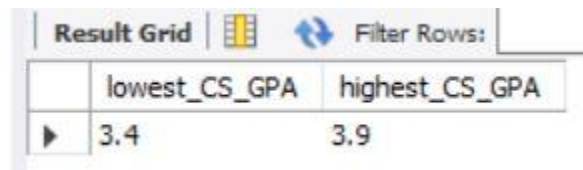


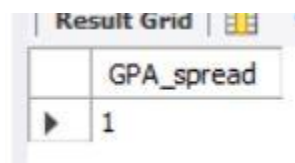
Lab 7 Part 2 Report

- 1.) SELECT MIN(gpa) AS lowest_CS_GPA, MAX(gpa) AS highest_CS_GPA FROM student
WHERE sid IN (SELECT DISTINCT sid FROM apply WHERE major = 'CS');



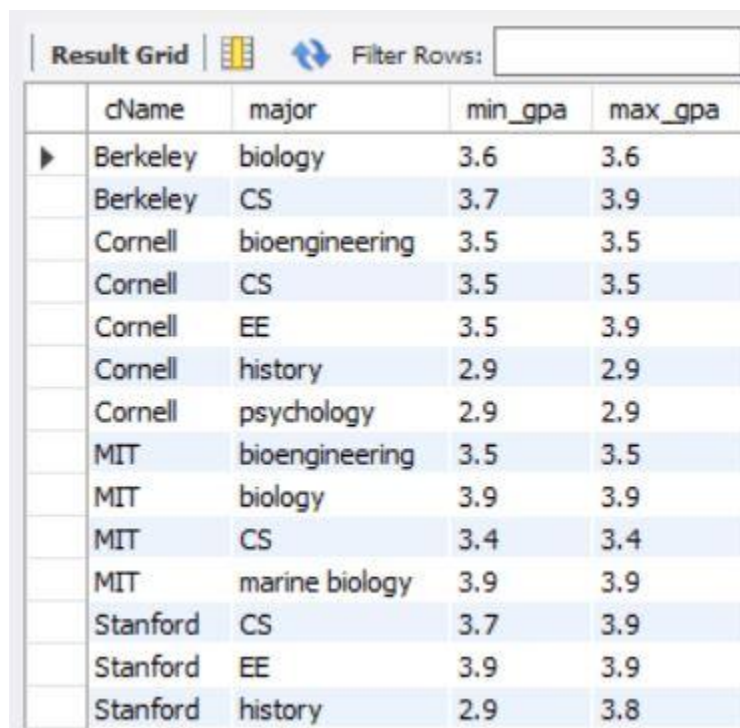
	lowest_CS_GPA	highest_CS_GPA
▶	3.4	3.9

- 2.) SELECT MAX(gpa) - MIN(gpa) AS GPA_spread FROM student;



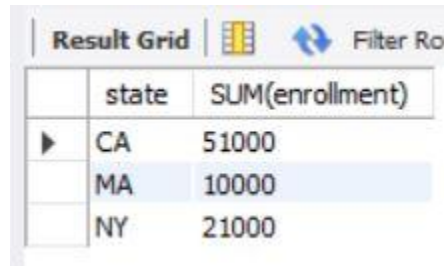
	GPA_spread
▶	1

- 3.) SELECT cName, major, MIN(GPA) AS min_gpa, MAX(GPA) AS max_gpa
FROM student, apply
WHERE student.sid = apply.sid GROUP BY cName, major ORDER BY cName;



	cName	major	min_gpa	max_gpa
▶	Berkeley	biology	3.6	3.6
	Berkeley	CS	3.7	3.9
	Cornell	bioengineering	3.5	3.5
	Cornell	CS	3.5	3.5
	Cornell	EE	3.5	3.9
	Cornell	history	2.9	2.9
	Cornell	psychology	2.9	2.9
	MIT	bioengineering	3.5	3.5
	MIT	biology	3.9	3.9
	MIT	CS	3.4	3.4
	MIT	marine biology	3.9	3.9
	Stanford	CS	3.7	3.9
	Stanford	EE	3.9	3.9
	Stanford	history	2.9	3.8

