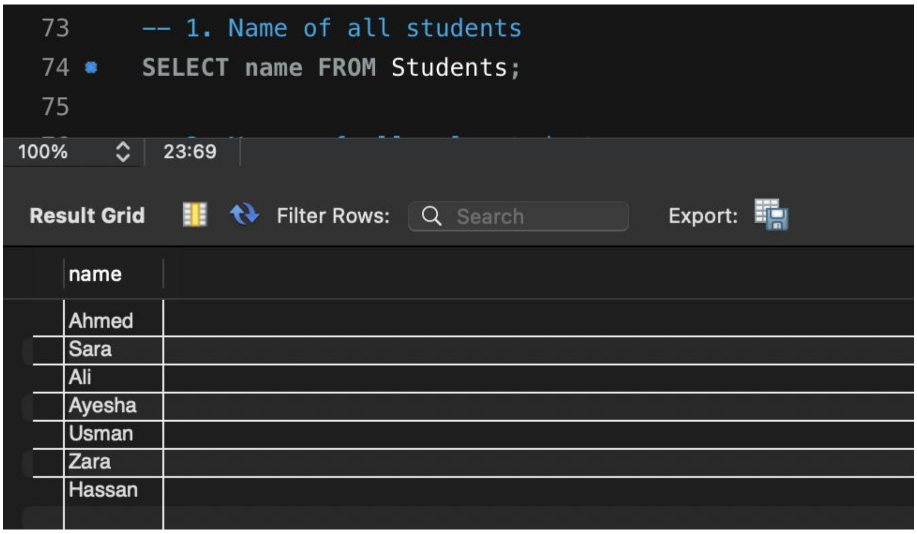
�� Student Management System-Documentation

�� 1. Get names of all students

Description: Fetches all student names from the `Students` table.

Screenshot below:



```sql
-- 1. Name of all students
SELECT name FROM Students;
```

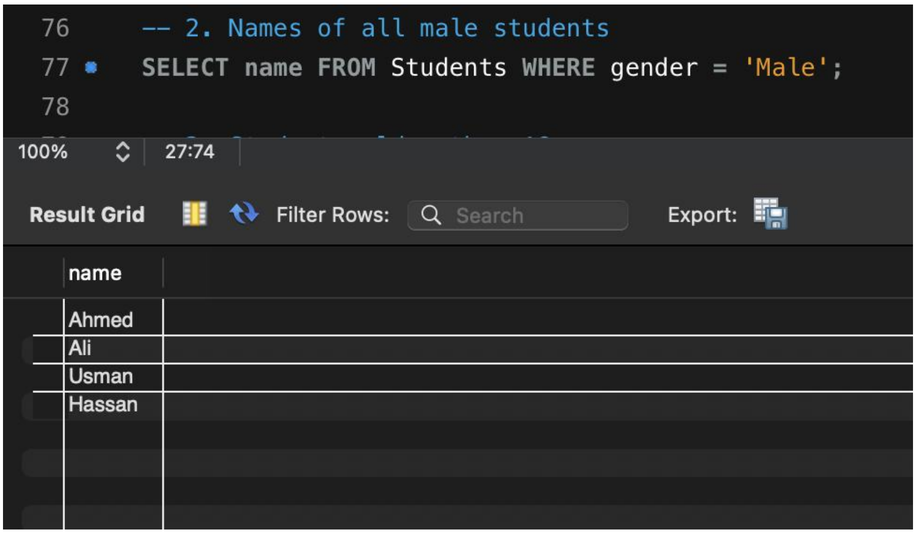
name
Ahmed
Sara
Ali
Ayesha
Usman
Zara
Hassan

Working: Selects and displays only the name column from the Students table.

�� 2. Get names of all male students

Description: Filters and shows names of students whose gender is 'Male'.

