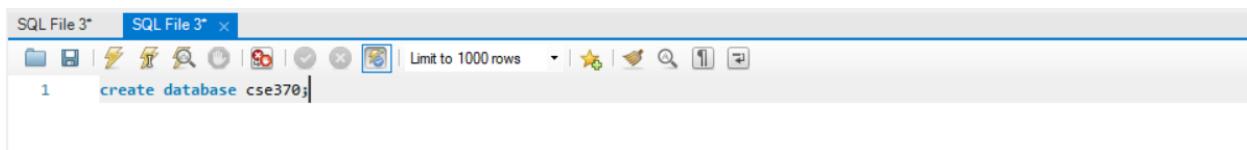
Lab 03: SQL Subqueries & Aggregate Functions

Lab 05: Introduction to Bank DB, SQL Joins and Constraints

Lab 1: Create and Insert

- **Create a database cse370 from Lab Grades**

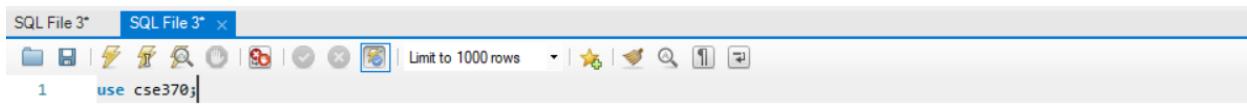
```
create database cse370
```



A screenshot of the MySQL Workbench interface. The title bar says "SQL File 3*". The main pane shows the SQL command: "1 create database cse370;". Below the title bar are various icons for database management.

- **Use cse370**

```
use cse370
```



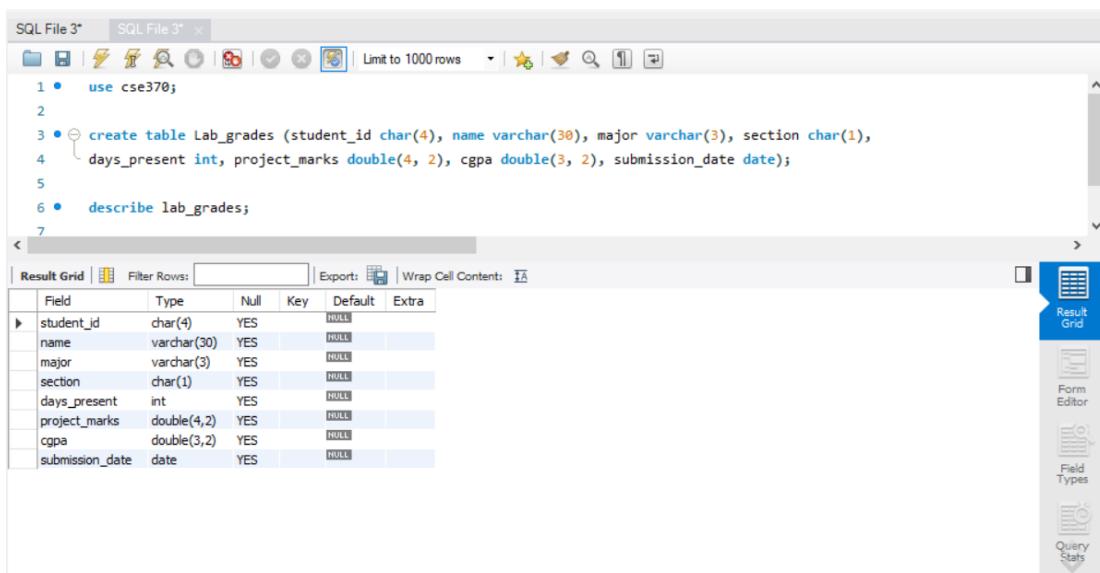
A screenshot of the MySQL Workbench interface. The title bar says "SQL File 3*". The main pane shows the SQL command: "1 use cse370;". Below the title bar are various icons for database management.

- **Create a Lab Grades table and Describe it.**

```
use cse370;
```

```
create table Lab_grades (student_id char(4), name varchar(30), major varchar(3), section char(1), days_present int, project_marks double(4, 2), cgpa double(3, 2), submission_date date);
```

```
describe lab_grades;
```



A screenshot of the MySQL Workbench interface. The title bar says "SQL File 3*". The main pane shows the following sequence of commands:
1 • use cse370;
2
3 • create table Lab_grades (student_id char(4), name varchar(30), major varchar(3), section char(1), days_present int, project_marks double(4, 2), cgpa double(3, 2), submission_date date);
4
5
6 • describe lab_grades;
7

Below the SQL pane is a "Result Grid" table showing the structure of the Lab_grades table:

Field	Type	Null	Key	Default	Extra
student_id	char(4)	YES		HULL	
name	varchar(30)	YES		HULL	
major	varchar(3)	YES		HULL	
section	char(1)	YES		HULL	
days_present	int	YES		HULL	
project_marks	double(4,2)	YES		HULL	
cgpa	double(3,2)	YES		HULL	
submission_date	date	YES		HULL	

The right side of the interface has a sidebar with icons for Result Grid, Form Editor, Field Types, and Query Stats.

- **Show Databases and Tables**

```
show database;
```

The screenshot shows the MySQL Workbench interface with two tabs: 'SQL File 3*' and 'SQL File 3*'. The SQL pane contains the command:

```
1 • show databases;
```

The results pane, titled 'Result Grid', displays a list of databases:

Database
cse370
dinodb
djangorestdb
exam
full-stack-e-commerce

On the right side of the results pane, there are icons for 'Result Grid' (selected) and 'Form Editor'.

```
use cse370;
```

```
show tables;
```

The screenshot shows the MySQL Workbench interface with two tabs: 'SQL File 3*' and 'SQL File 3*'. The SQL pane contains the commands:

```
1 • use cse370;
2
3 • show tables;
4
5
```

The results pane, titled 'Result Grid', displays a list of tables:

Tables_in_cse370
lab_grades

On the right side of the results pane, there are icons for 'Result Grid' (selected) and 'Form Editor'.

- **Insert data into the Lab Grades table**

```
use cse370;
```

```
insert into Lab_grades (student_id, name, major, section, days_present, project_marks, cgpa, submission_date) values ('s001', 'Abir', 'CS', '1', 10, 18.50, 3.91, '2018-09-15');
```

```
insert into Lab_grades (student_id, name, major, section, days_present, project_marks, cgpa, submission_date) values ('s002', 'Nafis', 'CSE', '1', 12, 20, 3.86, '2018-08-15');
```

```
insert into Lab_grades (student_id, name, major, section, days_present, project_marks, cgpa, submission_date) values ('s019', 'Naima', 'CSE', '2', 12, 20, 3.7, '2018-08-14');
```

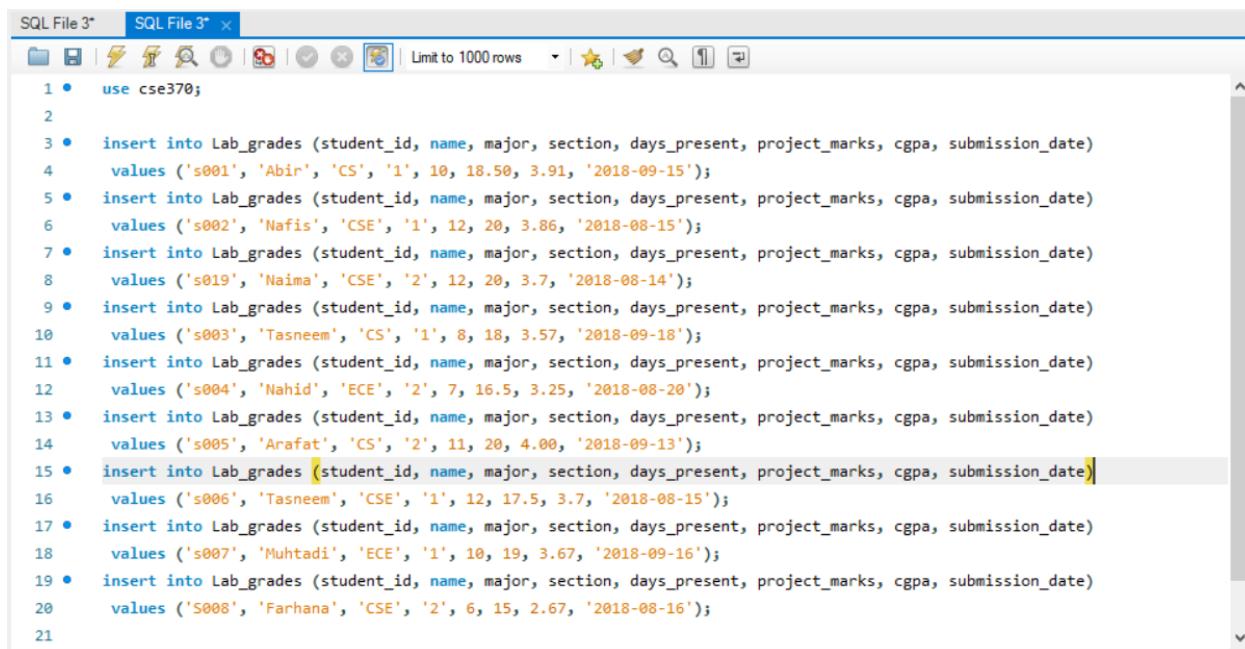
```
insert into Lab_grades (student_id, name, major, section, days_present, project_marks, cgpa, submission_date) values ('s003', 'Tasneem', 'CS', '1', 8, 18, 3.57, '2018-09-18');
insert into Lab_grades (student_id, name, major, section, days_present, project_marks, cgpa, submission_date) values ('s004', 'Nahid', 'ECE', '2', 7, 16.5, 3.25, '2018-08-20');
```

```
insert into Lab_grades (student_id, name, major, section, days_present, project_marks, cgpa, submission_date) values ('s005', 'Arafat', 'CS', '2', 11, 20, 4.00, '2018-09-13');
```

```
insert into Lab_grades (student_id, name, major, section, days_present, project_marks, cgpa, submission_date) values ('s006', 'Tasneem', 'CSE', '1', 12, 17.5, 3.7, '2018-08-15');
```

```
insert into Lab_grades (student_id, name, major, section, days_present, project_marks, cgpa, submission_date) values ('s007', 'Muhtadi', 'ECE', '1', 10, 19, 3.67, '2018-09-16');
```

```
insert into Lab_grades (student_id, name, major, section, days_present, project_marks, cgpa, submission_date) values ('S008', 'Farhana', 'CSE', '2', 6, 15, 2.67, '2018-08-16');
```



```
SQL File 3* SQL File 3* 
1 • use cse370;
2
3 • insert into Lab_grades (student_id, name, major, section, days_present, project_marks, cgpa, submission_date)
4   values ('s001', 'Abir', 'CS', '1', 10, 18.50, 3.91, '2018-09-15');
5 • insert into Lab_grades (student_id, name, major, section, days_present, project_marks, cgpa, submission_date)
6   values ('s002', 'Nafis', 'CSE', '1', 12, 20, 3.86, '2018-08-15');
7 • insert into Lab_grades (student_id, name, major, section, days_present, project_marks, cgpa, submission_date)
8   values ('s019', 'Naima', 'CSE', '2', 12, 20, 3.7, '2018-08-14');
9 • insert into Lab_grades (student_id, name, major, section, days_present, project_marks, cgpa, submission_date)
10  values ('s003', 'Tasneem', 'CS', '1', 8, 18, 3.57, '2018-09-18');
11 • insert into Lab_grades (student_id, name, major, section, days_present, project_marks, cgpa, submission_date)
12  values ('s004', 'Nahid', 'ECE', '2', 7, 16.5, 3.25, '2018-08-20');
13 • insert into Lab_grades (student_id, name, major, section, days_present, project_marks, cgpa, submission_date)
14  values ('s005', 'Arafat', 'CS', '2', 11, 20, 4.00, '2018-09-13');
15 • insert into Lab_grades [student_id, name, major, section, days_present, project_marks, cgpa, submission_date]
16  values ('s006', 'Tasneem', 'CSE', '1', 12, 17.5, 3.7, '2018-08-15');
17 • insert into Lab_grades (student_id, name, major, section, days_present, project_marks, cgpa, submission_date)
18  values ('s007', 'Muhtadi', 'ECE', '1', 10, 19, 3.67, '2018-09-16');
19 • insert into Lab_grades (student_id, name, major, section, days_present, project_marks, cgpa, submission_date)
20  values ('S008', 'Farhana', 'CSE', '2', 6, 15, 2.67, '2018-08-16');
21
```

- **Show the Lab Grades Table**

```
select * from Lab_grades;
```

The screenshot shows the MySQL Workbench interface. At the top, there are two tabs: "SQL File 3*" and "Result Grid". Below the tabs is a toolbar with various icons. The main area contains the following SQL code:

```
1 • use cse370;
2
3 • select * from lab_grades;
4
5
```

Below the code is the "Result Grid" section, which displays the data from the "lab_grades" table. The columns are: student_id, name, major, section, days_present, project_marks, cgpa, and submission_date. The data rows are:

student_id	name	major	section	days_present	project_marks	cgpa	submission_date
s001	Abir	CS	1	10	18.50	3.91	2018-09-15
s002	Nafis	CSE	1	12	20.00	3.86	2018-08-15
s019	Naima	CSE	2	12	20.00	3.70	2018-08-14
s003	Tasneem	CS	1	8	18.00	3.57	2018-09-18
s004	Nahid	ECE	2	7	16.50	3.25	2018-08-20
s005	Arafat	CS	2	11	20.00	4.00	2018-09-13
s006	Tasneem	CSE	1	12	17.50	3.70	2018-08-15
s007	Muhtadi	ECE	1	10	19.00	3.67	2018-09-16
S008	Farhana	CSE	2	6	15.00	2.67	2018-08-16

On the right side of the result grid, there is a vertical toolbar with three buttons: "Result Grid" (selected), "Form Editor", and "Field Types".