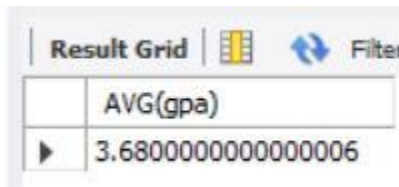
4.) SELECT state, SUM(enrollment) FROM college GROUP BY state;



A screenshot of a database result grid. The title bar says 'Result Grid' and 'Filter Rows'. The grid has two columns: 'state' and 'SUM(enrollment)'. There are three rows of data: CA with 51000, MA with 10000, and NY with 21000. The MA row is highlighted in blue.

	state	SUM(enrollment)
▶	CA	51000
	MA	10000
	NY	21000

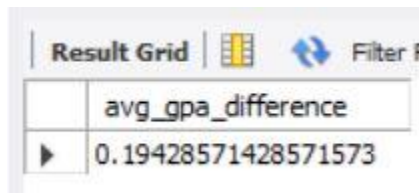
5.) SELECT AVG(gpa) FROM student
JOIN (SELECT DISTINCT sID FROM apply WHERE major = 'CS') AS CS_GPAs
ON student.sID = CS_GPAs.sID;



A screenshot of a database result grid. The title bar says 'Result Grid' and 'Filter'. The grid has one column: 'AVG(gpa)'. There is one row of data: 3.6800000000000006.

	AVG(gpa)
▶	3.6800000000000006

6.) SELECT CS_GPA.avgGPA - NotCS_GPA.avgGPA
FROM (SELECT AVG(GPA) AS avgGPA FROM student WHERE sID in (SELECT sID
FROM apply WHERE major = 'CS')) AS CS_GPA,
(SELECT AVG(GPA) as avgGPA FROM student WHERE sID not in (SELECT sID FROM
apply WHERE major = 'CS')) AS NotCS_GPA;



A screenshot of a database result grid. The title bar says 'Result Grid' and 'Filter F'. The grid has one column: 'avg_gpa_difference'. There is one row of data: 0.19428571428571573.

	avg_gpa_difference
▶	0.19428571428571573

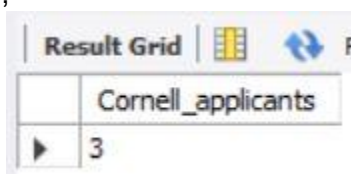
7.) SELECT COUNT(*) AS numRows FROM student;



A screenshot of a database result grid. The title bar says 'Result Grid'. The grid has one column: 'numRows'. There is one row of data: 12.

	numRows
▶	12

8.) SELECT COUNT(DISTINCT sID) AS Cornell_applicants FROM apply
WHERE cName = 'Cornell';



A screenshot of a database result grid. The title bar says 'Result Grid' and 'F'. The grid has one column: 'Cornell_applicants'. There is one row of data: 3.

	Cornell_applicants
▶	3

9a.) SELECT sID, COUNT(DISTINCT cName) AS numCollegesAppliedTo FROM apply
GROUP BY sID;

Result Grid			Filter Rows:
	sID	numCollegesAppliedTo	
▶	123	3	
	234	1	
	345	2	
	543	1	
	678	1	
	765	2	
	876	2	
	987	2	

9b.) SELECT student.sID, sName, COUNT(DISTINCT cName) AS numCollegesAppliedTo
FROM student, apply WHERE student.sID = apply.sID GROUP BY sID, sName;

Result Grid				Filter Rows:
	sID	sName	numCollegesAppliedTo	
▶	123	Amy	3	
	234	Bob	1	
	345	Craig	2	
	543	Craig	1	
	678	Fay	1	
	765	Jay	2	
	876	Irene	2	
	987	Helen	2	

10.) SELECT cName FROM apply GROUP BY cName HAVING COUNT(DISTINCT sID) < 5;

Result Grid		Filter Rows:
	cName	
▶	Berkeley	
	Cornell	
	MIT	

11.)