Screenshot below:



```sql
-- 2. Names of all male students
SELECT name FROM Students WHERE gender = 'Male';
```

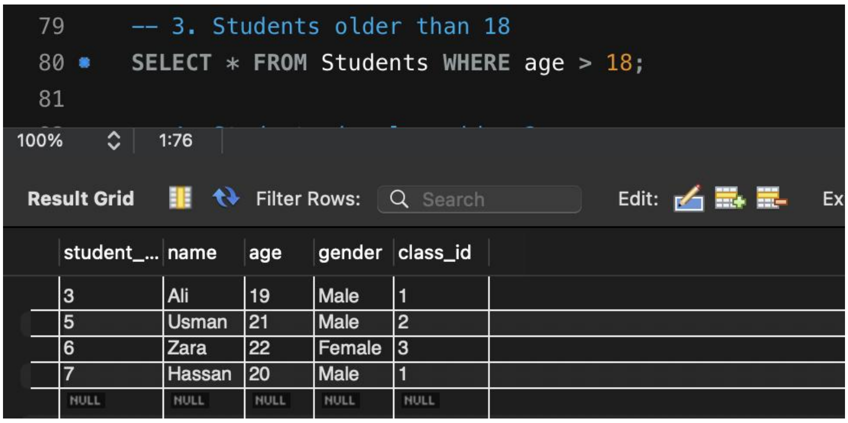
name
Ahmed
Ali
Usman
Hassan

Working: Filters records where gender = 'Male' and returns their names.

�� 3. Find all students older than 18 years

Description: Displays student details where age is greater than 18.

Screenshot below:



```sql
-- 3. Students older than 18
SELECT * FROM Students WHERE age > 18;
```

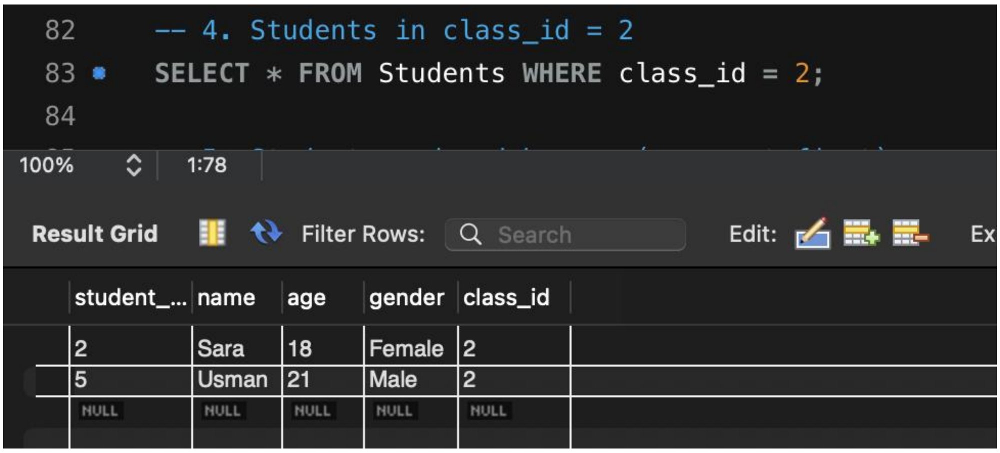
student_...	name	age	gender	class_id
3	Ali	19	Male	1
5	Usman	21	Male	2
6	Zara	22	Female	3
7	Hassan	20	Male	1
NULL	NULL	NULL	NULL	NULL

Working: Returns full student details where age is more than 18.

�� 4. Get details of students in class_id = 2

Description: Lists all fields for students assigned to class ID 2.

Screenshot below:



```sql
-- 4. Students in class_id = 2
SELECT * FROM Students WHERE class_id = 2;
```

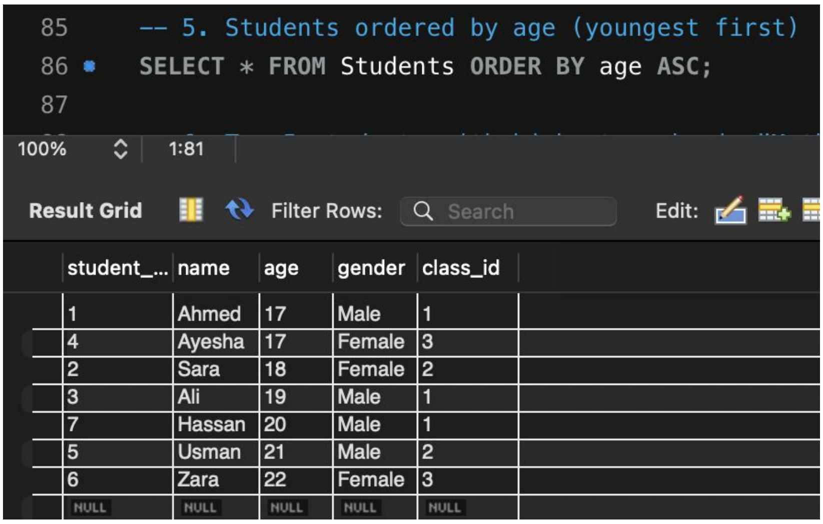
student_...	name	age	gender	class_id
2	Sara	18	Female	2
5	Usman	21	Male	2
NULL	NULL	NULL	NULL	NULL

Working: Shows all student data where the class_id is 2.