The Lab Grades Table:

This screenshot shows the same MySQL Workbench interface as the previous one, but the "Result Grid" section is now empty. It still displays the same header row and the same data rows as the previous screenshot, indicating that the query has been successfully executed.

student_id	name	major	section	days_present	project_marks	cgpa	submission_date
s001	Abir	CS	1	10	18.50	3.91	2018-09-15
s002	Nafis	CSE	1	12	20.00	3.86	2018-08-15
s019	Naima	CSE	2	12	20.00	3.70	2018-08-14
s003	Tasneem	CS	1	8	18.00	3.57	2018-09-18
s004	Nahid	ECE	2	7	16.50	3.25	2018-08-20
s005	Arafat	CS	2	11	20.00	4.00	2018-09-13
s006	Tasneem	CSE	1	12	17.50	3.70	2018-08-15
s007	Muhtadi	ECE	1	10	19.00	3.67	2018-09-16
S008	Farhana	CSE	2	6	15.00	2.67	2018-08-16

Lab 02: SQL Update, Delete & Basic Select Queries

- Add Column Project Title to the Lab Grades table

```
use cse370;
```

```
alter table lab_grades add project_title char(30);
```

```
describe lab_grades;
```

The screenshot shows the MySQL Workbench interface with a SQL editor window containing the following code:

```
SQL File 3* SQL File 3*  
1 • use cse370;  
2  
3 • alter table lab_grades add project_title char(30);  
4  
5 • describe lab_grades;
```

Below the code, the 'Result Grid' pane displays the structure of the 'lab_grades' table, which now includes the 'project_title' column:

Field	Type	Null	Key	Default	Extra
student_id	char(4)	YES		NULL	
name	varchar(30)	YES		NULL	
major	varchar(3)	YES		NULL	
section	char(1)	YES		NULL	
days_present	int	YES		NULL	
project_marks	double(4,2)	YES		NULL	
cpga	double(3,2)	YES		NULL	
submission_date	date	YES		NULL	
project_title	char(30)	YES		NULL	

- Change the Data type of Project Title column from char to varchar

```
use cse370;
```

```
alter table lab_grades modify column project_title varchar(50);
```

```
describe lab_grades;
```

The screenshot shows the MySQL Workbench interface with a SQL editor window containing the following code:

```
SQL File 3* SQL File 3*  
1 • use cse370;  
2  
3 • alter table lab_grades modify column project_title varchar(50);  
4  
5 • describe lab_grades;
```

Below the code, the 'Result Grid' pane displays the structure of the 'lab_grades' table, showing that the 'project_title' column has been successfully modified to a 'varchar(50)' type:

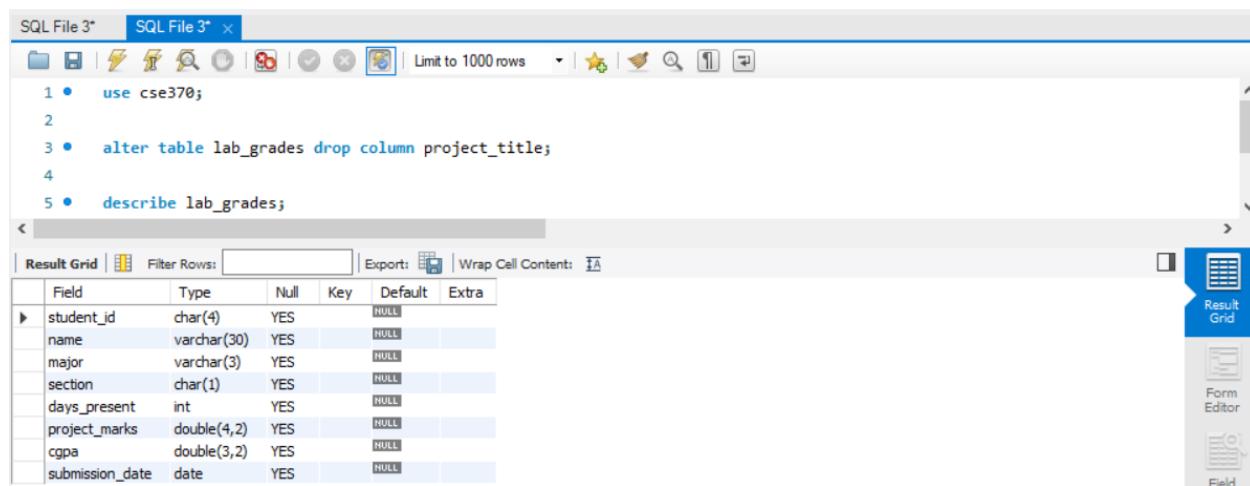
Field	Type	Null	Key	Default	Extra
student_id	char(4)	YES		NULL	
name	varchar(30)	YES		NULL	
major	varchar(3)	YES		NULL	
section	char(1)	YES		NULL	
days_present	int	YES		NULL	
project_marks	double(4,2)	YES		NULL	
cpga	double(3,2)	YES		NULL	
submission_date	date	YES		NULL	
project_title	varchar(50)	YES		NULL	

- Delete Project Title column from table

```
use cse370;
```

```
alter table lab_grades drop column project_title;
```

```
describe lab_grades;
```



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

```
1 • use cse370;
2
3 • alter table lab_grades drop column project_title;
4
5 • describe lab_grades;
```

The results grid displays the structure of the 'lab_grades' table:

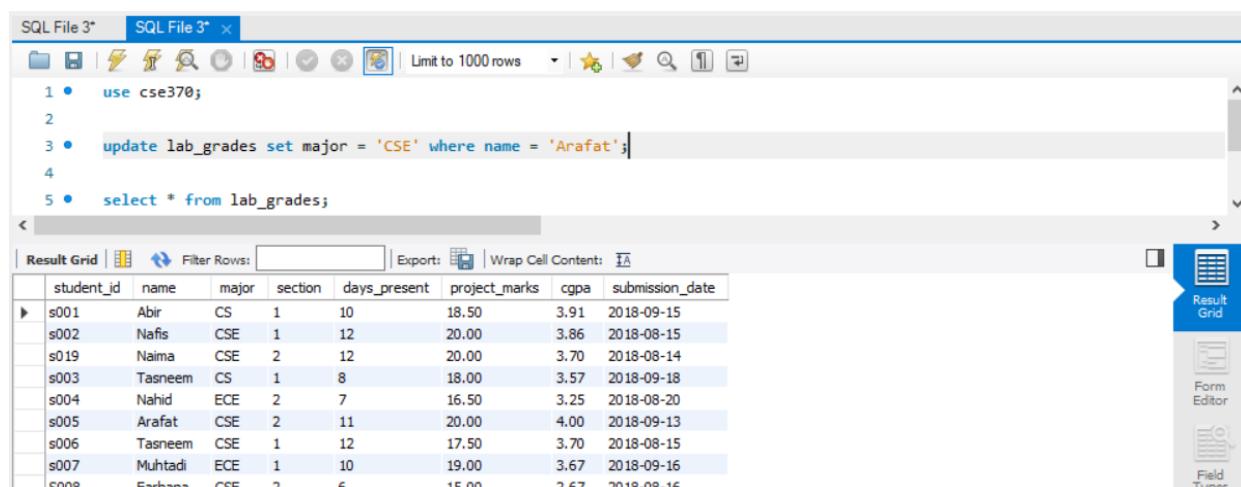
Field	Type	Null	Key	Default	Extra
student_id	char(4)	YES		NULL	
name	varchar(30)	YES		NULL	
major	varchar(3)	YES		NULL	
section	char(1)	YES		NULL	
days_present	int	YES		NULL	
project_marks	double(4,2)	YES		NULL	
cgpa	double(3,2)	YES		NULL	
submission_date	date	YES		NULL	

- Change Arafat's Major from CS to CSE

```
use cse370;
```

```
update lab_grades set major = 'CSE' where name = 'Arafat';
```

```
select * from lab_grades;
```



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

```
1 • use cse370;
2
3 • update lab_grades set major = 'CSE' where name = 'Arafat';
4
5 • select * from lab_grades;
```

The results grid displays the updated data in the 'lab_grades' table:

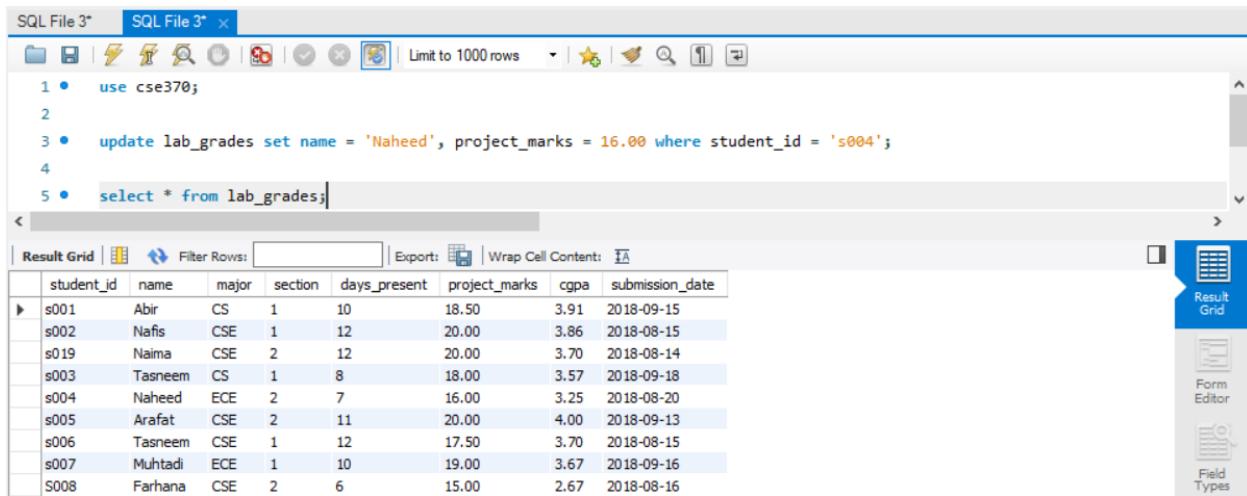
student_id	name	major	section	days_present	project_marks	cgpa	submission_date
s001	Abir	CS	1	10	18.50	3.91	2018-09-15
s002	Nafis	CSE	1	12	20.00	3.86	2018-08-15
s019	Naima	CSE	2	12	20.00	3.70	2018-08-14
s003	Tasneem	CS	1	8	18.00	3.57	2018-09-18
s004	Nahid	ECE	2	7	16.50	3.25	2018-08-20
s005	Arafat	CSE	2	11	20.00	4.00	2018-09-13
s006	Tasneem	CSE	1	12	17.50	3.70	2018-08-15
s007	Muhtadi	ECE	1	10	19.00	3.67	2018-09-16
S008	Farhana	CSE	2	6	15.00	2.67	2018-08-16

- **Change Nahid name to Naheed and his project mark is 16**

```
use cse370;
```

```
update lab_grades set name = 'Naheed', project_marks = 16.00 where student_id = 's004';
```

```
select * from lab_grades;
```



The screenshot shows the MySQL Workbench interface with two tabs open: "SQL File 3*" and "Result Grid". The SQL tab contains the following code:

```

1 • use cse370;
2
3 • update lab_grades set name = 'Naheed', project_marks = 16.00 where student_id = 's004';
4
5 • select * from lab_grades;

```

The Result Grid shows the following data:

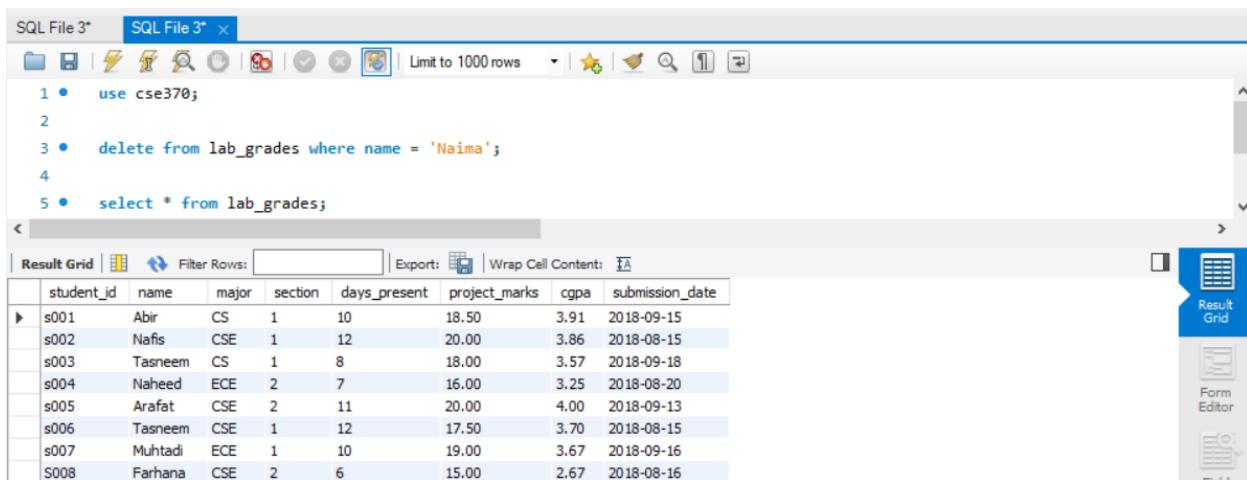
student_id	name	major	section	days_present	project_marks	cgpa	submission_date
s001	Abir	CS	1	10	18.50	3.91	2018-09-15
s002	Nafis	CSE	1	12	20.00	3.86	2018-08-15
s019	Naima	CSE	2	12	20.00	3.70	2018-08-14
s003	Tasneem	CS	1	8	18.00	3.57	2018-09-18
s004	Naheed	ECE	2	7	16.00	3.25	2018-08-20
s005	Arafat	CSE	2	11	20.00	4.00	2018-09-13
s006	Tasneem	CSE	1	12	17.50	3.70	2018-08-15
s007	Muhtadi	ECE	1	10	19.00	3.67	2018-09-16
S008	Farhana	CSE	2	6	15.00	2.67	2018-08-16

- **Naima has been dropped out so delete her data**

```
use cse370;
```

```
delete from lab_grades where name = 'Naima';
```

```
select * from lab_grades;
```



The screenshot shows the MySQL Workbench interface with two tabs open: "SQL File 3*" and "Result Grid". The SQL tab contains the following code:

```

1 • use cse370;
2
3 • delete from lab_grades where name = 'Naima';
4
5 • select * from lab_grades;

```

The Result Grid shows the following data:

student_id	name	major	section	days_present	project_marks	cgpa	submission_date
s001	Abir	CS	1	10	18.50	3.91	2018-09-15
s002	Nafis	CSE	1	12	20.00	3.86	2018-08-15
s003	Tasneem	CS	1	8	18.00	3.57	2018-09-18
s004	Naheed	ECE	2	7	16.00	3.25	2018-08-20
s005	Arafat	CSE	2	11	20.00	4.00	2018-09-13
s006	Tasneem	CSE	1	12	17.50	3.70	2018-08-15
s007	Muhtadi	ECE	1	10	19.00	3.67	2018-09-16
S008	Farhana	CSE	2	6	15.00	2.67	2018-08-16

- **Retrieve only Student Id, Name and Project Marks**

```
use cse370;
```

```
select student_id, name, project_marks from lab_grades;
```

The screenshot shows the MySQL Workbench interface with two tabs open: "SQL File 3*" and "Result Grid". The SQL tab contains the following code:

```
1 • use cse370;
2
3 • select student_id, name, project_marks from lab_grades;
4
5
```

The Result Grid tab displays the following data:

student_id	name	project_marks
s001	Abir	18.50
s002	Nafis	20.00
s003	Tasneem	18.00
s004	Naheed	16.00
s005	Arafat	20.00
s006	Tasneem	17.50
s007	Muhtadi	19.00
s008	Farhana	15.00

- **Retrieve Student Name and Total Marks of Students out of 25 (Project + Attendance)**

```
use cse370;
```

```
select name, (project_marks + days_present) * 5 / 12 as total_marks from lab_grades;
```

The screenshot shows the MySQL Workbench interface with two tabs open: "SQL File 3*" and "Result Grid". The SQL tab contains the following code:

```
1 • use cse370;
2
3 • select name, (project_marks + days_present) * 5 / 12 as total_marks from lab_grades;
4
5
```

The Result Grid tab displays the following data:

name	total_marks
Abir	11.875000
Nafis	13.333333
Tasneem	10.833333
Naheed	9.583333
Arafat	12.916667
Tasneem	12.291667
Muhtadi	12.083333
Farhana	8.750000

- **Select Student Name in Upper and Lower Case**

```
use cse370;
```

```
select upper(name) as uppercase_names, lower(name) as lowercase_names from lab_grades;
```

The screenshot shows the MySQL Workbench interface with two tabs: "SQL File 3*" and "SQL File 3*". The SQL editor contains the following code:

```
1 • use cse370;
2
3 • select upper(name) as uppercase_names, lower(name) as lowercase_names from lab_grades;
4
5
```

The results are displayed in a "Result Grid" table:

	uppercase_names	lowercase_names
▶	ABIR	abir
	NAFIS	nafis
	TASNEEM	tasneem
	NAHEED	naheed
	ARAFAT	arafat
	TASNEEM	tasneem
	MUHTADI	muhtadi
	FARHANA	farhana

- **Select Distinct Majors**

```
use cse370;
```

```
select distinct(major) as different_majors from lab_grades;
```

The screenshot shows the MySQL Workbench interface with two tabs: "SQL File 3*" and "SQL File 3*". The SQL editor contains the following code:

```
1 • use cse370;
2
3 • select distinct(major) as different_majors from lab_grades;
4
5
```

The results are displayed in a "Result Grid" table:

	different_majors
▶	CS
	CSE
	ECE

- **View all the details sorted by Name**

```
use cse370;
```

```
select * from lab_grades order by name;
```

student_id	name	major	section	days_present	project_marks	cgpa	submission_date
s001	Abir	CS	1	10	18.50	3.91	2018-09-15
s005	Arafat	CSE	2	11	20.00	4.00	2018-09-13
S008	Farhana	CSE	2	6	15.00	2.67	2018-08-16
s007	Muhtadi	ECE	1	10	19.00	3.67	2018-09-16
s002	Nafis	CSE	1	12	20.00	3.86	2018-08-15
s004	Naheed	ECE	2	7	16.00	3.25	2018-08-20
s003	Tasneem	CS	1	8	18.00	3.57	2018-09-18
s006	Tasneem	CSE	1	12	17.50	3.70	2018-08-15

- **Sort all details according to Name and then by Submission Date. There are two students named Tasneem**

```
use cse370;
```

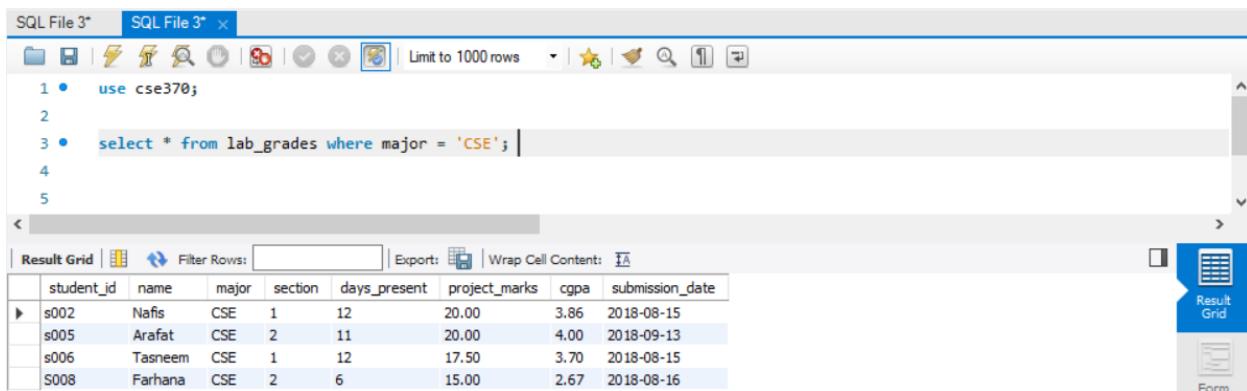
```
select * from lab_grades order by name desc, submission_date asc;
```

student_id	name	major	section	days_present	project_marks	cgpa	submission_date
s006	Tasneem	CSE	1	12	17.50	3.70	2018-08-15
s003	Tasneem	CS	1	8	18.00	3.57	2018-09-18
s004	Naheed	ECE	2	7	16.00	3.25	2018-08-20
s002	Nafis	CSE	1	12	20.00	3.86	2018-08-15
► s007	Muhtadi	ECE	1	10	19.00	3.67	2018-09-16
S008	Farhana	CSE	2	6	15.00	2.67	2018-08-16
s005	Arafat	CSE	2	11	20.00	4.00	2018-09-13
s001	Abir	CS	1	10	18.50	3.91	2018-09-15

- **View the Name and Project marks for only CSE students**

```
use cse370;
```

```
select * from lab_grades where major = 'CSE';
```



The screenshot shows the MySQL Workbench interface with two tabs open: "SQL File 3*" and "Result Grid". The SQL tab contains the following code:

```

1 • use cse370;
2
3 • select * from lab_grades where major = 'CSE';
4
5

```

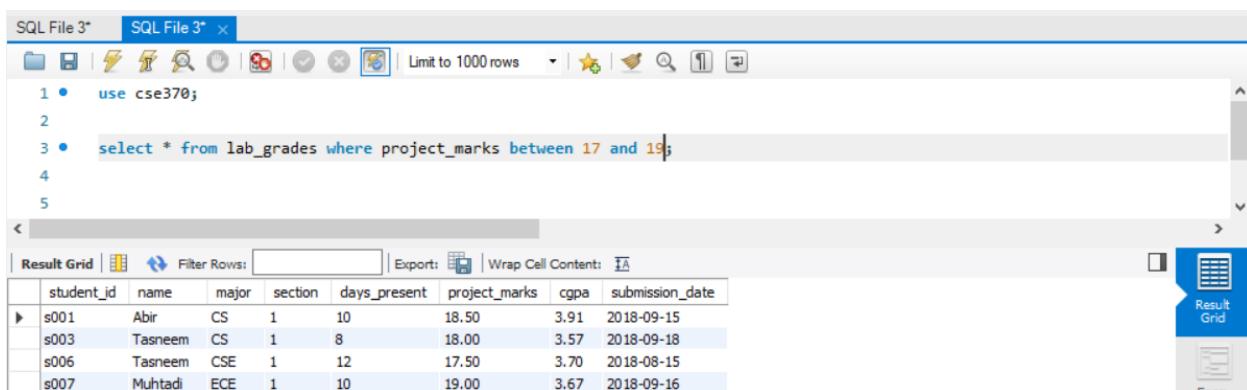
The Result Grid tab displays the following data:

student_id	name	major	section	days_present	project_marks	cgpa	submission_date
s002	Nafis	CSE	1	12	20.00	3.86	2018-08-15
s005	Arafat	CSE	2	11	20.00	4.00	2018-09-13
s006	Tasneem	CSE	1	12	17.50	3.70	2018-08-15
s008	Farhana	CSE	2	6	15.00	2.67	2018-08-16

- **Retrieve the Name and Project Marks of students whose Project Marks is between 17 and 19**

```
use cse370;
```

```
select * from lab_grades where project_marks between 17 and 19;
```



The screenshot shows the MySQL Workbench interface with two tabs open: "SQL File 3*" and "Result Grid". The SQL tab contains the following code:

```

1 • use cse370;
2
3 • select * from lab_grades where project_marks between 17 and 19;
4
5

```

The Result Grid tab displays the following data:

student_id	name	major	section	days_present	project_marks	cgpa	submission_date
s001	Abir	CS	1	10	18.50	3.91	2018-09-15
s003	Tasneem	CS	1	8	18.00	3.57	2018-09-18
s006	Tasneem	CSE	1	12	17.50	3.70	2018-08-15
s007	Muhtadi	ECE	1	10	19.00	3.67	2018-09-16

- **Retrieve the details of students who are majoring in either CS or CSE**

```
use cse370;
```

```
select * from lab_grades where major = 'CSE' or major = 'CS';
```

The screenshot shows the MySQL Workbench interface with two tabs open: "SQL File 3*" and "Result Grid". The SQL tab contains the following code:

```
1 • use cse370;
2
3 • select * from lab_grades where major = 'CSE' or major = 'CS';
4
5
```

The Result Grid tab displays the following data:

student_id	name	major	section	days_present	project_marks	cgpa	submission_date
s001	Abir	CS	1	10	18.50	3.91	2018-09-15
s002	Nafis	CSE	1	12	20.00	3.86	2018-08-15
s003	Tasneem	CS	1	8	18.00	3.57	2018-09-18
s005	Arafat	CSE	2	11	20.00	4.00	2018-09-13
s006	Tasneem	CSE	1	12	17.50	3.70	2018-08-15
S008	Farhana	CSE	2	6	15.00	2.67	2018-08-16

```
use cse370;
```

```
select * from lab_grades where major in ('CSE', 'CS');
```