Results for A (SELECT statement before the INSERT statements):

```
53 #Question 11
54 #SQL Script from part A:
55 • SELECT * FROM student;
56 • INSERT INTO student VALUES (432, 'Kevin', null, 1500);
57 • INSERT INTO student VALUES (321, 'Lori', null, 2500);
58 • SELECT * FROM student;
```

	sID	sName	GPA	sizeHS
▶	123	Amy	3.9	1000
	234	Bob	3.6	1500
	345	Craig	3.5	500
	456	Doris	3.9	1000
	567	Edward	2.9	2000
	678	Fay	3.8	200
	789	Gary	3.4	800
	987	Helen	3.7	800
	876	Irene	3.9	400
	765	Jay	2.9	1500
	654	Amy	3.9	1000
	543	Craig	3.4	2000


Results for A (SELECT statement after the INSERT statements):

```
53 #Question 11
54 #SQL Script from part A:
55 • SELECT * FROM student;
56 • INSERT INTO student VALUES (432, 'Kevin', null, 1500);
57 • INSERT INTO student VALUES (321, 'Lori', null, 2500);
58 • SELECT * FROM student;
```


	sID	sName	GPA	sizeHS
▶	123	Amy	3.9	1000
	234	Bob	3.6	1500
	345	Craig	3.5	500
	456	Doris	3.9	1000
	567	Edward	2.9	2000
	678	Fay	3.8	200
	789	Gary	3.4	800
	987	Helen	3.7	800
	876	Irene	3.9	400
	765	Jay	2.9	1500
	654	Amy	3.9	1000
	543	Craig	3.4	2000
	432	Kevin	NULL	1500
	321	Lori	NULL	2500

After the INSERT statements ran, two new rows were added to the Student table, and they were added to the end of the table.

Results for B:

62	#SQL Script from part B:
63	• <code>SELECT COUNT(*) FROM student;</code>
<hr/>	
Result Grid  Filter Rows: <input type="text"/>	
	COUNT(*)
▶	14


Results for C:

65	#SQL Script from part C:
66	• <code>SELECT COUNT(GPA) FROM student;</code>
<hr/>	
Result Grid  Filter Rows: <input type="text"/>	
	COUNT(GPA)
▶	12

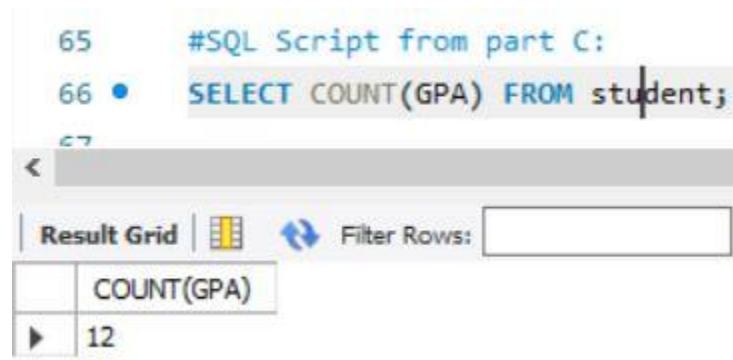
The COUNT result shown in B is 14, while the COUNT result shown in C is 12. The difference is caused by what the COUNT function is told to count. COUNT(*) counts all rows in the table, even the rows that have null values. COUNT(GPA), meanwhile, counts all the rows in the GPA column, but does not count null values. Since the inserted rows both have the GPA value set to NULL, these rows are not counted.

12.) The statement in D deletes any rows in the Student table where GPA is null, so the two rows added in A are deleted.

Results for B (after deletion):

62	#SQL Script from part B:
63	• <code>SELECT COUNT(*) FROM student;</code>
64	
<hr/>	
Result Grid  Filter Rows: <input type="text"/>	
	COUNT(*)
▶	12

Results of C (after deletion):



The screenshot shows a SQL script editor with the following content:

```
65 #SQL Script from part C:  
66 • SELECT COUNT(GPA) FROM student;  
67
```

Below the script, there is a "Result Grid" tab and a "Filter Rows" input field. The result grid displays the following data:

	COUNT(GPA)
▶	12

The result of C is the same, since it wasn't counting the null rows before. However, the results of B are different. Instead of 14, it is now 12, the same as the C results. This is because the two null rows have been removed, so there are now 12 rows instead of 14.