�� 5. List all students ordered by age, youngest first

Description: Sorts students by age in ascending order.

Screenshot below:



```sql
-- 5. Students ordered by age (youngest first)
SELECT * FROM Students ORDER BY age ASC;
```

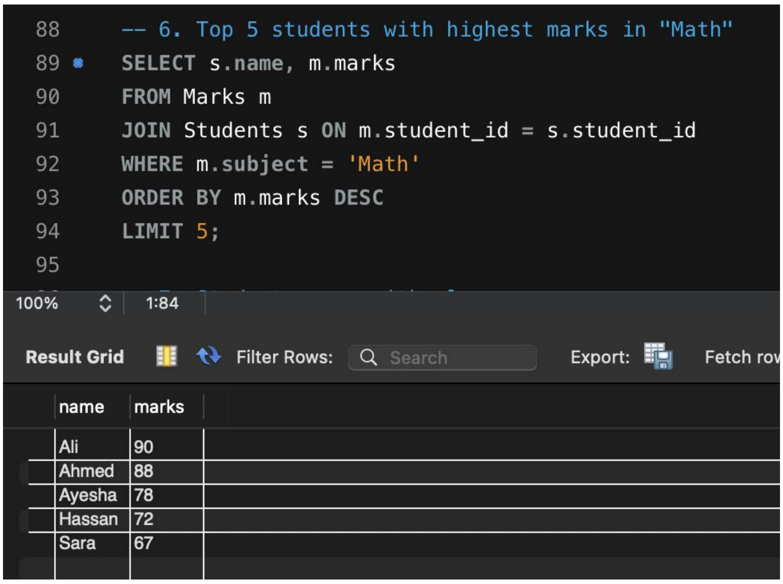
student_...	name	age	gender	class_id
1	Ahmed	17	Male	1
4	Ayesha	17	Female	3
2	Sara	18	Female	2
3	Ali	19	Male	1
7	Hassan	20	Male	1
5	Usman	21	Male	2
6	Zara	22	Female	3
NULL	NULL	NULL	NULL	NULL

Working: Selects and displays only the name column from the Students table.

�� 6. Show top 5 students with highest marks in "Math"

Description: Returns student names and marks who scored highest in Math.

Screenshot below:



```sql
-- 6. Top 5 students with highest marks in "Math"
SELECT s.name, m.marks
FROM Marks m
JOIN Students s ON m.student_id = s.student_id
WHERE m.subject = 'Math'
ORDER BY m.marks DESC
LIMIT 5;
```

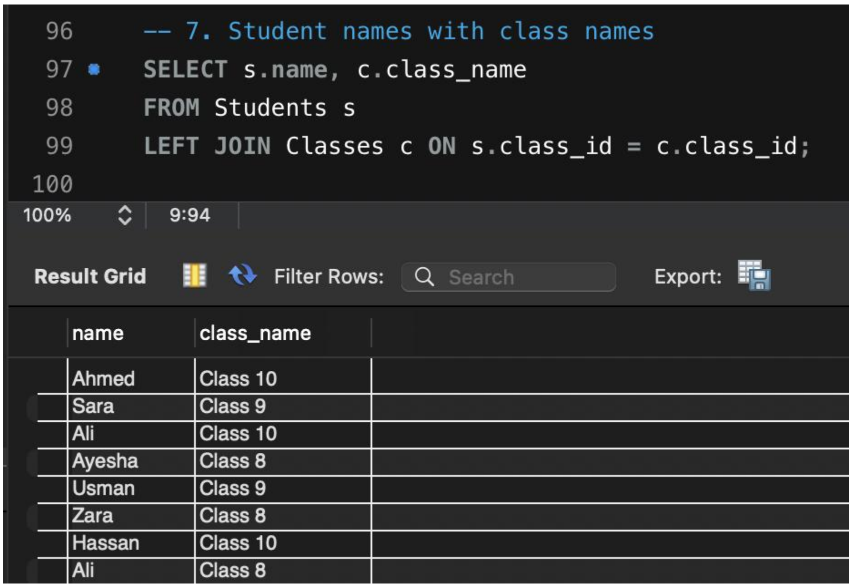
name	marks
Ali	90
Ahmed	88
Ayesha	78
Hassan	72
Sara	67

Working: Selects and displays only the name column from the Students table.