The screenshot shows the MySQL Workbench interface with two tabs open: "SQL File 3*" and "Result Grid". The SQL tab contains the following code:

```
1 • use cse370;
2
3 • select * from lab_grades where major in ('CSE', 'CS');
4
5
```

The Result Grid tab displays the following data:

student_id	name	major	section	days_present	project_marks	cgpa	submission_date
s001	Abir	CS	1	10	18.50	3.91	2018-09-15
s002	Nafis	CSE	1	12	20.00	3.86	2018-08-15
s003	Tasneem	CS	1	8	18.00	3.57	2018-09-18
s005	Arafat	CSE	2	11	20.00	4.00	2018-09-13
s006	Tasneem	CSE	1	12	17.50	3.70	2018-08-15
S008	Farhana	CSE	2	6	15.00	2.67	2018-08-16

- **Retrieve the details of the students who submitted their Project in August and whose marks is greater than 18**

```
use cse370;
```

```
select * from lab_grades where Project_marks > 18 and submission_date between '2018-08-01' and '2018-08-31';
```

The screenshot shows the MySQL Workbench interface with two tabs open: "SQL File 3*" and "Result Grid". The SQL tab contains the following code:

```

1 • use cse370;
2
3 • Select * from lab_grades where Project_marks > 18 and Submission_date between '2018-08-01' and '2018-08-31';
4
5

```

The Result Grid tab displays the following data:

student_id	name	major	section	days_present	project_marks	cgpa	submission_date
s002	Nafis	CSE	1	12	20.00	3.86	2018-08-15

- **Retrieve the details of students whose name start with 'a'**

```
use cse370;
```

```
select * from lab_grades where name like 'a%';
```

The screenshot shows the MySQL Workbench interface with two tabs open: "SQL File 3*" and "Result Grid". The SQL tab contains the following code:

```

1 • use cse370;
2
3 • Select * from lab_grades where name like 'a%' ;
4
5

```

The Result Grid tab displays the following data:

student_id	name	major	section	days_present	project_marks	cgpa	submission_date
s001	Abir	CS	1	10	18.50	3.91	2018-09-15
s005	Arafat	CSE	2	11	20.00	4.00	2018-09-13

- **Retrieve the details of students whose name contains at least 2 a's**

```
use cse370;
```

```
Select * from lab_grades where name like '%a%a%';
```

The screenshot shows the MySQL Workbench interface with a query editor window titled "SQL File 3*". The code entered is:

```
1 • use cse370;
2
3 • Select * from lab_grades where name like '%a%a%';
4
5
```

Below the code, the results are displayed in a "Result Grid" table:

student_id	name	major	section	days_present	project_marks	cgpa	submission_date
s005	Arafat	CSE	2	11	20.00	4.00	2018-09-13
S008	Farhana	CSE	2	6	15.00	2.67	2018-08-16

- **Understand the pattern**

```
use cse370;
```

```
select name from lab_grades where name like 'f_r____';
```

The screenshot shows the MySQL Workbench interface with a query editor window titled "SQL File 3*". The code entered is:

```
1 • use cse370;
2
3 • select name from lab_grades where name like 'f_r____';
4
5
```

Below the code, the results are displayed in a "Result Grid" table:

name
Farhana

Lab 3: SQL Subqueries & Aggregate Functions

For This Lab Grades table has been refactored

```
use cse370;
```

```
delete from lab_grades where student_id in ('s004', 'S008');
```

```
select * from lab_grades;
```

The screenshot shows the MySQL Workbench interface with two tabs open: "SQL File 3*" and "Result Grid". The SQL tab contains the following code:

```
1 • use cse370;
2
3 • delete from lab_grades where student_id in ('s004', 'S008');
4
5 • select * from lab_grades;
```

The Result Grid tab displays the following data:

student_id	name	major	section	days_present	project_marks	cgpa	submission_date
s001	Abir	CS	1	10	18.50	3.91	2018-09-15
s002	Nafis	CSE	1	12	20.00	3.86	2018-08-15
s003	Tasneem	CS	1	8	18.00	3.57	2018-09-18
s005	Arafat	CSE	2	11	20.00	4.00	2018-09-13
s006	Tasneem	CSE	1	12	17.50	3.70	2018-08-15
s007	Muhtadi	ECE	1	10	19.00	3.67	2018-09-16

Current Table:

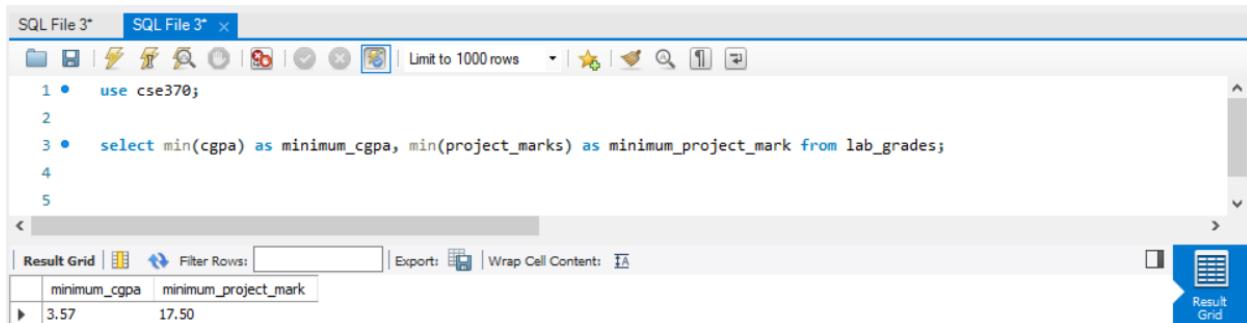
	student_id	name	major	section	days_present	project_marks	cgpa	submission_date
▶	s001	Abir	CS	1	10	18.50	3.91	2018-09-15
	s002	Nafis	CSE	1	12	20.00	3.86	2018-08-15
	s003	Tasneem	CS	1	8	18.00	3.57	2018-09-18
	s005	Arafat	CSE	2	11	20.00	4.00	2018-09-13
	s006	Tasneem	CSE	1	12	17.50	3.70	2018-08-15
	s007	Muhtadi	ECE	1	10	19.00	3.67	2018-09-16

Aggregate Functions, Group By and Having:

- **Retrieve the minimum CGPA/Project_marks from the table**

```
use cse370;
```

```
select min(CGPA) as minimum_cgpa, min(project_marks) as minimum_project_mark from lab_grades;
```



The screenshot shows the MySQL Workbench interface with a SQL editor window titled "SQL File 3*". The code entered is:

```
1 • use cse370;
2
3 • select min(CGPA) as minimum_cgpa, min(project_marks) as minimum_project_mark from lab_grades;
4
5
```

Below the code, the result grid displays the following data:

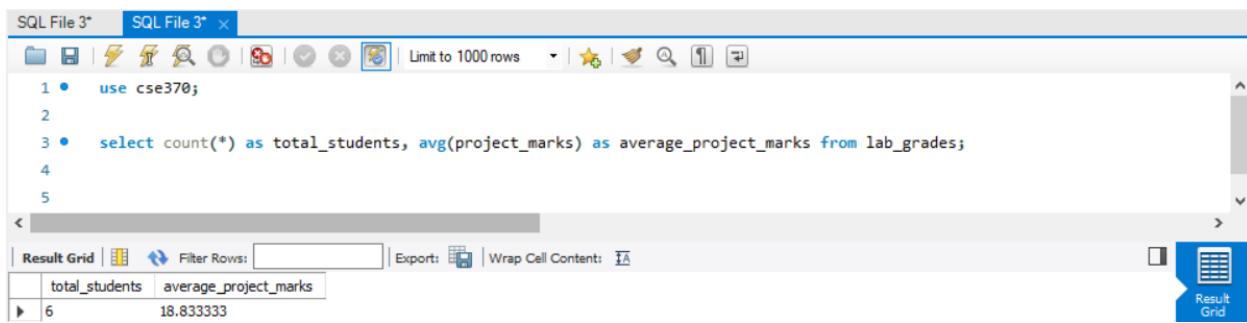
minimum_cgpa	minimum_project_mark
3.57	17.50

A blue "Result Grid" button is visible on the right side of the result grid area.

- **Retrieve the total number of students and the average Projects Marks**

```
use cse370;
```

```
select count(*) as total_students, avg(project_marks) as average_project_marks from lab_grades;
```



The screenshot shows the MySQL Workbench interface with a SQL editor window titled "SQL File 3*". The code entered is:

```
1 • use cse370;
2
3 • select count(*) as total_students, avg(project_marks) as average_project_marks from lab_grades;
4
5
```

Below the code, the result grid displays the following data:

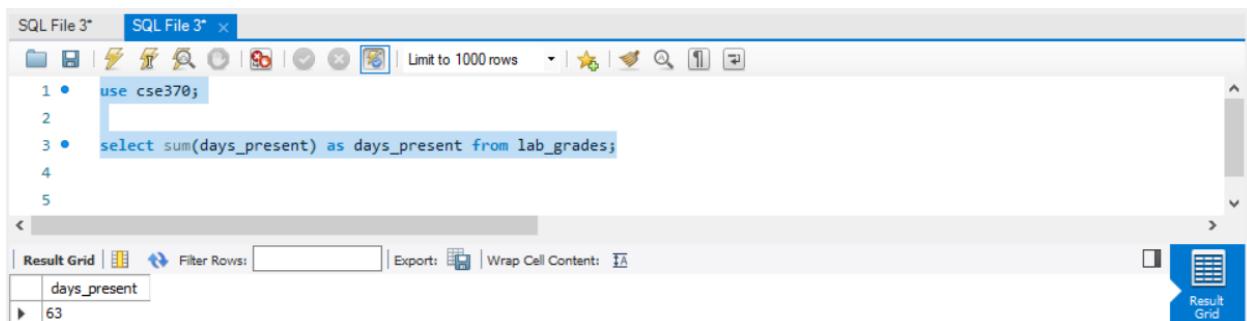
total_students	average_project_marks
6	18.833333

A blue "Result Grid" button is visible on the right side of the result grid area.

- **Find the sum of the number of Days Present**

```
use cse370;
```

```
select sum(days_present) as days_present from lab_grades;
```



The screenshot shows the MySQL Workbench interface with a single tab titled "SQL File 3". The SQL editor contains the following code:

```
1 • use cse370;
2
3 • select sum(days_present) as days_present from lab_grades;
4
5
```

The results pane below shows a single row of data:

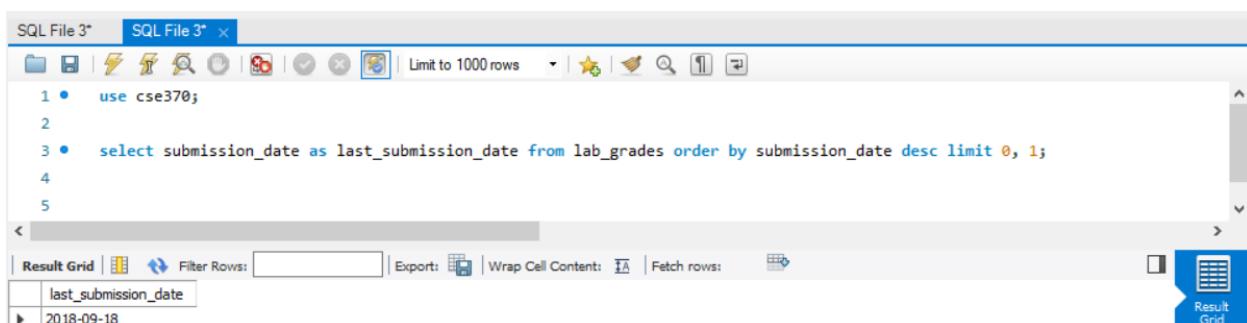
days_present
63

A "Result Grid" button is visible in the bottom right corner of the results pane.

- **Retrieve the last Submission Date**

```
use cse370;
```

```
select submission_date as last_submission_date from lab_grades order by submission_date
desc limit 0, 1;
```



The screenshot shows the MySQL Workbench interface with a single tab titled "SQL File 3". The SQL editor contains the following code:

```
1 • use cse370;
2
3 • select submission_date as last_submission_date from lab_grades order by submission_date desc limit 0, 1;
4
5
```

The results pane below shows a single row of data:

last_submission_date
2018-09-18

A "Result Grid" button is visible in the bottom right corner of the results pane.

- **Find Minimum and Maximum CGPA / Project Marks of each Major**

```
use cse370;
```

```
select major, min(CGPA) as minimum_cgpa, max(CGPA) as max_cgpa, min(project_marks) as minimum_project_mark, max(project_marks) as max_project_mark from lab_grades group by major;
```

The screenshot shows the MySQL Workbench interface with a SQL editor tab titled "SQL File 3". The code entered is:

```
1 • use cse370;
2
3 • select major, min(CGPA) as minimum_cgpa, max(CGPA) as max_cgpa, min(project_marks) as minimum_project_mark,
4     max(project_marks) as max_project_mark from lab_grades group by major;
5
```

The results are displayed in a "Result Grid" table:

major	minimum_cgpa	max_cgpa	minimum_project_mark	max_project_mark
CS	3.57	3.91	18.00	18.50
CSE	3.70	4.00	17.50	20.00
ECE	3.67	3.67	19.00	19.00

- **Retrieve total number of students for each Major**

```
use cse370;
```

```
select major, count(student_id) as number_of_students from lab_grades group by major;
```

The screenshot shows the MySQL Workbench interface with a SQL editor tab titled "SQL File 3". The code entered is:

```
1 • use cse370;
2
3 • select major, count(student_id) as number_of_students from lab_grades group by major;
4
5
```

The results are displayed in a "Result Grid" table:

major	number_of_students
CS	2
CSE	3
ECE	1

- For each major find the minimum and maximum CGPA / Project Marks, but only if there were at least 2 students in the Major

```
use cse370;
```

```
select major, min(cgpa) as minimum_cgpa, max(cgpa) as max_cgpa, min(project_marks) as minimum_project_mark, max(project_marks) as max_project_mark from lab_grades group by major having count(*) >= 2;
```

The screenshot shows the MySQL Workbench interface with two tabs open: "SQL File 3*" and "Result Grid". The SQL tab contains the following code:

```
1 • use cse370;
2
3 • select major, min(cgpa) as minimum_cgpa, max(cgpa) as max_cgpa, min(project_marks) as minimum_project_mark,
4     max(project_marks) as max_project_mark from lab_grades group by major having count(*) >= 2;
5
```

The Result Grid tab displays the following table:

major	minimum_cgpa	max_cgpa	minimum_project_mark	max_project_mark
CS	3.57	3.91	18.00	18.50
CSE	3.70	4.00	17.50	20.00

- For each major find the minimum and maximum CGPA / Project Marks, but consider only students who submitted before or on 15th September

```
use cse370;
```

```
select submission_date, major, min(cgpa) as minimum_cgpa, max(cgpa) as max_cgpa,
min(project_marks) as minimum_project_mark, max(project_marks) as max_project_mark from
lab_grades where submission_date <= '2018-09-15' group by major;
```

The screenshot shows the MySQL Workbench interface with two tabs open: "SQL File 3*" and "Result Grid". The SQL tab contains the following code:

```
1 • use cse370;
2
3 • select submission_date, major, min(cgpa) as minimum_cgpa, max(cgpa) as max_cgpa, min(project_marks) as minimum_project_mark,
4     max(project_marks) as max_project_mark from lab_grades where submission_date <= '2018-09-15' group by major;
5
```

The Result Grid tab displays the following table:

submission_date	major	minimum_cgpa	max_cgpa	minimum_project_mark	max_project_mark
2018-09-15	CS	3.91	3.91	18.50	18.50
2018-08-15	CSE	3.70	4.00	17.50	20.00

A HAVING clause is like a WHERE clause, but applies only to groups as a whole (that is, to the rows in the result set representing groups), whereas the WHERE clause applies to individual rows.

Sub Queries / Nested Queries, Any and All:

- Name of the Student who received highest Project Mark

```
use cse370;
```

```
select name, project_marks from lab_grades where project_marks = (select max(project_marks) from lab_grades);
```

The screenshot shows the MySQL Workbench interface with a SQL editor tab titled "SQL File 3*". The code entered is:

```
1 • use cse370;
2
3 • select name, project_marks from lab_grades where project_marks = (select max(project_marks) from lab_grades);
4
5
```

The result grid displays the following data:

name	project_marks
Nafis	20.00
Arafat	20.00

- For each major find the Name of the student who has the lowest attendance

```
use cse370;
```

```
select name, major, days_present from lab_grades where (major, days_present) in (select major, min(days_present) from lab_grades group by major);
```

The screenshot shows the MySQL Workbench interface with a SQL editor tab titled "SQL File 3*". The code entered is:

```
1 • use cse370;
2
3 • select name, major, days_present from lab_grades where (major, days_present) in
4   (select major, min(days_present) from lab_grades group by major);
5
```

The result grid displays the following data:

name	major	days_present
Tasneem	CS	8
Arafat	CSE	11
Muhtadi	ECE	10

Table:

	student_id	name	major	section	days_present	project_marks	cgpa	submission_date
▶	s001	Abir	CS	1	10	18.50	3.91	2018-09-15
	s002	Nafis	CSE	1	12	20.00	3.86	2018-08-15
	s003	Tasneem	CS	1	8	18.00	3.57	2018-09-18
	s005	Arafat	CSE	2	11	20.00	4.00	2018-09-13
	s006	Tasneem	CSE	1	12	17.50	3.70	2018-08-15
	s007	Muhtadi	ECE	1	10	19.00	3.67	2018-09-16

- Retrieve the CSE students whose CGPA / Project Marks is higher than at least 1 CS students

```
use cse370;
```

```
select name, project_marks from lab_grades where major = 'CSE' and project_marks > any (select project_marks from lab_grades where major = 'CS');
```

The screenshot shows the MySQL Workbench interface with a SQL editor window containing the following code:

```
1 • use cse370;
2
3 • select name, project_marks from lab_grades where major = 'CSE' and project_marks >
4   any (select project_marks from lab_grades where major = 'CS');
5
```

The result grid shows two rows of data:

name	project_marks
Nafis	20.00
Arafat	20.00

- Retrieve the CSE students whose CGPA / Project Marks is higher than all CS students

```
use cse370;
```

```
select name, cgpa from lab_grades where major = 'CSE' and cgpa > all (select cgpa from lab_grades where major = 'CS');
```

The screenshot shows the MySQL Workbench interface with a SQL editor window containing the following code:

```
1 • use cse370;
2
3 • select name, cgpa from lab_grades where major = 'CSE' and cgpa >
4   all (select cgpa from lab_grades where major = 'CS');
5
```

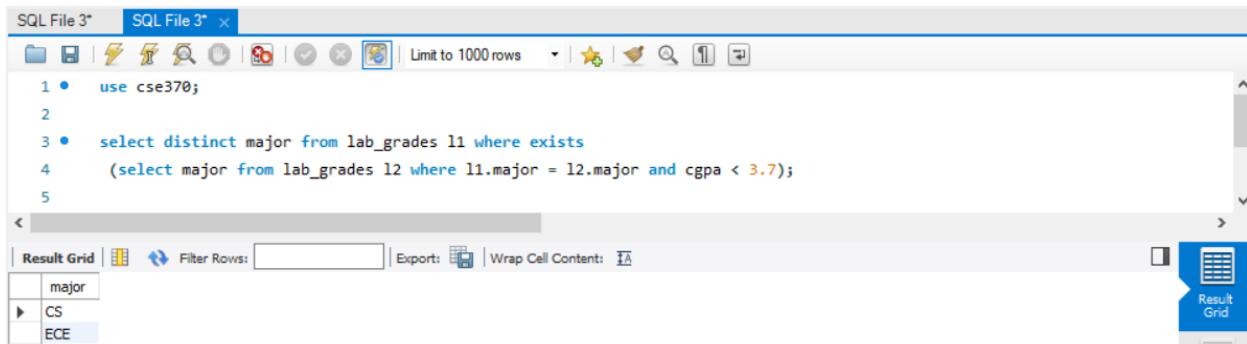
The result grid shows one row of data:

name	cgpa
Arafat	4.00

- **Select those Majors for which at least 1 student has CGPA lower than 3.7 / Project Marks < 18**

```
use cse370;
```

```
select distinct major from lab_grades l1 where exists (select major from lab_grades l2 where l1.major = l2.major and cgpa < 3.7);
```



The screenshot shows the SQL Server Management Studio interface. The query window contains the following code:

```
1 • use cse370;
2
3 • select distinct major from lab_grades l1 where exists
4   (select major from lab_grades l2 where l1.major = l2.major and cgpa < 3.7);
5
```

The results pane displays the output of the query:

major
CS
ECE

- **Select 2nd most Days Present**

```
use cse370;
```

```
select days_present as second_most_days_present from lab_grades where days_present =
(select days_present from lab_grades group by days_present order by days_present limit 2, 1);
```



The screenshot shows the SQL Server Management Studio interface. The query window contains the following code:

```
1 • use cse370;
2
3 • select days_present as second_most_days_present from lab_grades where days_present
4   = (select days_present from lab_grades group by days_present order by days_present limit 2, 1);
5
```

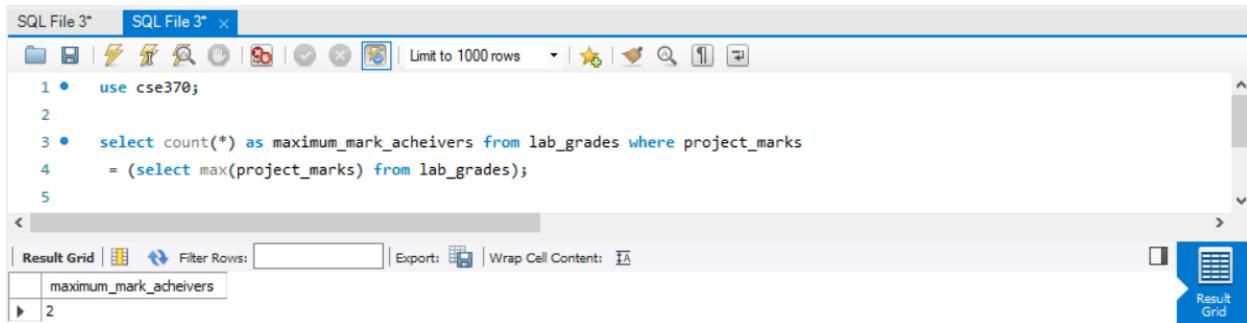
The results pane displays the output of the query:

second_most_days_present
11

- **Retrieve the total number of students who obtained the maximum Marks**

```
use cse370;
```

```
select count(*) as maximum_mark_acheivers from lab_grades where project_marks = (select max(project_marks) from lab_grades);
```



The screenshot shows the MySQL Workbench interface with a SQL editor tab titled "SQL File 3". The query is:

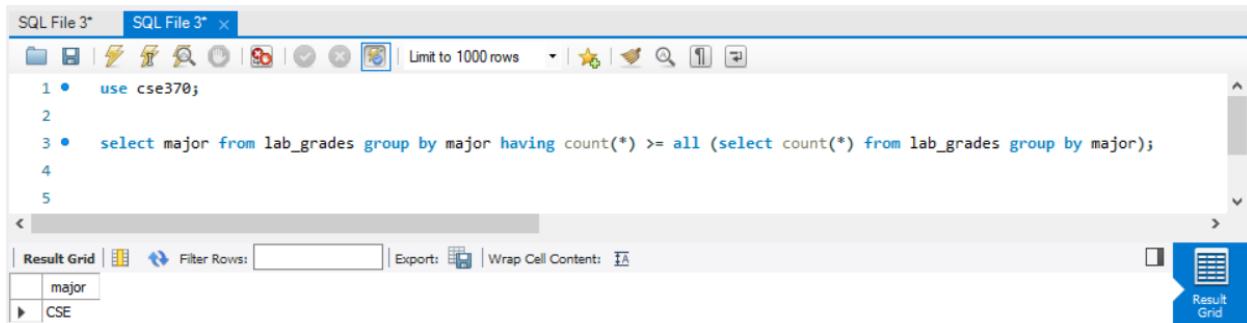
```
1 • use cse370;
2
3 • select count(*) as maximum_mark_acheivers from lab_grades where project_marks
4     = (select max(project_marks) from lab_grades);
5
```

The result grid shows one row with the value 2.

- **Retrieve the Major which has the highest number of students enrolled**

```
use cse370;
```

```
select major from lab_grades group by major having count(*) >= all (select count(*) from lab_grades group by major);
```



The screenshot shows the MySQL Workbench interface with a SQL editor tab titled "SQL File 3". The query is:

```
1 • use cse370;
2
3 • select major from lab_grades group by major having count(*) >= all (select count(*) from lab_grades group by major);
4
5
```

The result grid shows one row with the value "CSE".