�� 7. List student names along with their class names

Description: Joins `Students` and `Classes` to show student names and their class.

Screenshot below:



```
-- 7. Student names with class names
SELECT s.name, c.class_name
FROM Students s
LEFT JOIN Classes c ON s.class_id = c.class_id;
```

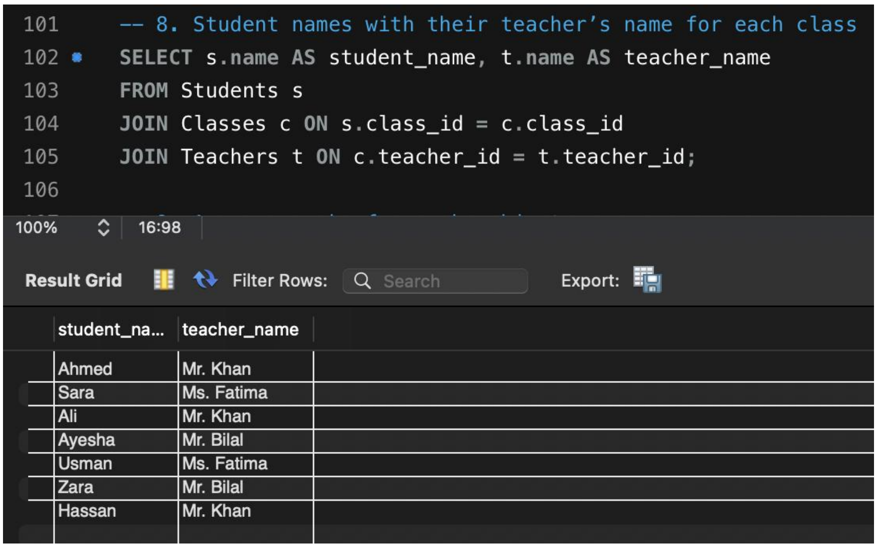
name	class_name
Ahmed	Class 10
Sara	Class 9
Ali	Class 10
Ayesha	Class 8
Usman	Class 9
Zara	Class 8
Hassan	Class 10
Ali	Class 8

Working: Selects and displays only the name column from the Students table.

�� 8. Show student names with their teacher’ s name

Description: Joins `Students`, `Classes`, and `Teachers` to show student and assigned teacher.

Screenshot below:



```
-- 8. Student names with their teacher’s name for each class
SELECT s.name AS student_name, t.name AS teacher_name
FROM Students s
JOIN Classes c ON s.class_id = c.class_id
JOIN Teachers t ON c.teacher_id = t.teacher_id;
```

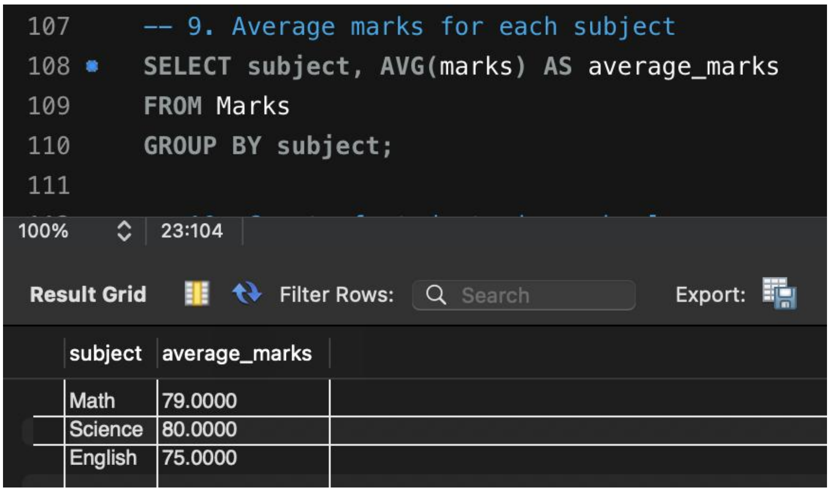
student_na...	teacher_name
Ahmed	Mr. Khan
Sara	Ms. Fatima
Ali	Mr. Khan
Ayesha	Mr. Bilal
Usman	Ms. Fatima
Zara	Mr. Bilal
Hassan	Mr. Khan

Working: Selects and displays only the name column from the Students table.