Order

```
SELECT column_name(s)
FROM table_name(s)
WHERE conditions
GROUP BY column_name(s)
HAVING conditions
ORDER BY column_name(s);
```

Introduction to Bank DB, SQL Joins and Constraints

- Create Tables

```
use cse370;
```

```
create table customer (customer_id varchar(10) not null, customer_name varchar(20) not null,  
customer_street varchar(30), customer_city varchar(30), primary key (customer_id)) ;
```

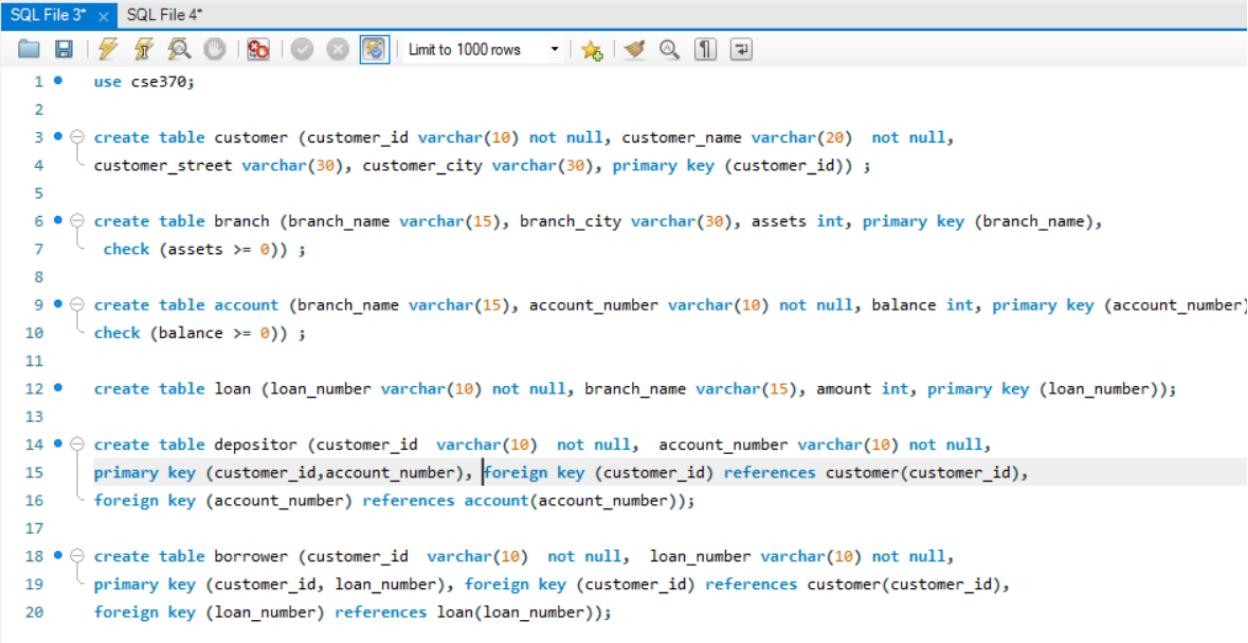
```
create table branch (branch_name varchar(15), branch_city varchar(30), assets int, primary key  
(branch_name), check (assets >= 0)) ;
```

```
create table account (branch_name varchar(15), account_number varchar(10) not null, balance  
int, primary key (account_number), check (balance >= 0)) ;
```

```
create table loan (loan_number varchar(10) not null, branch_name varchar(15), amount int,  
primary key (loan_number));
```

```
create table depositor (customer_id varchar(10) not null, account_number varchar(10) not  
null, primary key (customer_id,account_number), foreign key (customer_id) references  
customer(customer_id), foreign key (account_number) references account(account_number));
```

```
create table borrower (customer_id varchar(10) not null, loan_number varchar(10) not null,  
primary key (customer_id, loan_number), foreign key (customer_id) references  
customer(customer_id), foreign key (loan_number) references loan(loan_number));
```



```
SQL File 3* x SQL File 4*  
1 • use cse370;  
2  
3 • ⏷ create table customer (customer_id varchar(10) not null, customer_name varchar(20) not null,  
4     customer_street varchar(30), customer_city varchar(30), primary key (customer_id)) ;  
5  
6 • ⏷ create table branch (branch_name varchar(15), branch_city varchar(30), assets int, primary key (branch_name),  
7     check (assets >= 0)) ;  
8  
9 • ⏷ create table account (branch_name varchar(15), account_number varchar(10) not null, balance int, primary key (account_number)  
10    check (balance >= 0)) ;  
11  
12 •   create table loan (loan_number varchar(10) not null, branch_name varchar(15), amount int, primary key (loan_number));  
13  
14 • ⏷ create table depositor (customer_id varchar(10) not null, account_number varchar(10) not null,  
15     primary key (customer_id,account_number), [foreign key (customer_id) references customer(customer_id),  
16     foreign key (account_number) references account(account_number));  
17  
18 • ⏷ create table borrower (customer_id varchar(10) not null, loan_number varchar(10) not null,  
19     primary key (customer_id, loan_number), foreign key (customer_id) references customer(customer_id),  
20     foreign key (loan_number) references loan(loan_number));
```

- **Insert data into the tables**

```
use cse370;
```

```
insert into customer values ('C-101','Jones', 'Main', 'Harrison'), ('C-201','Smith', 'North', 'Rye'),  
('C-211','Hayes', 'Main', 'Harrison'), ('C-212','Curry', 'North', 'Rye'), ('C-215','Lindsay', 'Park',  
'Pittsfield'), ('C-220','Turner', 'Putnam', 'Stamford'), ('C-222','Williams', 'Nassau', 'Princeton'),  
('C-225','Adams', 'Spring', 'Pittsfield'), ('C-226','Johnson', 'Alma', 'Palo Alto'),  
('C-233','Glenn', 'Sand Hill', 'Woodside'), ('C-234','Brooks', 'Senator', 'Brooklyn'),  
('C-255','Green', 'Walnut', 'Stamford');
```

insert into branch values

```
('Downtown', 'Brooklyn',9000000), ('Redwood', 'Palo Alto',2100000), ('Perryridge',  
'Horseneck',1700000), ('Mianus', 'Horseneck',400000), ('Round Hill', 'Horseneck',8000000),  
('Pownal', 'Bennington',300000), ('North Town', 'Rye',3700000), ('Brighton', 'Brooklyn',7100000);
```

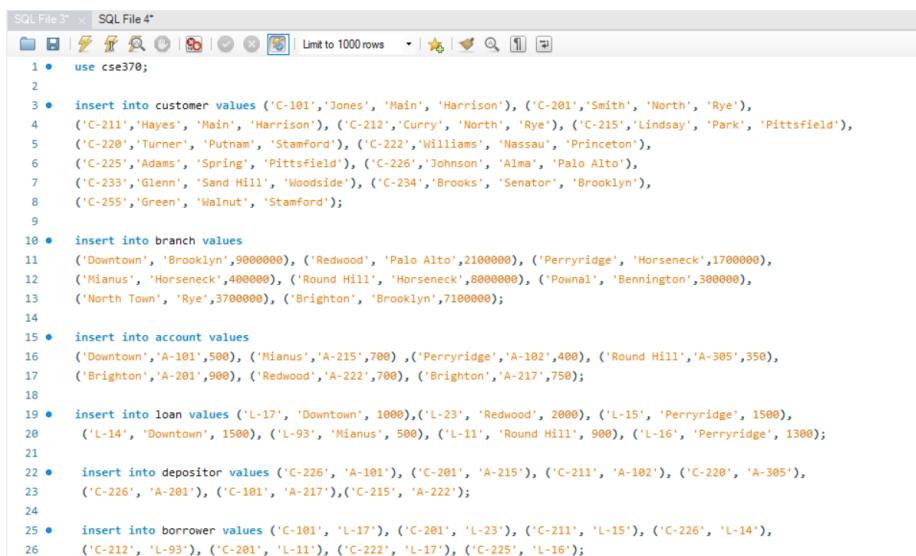
insert into account values

```
('Downtown','A-101',500), ('Mianus','A-215',700) ,('Perryridge','A-102',400), ('Round  
Hill','A-305',350), ('Brighton','A-201',900), ('Redwood','A-222',700), ('Brighton','A-217',750);
```

```
insert into loan values ('L-17', 'Downtown', 1000),('L-23', 'Redwood', 2000), ('L-15', 'Perryridge',  
1500), ('L-14', 'Downtown', 1500), ('L-93', 'Mianus', 500), ('L-11', 'Round Hill', 900), ('L-16',  
'Perryridge', 1300);
```

```
insert into depositor values ('C-226', 'A-101'), ('C-201', 'A-215'), ('C-211', 'A-102'), ('C-220',  
'A-305'), ('C-226', 'A-201'), ('C-101', 'A-217'), ('C-215', 'A-222');
```

```
insert into borrower values ('C-101', 'L-17'), ('C-201', 'L-23'), ('C-211', 'L-15'), ('C-226', 'L-14'),  
('C-212', 'L-93'), ('C-201', 'L-11'), ('C-222', 'L-17'), ('C-225', 'L-16');
```



The screenshot shows a SQL Server Management Studio window with two tabs: 'SQL File 3' and 'SQL File 4'. The code listed below is contained within the 'SQL File 4' tab.

```
SQL File 3 × SQL File 4*
use cse370;
1 • insert into customer values ('C-101','Jones', 'Main', 'Harrison'), ('C-201','Smith', 'North', 'Rye'),  
2 ('C-211','Hayes', 'Main', 'Harrison'), ('C-212','Curry', 'North', 'Rye'), ('C-215','Lindsay', 'Park', 'Pittsfield'),  
3 ('C-220','Turner', 'Putnam', 'Stamford'), ('C-222','Williams', 'Nassau', 'Princeton'),  
4 ('C-225','Adams', 'Spring', 'Pittsfield'), ('C-226','Johnson', 'Alma', 'Palo Alto'),  
5 ('C-233','Glenn', 'Sand Hill', 'Woodside'), ('C-234','Brooks', 'Senator', 'Brooklyn'),  
6 ('C-255','Green', 'Walnut', 'Stamford');
7
8 • insert into branch values
9 ('Downtown', 'Brooklyn',9000000), ('Redwood', 'Palo Alto',2100000), ('Perryridge',  
10 'Horseneck',1700000), ('Mianus', 'Horseneck',400000), ('Round Hill', 'Horseneck',8000000),  
11 ('Pownal', 'Bennington',300000), ('North Town', 'Rye',3700000), ('Brighton', 'Brooklyn',7100000);
12
13 • insert into account values
14 ('Downtown','A-101',500), ('Mianus','A-215',700) ,('Perryridge','A-102',400), ('Round  
Hill','A-305',350), ('Brighton','A-201',900), ('Redwood','A-222',700), ('Brighton','A-217',750);
15
16 • insert into loan values ('L-17', 'Downtown', 1000),('L-23', 'Redwood', 2000), ('L-15', 'Perryridge', 1500),
17 ('L-14', 'Downtown', 1500), ('L-93', 'Mianus', 500), ('L-11', 'Round Hill', 900), ('L-16', 'Perryridge', 1300);
18
19 • insert into depositor values ('C-226', 'A-101'), ('C-201', 'A-215'), ('C-211', 'A-102'), ('C-220', 'A-305'),
20 ('C-226', 'A-201'), ('C-101', 'A-217'), ('C-215', 'A-222');
21
22 • insert into borrower values ('C-101', 'L-17'), ('C-201', 'L-23'), ('C-211', 'L-15'), ('C-226', 'L-14'),
23 ('C-212', 'L-93'), ('C-201', 'L-11'), ('C-222', 'L-17'), ('C-225', 'L-16');
```

Customer Table:

	Field	Type	Null	Key	Default	Extra
▶	customer_id	varchar(10)	NO	PRI	NULL	
	customer_name	varchar(20)	NO		NULL	
	customer_street	varchar(30)	YES		NULL	
	customer_city	varchar(30)	YES		NULL	

	customer_id	customer_name	customer_street	customer_city
▶	C-101	Jones	Main	Harrison
	C-201	Smith	North	Rye
	C-211	Hayes	Main	Harrison
	C-212	Curry	North	Rye
	C-215	Lindsay	Park	Pittsfield
	C-220	Turner	Putnam	Stamford
	C-222	Williams	Nassau	Princeton
	C-225	Adams	Spring	Pittsfield
	C-226	Johnson	Alma	Palo Alto
	C-233	Glenn	Sand Hill	Woodside
	C-234	Brooks	Senator	Brooklyn
	C-255	Green	Walnut	Stamford
*	NULL	NULL	NULL	NULL

Branch Table:

	Field	Type	Null	Key	Default	Extra
▶	branch_name	varchar(15)	NO	PRI	NULL	
	branch_city	varchar(30)	YES		NULL	
	assets	int	YES		NULL	

	branch_name	branch_city	assets
▶	Brighton	Brooklyn	7100000
	Downtown	Brooklyn	9000000
	Mianus	Horseneck	400000
	North Town	Rye	3700000
	Perryridge	Horseneck	1700000
	Pownal	Bennington	300000
	Redwood	Palo Alto	2100000
	Round Hill	Horseneck	8000000
*	HULL	HULL	HULL

Account Table:

	Field	Type	Null	Key	Default	Extra
▶	branch_name	varchar(15)	YES		HULL	
	account_number	varchar(10)	NO	PRI	HULL	
	balance	int	YES		HULL	

	branch_name	account_number	balance
▶	Downtown	A-101	500
	Perryridge	A-102	400
	Brighton	A-201	900
	Mianus	A-215	700
	Brighton	A-217	750
	Redwood	A-222	700
	Round Hill	A-305	350
*	HULL	HULL	HULL

Loan Table:

	Field	Type	Null	Key	Default	Extra
▶	loan_number	varchar(10)	NO	PRI	HULL	
	branch_name	varchar(15)	YES		HULL	
	amount	int	YES		HULL	

	loan_number	branch_name	amount
▶	L-11	Round Hill	900
	L-14	Downtown	1500
	L-15	Perryridge	1500
	L-16	Perryridge	1300
	L-17	Downtown	1000
	L-23	Redwood	2000
	L-93	Mianus	500
*	NULL	NULL	NULL

Depositor Table:

	Field	Type	Null	Key	Default	Extra
▶	customer_id	varchar(10)	NO	PRI	NULL	
	account_number	varchar(10)	NO	PRI	NULL	

	customer_id	account_number
▶	C-226	A-101
	C-211	A-102
	C-226	A-201
	C-201	A-215
	C-101	A-217
	C-215	A-222
	C-220	A-305
*	NULL	NULL

Borrower Table:

	Field	Type	Null	Key	Default	Extra
▶	customer_id	varchar(10)	NO	PRI	NULL	
	loan_number	varchar(10)	NO	PRI	NULL	

	customer_id	loan_number
▶	C-201	L-11
	C-226	L-14
	C-211	L-15
	C-225	L-16
	C-101	L-17
	C-222	L-17
	C-201	L-23
	C-212	L-93
*	NULL	NULL

Customer, Account, Depositor Table Joined:

	customer_id	customer_name	customer_street	customer_city	customer_id	account_number	branch_name	account_number	balance
▶	C-226	Johnson	Alma	Palo Alto	C-226	A-101	Downtown	A-101	500
	C-211	Hayes	Main	Harrison	C-211	A-102	Perryridge	A-102	400
	C-226	Johnson	Alma	Palo Alto	C-226	A-201	Brighton	A-201	900
	C-201	Smith	North	Rye	C-201	A-215	Mianus	A-215	700
	C-101	Jones	Main	Harrison	C-101	A-217	Brighton	A-217	750
	C-215	Lindsay	Park	Pittsfield	C-215	A-222	Redwood	A-222	700
	C-220	Turner	Putnam	Stamford	C-220	A-305	Round Hill	A-305	350

Customer, Account, Depositor Table Joined:

	customer_id	customer_name	customer_street	customer_city	customer_id	loan_number	loan_number	branch_name	amount
▶	C-201	Smith	North	Rye	C-201	L-11	L-11	Round Hill	900
	C-226	Johnson	Alma	Palo Alto	C-226	L-14	L-14	Downtown	1500
	C-211	Hayes	Main	Harrison	C-211	L-15	L-15	Perryridge	1500
	C-225	Adams	Spring	Pittsfield	C-225	L-16	L-16	Perryridge	1300
	C-222	Williams	Nassau	Princeton	C-222	L-17	L-17	Downtown	1000
	C-101	Jones	Main	Harrison	C-101	L-17	L-17	Downtown	1000
	C-201	Smith	North	Rye	C-201	L-23	L-23	Redwood	2000
	C-212	Curry	North	Rye	C-212	L-93	L-93	Mianus	500

Retrieve all customer's Id, Name, City and Account Number using

Inner Join:

```
use cse370;
```

```
select c.customer_id, c.customer_name, c.customer_city, a.account_number from ((customer c
inner join depositor d on c.customer_id = d.customer_id) inner join account a on
a.account_number = d.account_number);
```

The screenshot shows the MySQL Workbench interface with three tabs at the top: SQL File 3*, SQL File 4*, and SQL File 5*. The SQL editor contains the following code:

```
1 • use cse370;
2
3 • select c.customer_id, c.customer_name, c.customer_city, a.account_number from ((depositor d join customer c on c.customer_id =
d.customer_id) join account a on a.account_number = d.account_number);
```

The results grid shows the following data:

customer_id	customer_name	customer_city	account_number
C-226	Johnson	Palo Alto	A-101
C-211	Hayes	Harrison	A-102
C-226	Johnson	Palo Alto	A-201
C-201	Smith	Rye	A-215
C-101	Jones	Harrison	A-217
C-215	Lindsay	Pittsfield	A-222
C-220	Turner	Stamford	A-305

Outer Joins:

Left Join:

```
use cse370;
```

```
select c.customer_id, c.customer_name, c.customer_city, a.account_number from ((customer c
left join depositor d on c.customer_id = d.customer_id) left join account a on a.account_number
= d.account_number);
```

The screenshot shows the MySQL Workbench interface with three tabs at the top: SQL File 3*, SQL File 4*, and SQL File 5*. The SQL editor contains the following code:

```
1 • use cse370;
2
3 • select c.customer_id, c.customer_name, c.customer_city, a.account_number from ((customer c left join depositor d on c.customer_id =
d.customer_id) left join account a on a.account_number = d.account_number);
```

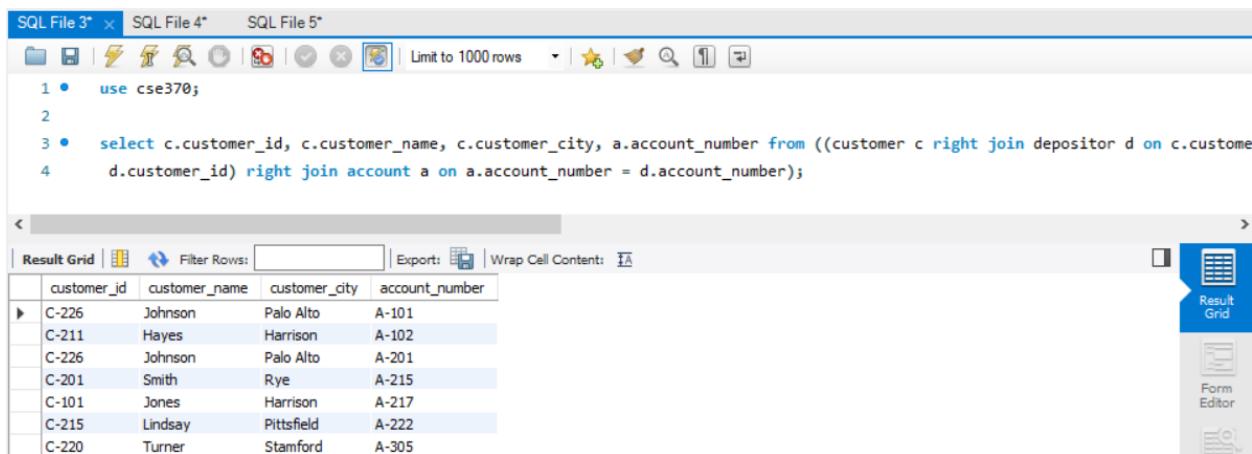
The results grid shows the following data:

customer_id	customer_name	customer_city	account_number
C-101	Jones	Harrison	A-217
C-201	Smith	Rye	A-215
C-211	Hayes	Harrison	A-102
C-212	Curry	Rye	NULL
C-215	Lindsay	Pittsfield	A-222
C-220	Turner	Stamford	A-305
C-222	Williams	Princeton	NULL
C-225	Adams	Pittsfield	NULL
C-226	Johnson	Palo Alto	A-101
C-226	Johnson	Palo Alto	A-201
C-233	Glenn	Woodside	NULL
C-234	Brooks	Brooklyn	NULL
C-255	Green	Stamford	NULL

Right Join:

```
use cse370;
```

```
select c.customer_id, c.customer_name, c.customer_city, a.account_number from ((customer c
right join depositor d on c.customer_id = d.customer_id) right join account a on
a.account_number = d.account_number);
```



The screenshot shows the SQL Server Management Studio interface. The query window contains the following code:

```
1 • use cse370;
2
3 • select c.customer_id, c.customer_name, c.customer_city, a.account_number from ((customer c
right join depositor d on c.customer_id = d.customer_id) right join account a on
d.customer_id) right join account a on a.account_number = d.account_number);
```

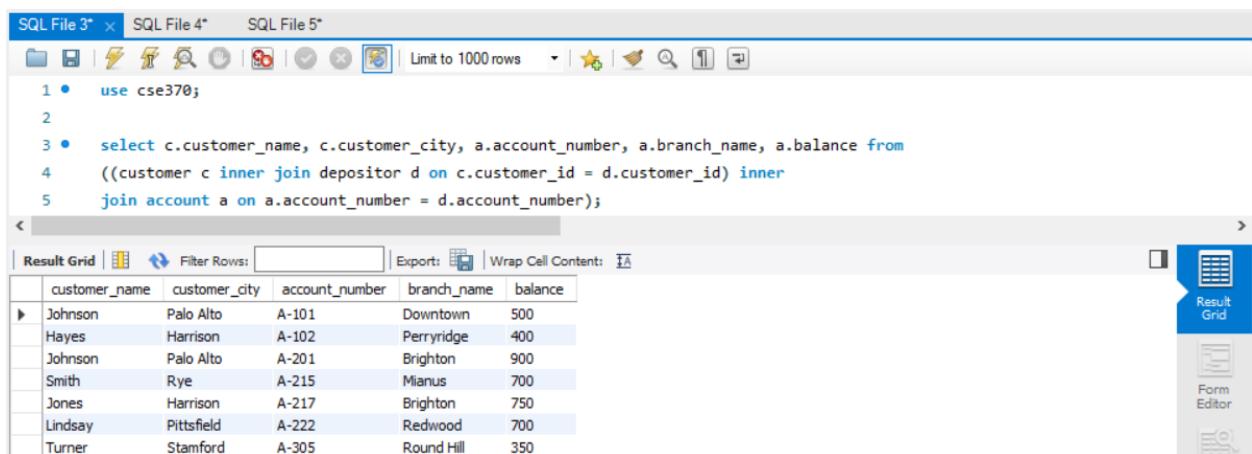
The results grid shows the following data:

customer_id	customer_name	customer_city	account_number
C-226	Johnson	Palo Alto	A-101
C-211	Hayes	Harrison	A-102
C-226	Johnson	Palo Alto	A-201
C-201	Smith	Rye	A-215
C-101	Jones	Harrison	A-217
C-215	Lindsay	Pittsfield	A-222
C-220	Turner	Stamford	A-305

- **Retrieve the following information from your database using “join”: Customer Name, City, Account Number, Balance and Branch name**

```
use cse370;
```

```
select c.customer_name, c.customer_city, a.account_number, a.branch_name, a.balance from
((customer c inner join depositor d on c.customer_id = d.customer_id) inner join account a on
a.account_number = d.account_number);
```



The screenshot shows the SQL Server Management Studio interface. The query window contains the following code:

```
1 • use cse370;
2
3 • select c.customer_name, c.customer_city, a.account_number, a.branch_name, a.balance from
((customer c inner join depositor d on c.customer_id = d.customer_id) inner
join account a on a.account_number = d.account_number);
```

The results grid shows the following data:

customer_name	customer_city	account_number	branch_name	balance
Johnson	Palo Alto	A-101	Downtown	500
Hayes	Harrison	A-102	Perryridge	400
Johnson	Palo Alto	A-201	Brighton	900
Smith	Rye	A-215	Mianus	700
Jones	Harrison	A-217	Brighton	750
Lindsay	Pittsfield	A-222	Redwood	700
Turner	Stamford	A-305	Round Hill	350

- **Find Names and Cities of Customers who have a Loan at Perryridge branch**

```
use cse370;
```

```
select customer_name as have_loan_at_Perryridge, customer_city from customer c join
borrower b on c.customer_id = b.customer_id join loan l on l.loan_number = b.loan_number
where branch_name = 'Perryridge'
```

The screenshot shows the MySQL Workbench interface with three tabs at the top: SQL File 3*, SQL File 4*, and SQL File 5*. Below the tabs is a toolbar with various icons. The main area contains the following SQL code:

```
1 • use cse370;
2
3 • select customer_name as have_loan_at_Perryridge, customer_city from customer c join
   borrower b on c.customer_id = b.customer_id join loan l on l.loan_number = b.loan_number
   where branch_name = 'Perryridge';
```

Below the code is a result grid titled "Result Grid". It has two columns: "have_loan_at_Perryridge" and "customer_city". The data is as follows:

have_loan_at_Perryridge	customer_city
Hayes	Harrison
Adams	Pittsfield

- **Find which Accounts with Balances between 700 and 900**

```
use cse370;
```

```
select account_number from account where balance between 700 and 900;
```

The screenshot shows the MySQL Workbench interface with three tabs at the top: SQL File 3*, SQL File 4*, and SQL File 5*. Below the tabs is a toolbar with various icons. The main area contains the following SQL code:

```
1 • use cse370;
2
3 • select account_number from account where balance between 700 and 900;
```

Below the code is a result grid titled "Result Grid". It has one column: "account_number". The data is as follows:

account_number
A-201
A-215
A-217
A-222
NULL

- **Find the Names of Customers on streets with Names ending in "Hill"**

```
use cse370;
```

```
select customer_name from customer where customer_street like '%Hill';
```

The screenshot shows the MySQL Workbench interface with three tabs at the top: SQL File 3*, SQL File 4*, and SQL File 5*. The SQL editor contains the following code:

```
1 • use cse370;
2
3 • select customer_name from customer where customer_street like '%Hill';
```

The results pane shows a single row in the 'Result Grid' tab:

customer_name
Glenn

- **Find the Names of Branches whose assets are greater than the assets of some Branch in Brooklyn**

```
use cse370;
```

```
select branch_name from branch where assets > any (select assets from branch where branch_city = 'Brooklyn');
```

[Brooklyn has two Branches Downtown (90000000) and Brighton (7100000). Here the Downtown branch and Round Hill (8000000) Branch is bigger than any of the two Branches which is Brighton.]

The screenshot shows the MySQL Workbench interface with three tabs at the top: SQL File 3*, SQL File 4*, and SQL File 5*. The SQL editor contains the following code:

```
1 • use cse370;
2
3 • select branch_name from branch where assets > any (select assets from branch where branch_city = 'Brooklyn');
```

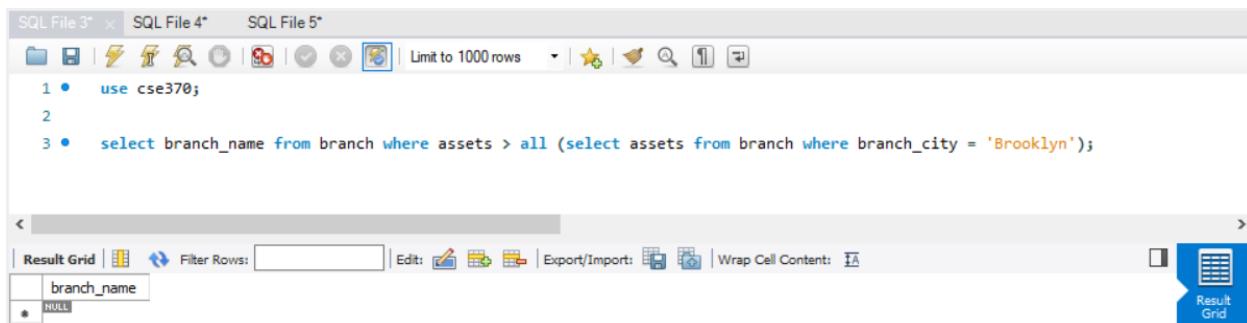
The results pane shows three rows in the 'Result Grid' tab:

branch_name
Downtown
Round Hill
NULL

```
use cse370;
```

```
select branch_name from branch where assets > all (select assets from branch where branch_city = 'Brooklyn');
```

[In terms of 'all' both Downtown (90000000) and Brighton (7100000) have to be considered and no other branch has more assets than Downtown Branch. That's why result is null]



The screenshot shows a SQL query window with three tabs: SQL File 3*, SQL File 4*, and SQL File 5*. The query is:

```
1 • use cse370;
2
3 • select branch_name from branch where assets > all (select assets from branch where branch_city = 'Brooklyn');
```

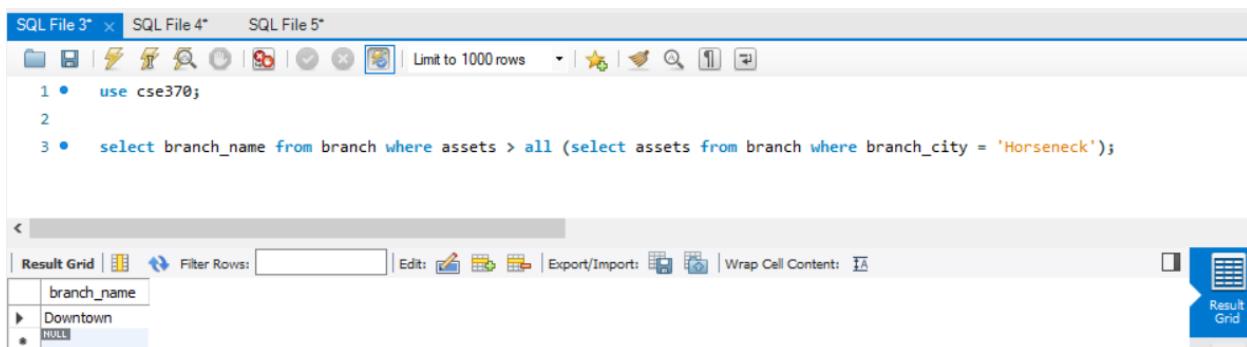
The results grid shows one row with the value 'NULL'.

- **Find the set of Names of branches whose Assets are greater than the Assets of all branches in Horseneck**

```
use cse370;
```

```
select branch_name from branch where assets > all (select assets from branch where branch_city = 'Horseneck');
```

[There are three Branches in Horseneck City Mianus (400000), Perryridge (1700000) and RoundHill (8000000). Since 'all' is used all Branches need to be considered. Only other Branch that has more assets than RoundHill is Downtown Branch]



The screenshot shows a SQL query window with three tabs: SQL File 3*, SQL File 4*, and SQL File 5*. The query is:

```
1 • use cse370;
2
3 • select branch_name from branch where assets > all (select assets from branch where branch_city = 'Horseneck');
```

The results grid shows one row with the value 'Downtown'.

- **Find the set of names of customers at Brighton branch, in alphabetical order**

```
use cse370;
```

```
select customer_name from customer order by customer_name asc;
```

The screenshot shows the MySQL Workbench interface with three tabs at the top: "SQL File 3*", "SQL File 4*", and "SQL File 5*". The SQL editor contains the following code:

```
1 • use cse370;
2
3 • select customer_name from customer order by customer_name asc;
```

The results pane displays a table titled "Result Grid" with the column "customer_name". The data is as follows:

customer_name
Adams
Brooks
Curry
Glenn
Green
Hayes
Johnson
Jones
Lindsay
Smith
Turner
Williams

The right sidebar of the interface includes icons for "Result Grid", "Form Editor", "Field Types", and "Query".

- **Show the loan data, ordered by decreasing Amounts, then increasing Loan numbers**

```
use cse370;
```

```
select loan_number, amount from loan order by amount desc, loan_number asc;
```

The screenshot shows the MySQL Workbench interface with three tabs at the top: "SQL File 3*", "SQL File 4*", and "SQL File 5*". The SQL editor contains the following code:

```
1 • use cse370;
2
3 • select loan_number, amount from loan order by amount desc, loan_number asc;
```

The results pane displays a table titled "Result Grid" with the columns "loan_number" and "amount". The data is as follows:

loan_number	amount
L-23	2000
L-14	1500
L-15	1500
L-16	1300
L-17	1000
L-11	900
L-93	500
*	HULL

The right sidebar of the interface includes icons for "Result Grid", "Form Editor", "Field Types", and "Query".

- **Find the Names of Branches having at least one Account, with average Balances greater than or equal 700**

```
use cse370;
```

```
select branch_name from account group by branch_name having avg(balance) > 700;
```

```
SQL File 3* | SQL File 4* | SQL File 5*
use cse370;
select branch_name from account group by branch_name having avg(balance) > 700;
```

branch_name
Brighton

- **Find the Names and Account number of Customers who have the 3 highest Balances in their Accounts**

```
use cse370;
```

```
select c.customer_name, a.account_number from customer c inner join depositor d on c.customer_id = d.customer_id inner join account a on d.account_number = a.account_number order by a.balance desc limit 0, 3;
```

```
SQL File 3* | SQL File 4* | SQL File 5*
use cse370;
select c.customer_name, a.account_number from customer c inner join depositor d on c.customer_id = d.customer_id inner join account a on d.account_number = a.account_number order by a.balance desc limit 0, 3;
```

customer_name	account_number
Johnson	A-201
Jones	A-217
Smith	A-215

- **Find the Names of Customers with Accounts at a Branch where Johnson has an Account**

```
use cse370;
```

```
select c.customer_name from customer c join depositor d on c.customer_id = d.customer_id
join account a on d.account_number = a.account_number where c.customer_name != 'Johnson'
and a.branch_name in ( select a.branch_name from account a join depositor d on
a.account_number = d.account_number join customer c on c.customer_id = d.customer_id
where customer_name = 'Johnson' );
```

The screenshot shows a SQL query editor with the following details:

- SQL File 3***: The active tab.
- SQL File 4*** and **SQL File 5***: Other tabs.
- Toolbar**: Includes icons for file operations, search, and result export.
- Query Area** (containing the SQL code):


```
1 • use cse370;
2
3 •   select c.customer_name from customer c join depositor d on c.customer_id = d.customer_id
4     join account a on d.account_number = a.account_number where c.customer_name != 'Johnson' and
5       a.branch_name in ( select a.branch_name from account a join depositor d on a.account_number = d.account_number
6         join customer c on c.customer_id = d.customer_id where customer_name = 'Johnson' );
```
- Result Grid** (displayed below the query area):

customer_name
Jones

- **Find the Names of Customers with an Account on Mianus but not a Loan at Mianus branch**

```
use cse370;
```

```
select c.customer_name from customer c join depositor d on c.customer_id = d.customer_id join
account a on d.account_number = a.account_number where a.branch_name = 'Mianus' and
c.customer_id not in ( select c.customer_id from customer c join borrower b on c.customer_id =
b.customer_id join loan l on b.loan_number = l.loan_number where l.branch_name = 'Mianus' );
```

The screenshot shows a SQL query editor with the following details:

- SQL File 3***: The active tab.
- SQL File 4*** and **SQL File 5***: Other tabs.
- Toolbar**: Includes icons for file operations, search, and result export.
- Query Area** (containing the SQL code):


```
1 • use cse370;
2
3 •   select c.customer_name from customer c join depositor d on c.customer_id = d.customer_id join account a on d.account_number =
4     a.account_number where a.branch_name = 'Mianus' and c.customer_id not in ( select c.customer_id from customer c join borrower
5       b on c.customer_id = b.customer_id join loan l on b.loan_number = l.loan_number where l.branch_name = 'Mianus' );
6
```
- Result Grid** (displayed below the query area):

customer_name
Smith

- **Find the Names of each Branch and the number of Customers having at least one Account at that Branch**

```
use cse370;
```

```
select a.branch_name, count(*) as number_of_customer from customer c join depositor d on
c.customer_id = d.customer_id join account a on a.account_number = d.account_number group
by a.branch_name;
```

The screenshot shows the MySQL Workbench interface with three tabs at the top: SQL File 3*, SQL File 4*, and SQL File 5*. The SQL editor contains the following code:

```
1 • use cse370;
2
3 • select a.branch_name, count(*) as number_of_customer from customer c join depositor d on c.customer_id = d.customer_id
4     join account a on a.account_number = d.account_number group by a.branch_name;
5
```

The results grid displays the following data:

branch_name	number_of_customer
Downtown	1
Perryridge	1
Brighton	2
Mianus	1
Redwood	1
Round Hill	1

- **Find the average Balance of all Customers in 'Palo Alto' having at least 2 Accounts**

```
use cse370;
```

```
select avg(a.balance) as average_balance from account a join depositor d on
a.account_number = d.account_number join customer c on d.customer_id = c.customer_id
group by customer_city having customer_city = 'Palo Alto' and count(*) >= 2;
```

The screenshot shows the MySQL Workbench interface with three tabs at the top: SQL File 3*, SQL File 4*, and SQL File 5*. The SQL editor contains the following code:

```
1 • use cse370;
2
3 • select avg(a.balance) as average_balance from account a join depositor d on a.account_number = d.account_number join
4     customer c on d.customer_id = c.customer_id group by customer_city having customer_city = 'Palo Alto' and
5     count(*) >= 2;
6
```

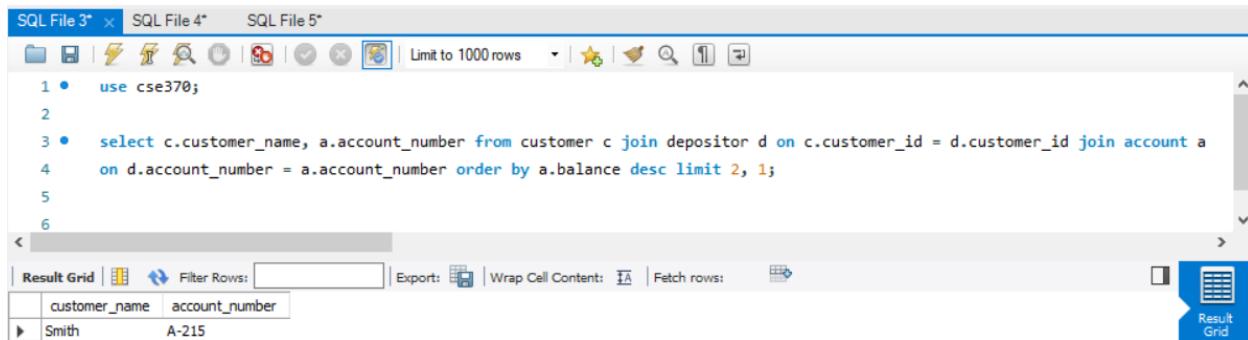
The results grid displays the following data:

average_balance
700.0000

- **Find the Name and Account number of the customer who has the 3rd highest Balance in their Account**

```
use cse370;
```

```
select c.customer_name, a.account_number from customer c join depositor d on c.customer_id = d.customer_id join account a on d.account_number = a.account_number order by a.balance desc limit 2, 1;
```



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

```
1 • use cse370;
2
3 • select c.customer_name, a.account_number from customer c join depositor d on c.customer_id = d.customer_id join account a
4   on d.account_number = a.account_number order by a.balance desc limit 2, 1;
5
6
```

The results pane displays the output of the query:

customer_name	account_number
Smith	A-215

Thank You