�� 9. Find the average marks for each subject

Description: Groups data by subject and calculates average marks.

Screenshot below:



```
-- 9. Average marks for each subject
SELECT subject, AVG(marks) AS average_marks
FROM Marks
GROUP BY subject;
```

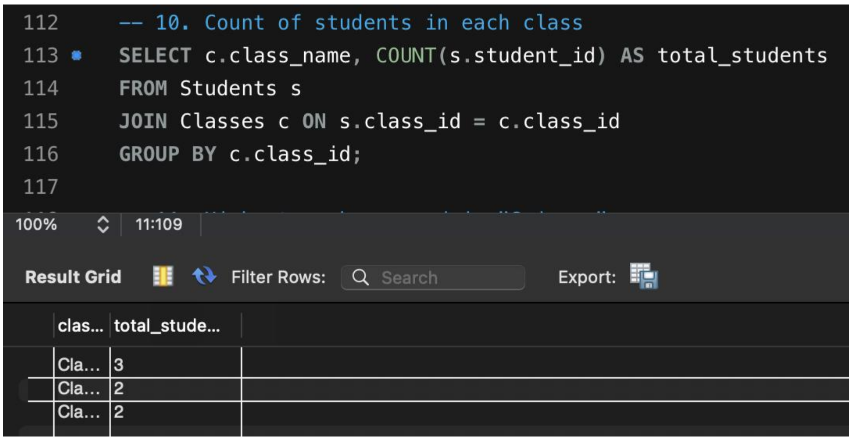
subject	average_marks
Math	79.0000
Science	80.0000
English	75.0000

Working: Selects and displays only the name column from the Students table.

�� 10. Count how many students are in each class

Description: Groups students by class and returns count.

Screenshot below:



```
-- 10. Count of students in each class
SELECT c.class_name, COUNT(s.student_id) AS total_students
FROM Students s
JOIN Classes c ON s.class_id = c.class_id
GROUP BY c.class_id;
```

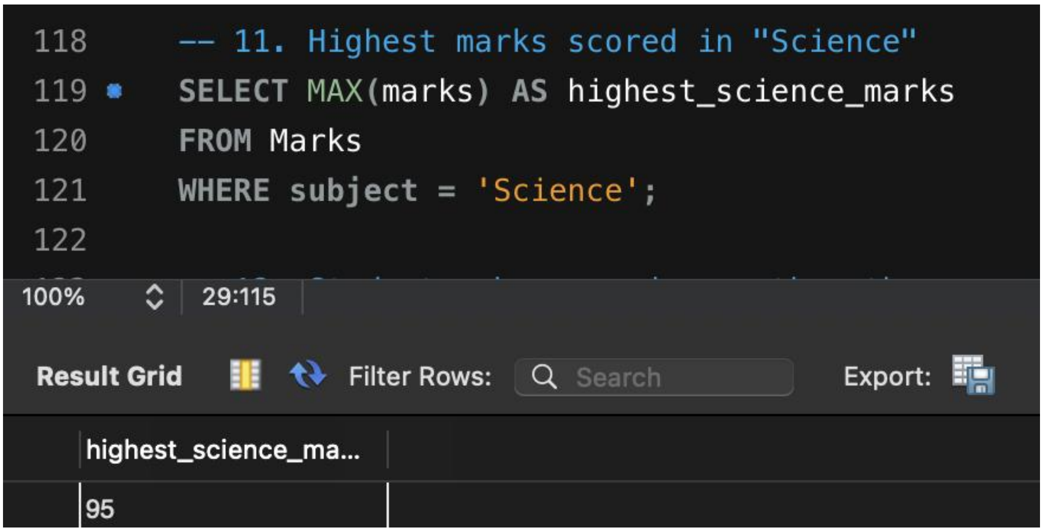
clas...	total_stude...
Cla...	3
Cla...	2
Cla...	2

Working: Selects and displays only the name column from the Students table.

�� 11. Find the highest marks scored in "Science"

Description: Fetches the maximum marks from subject 'Science'.

Screenshot below:



```sql
-- 11. Highest marks scored in "Science"
SELECT MAX(marks) AS highest_science_marks
FROM Marks
WHERE subject = 'Science';
```

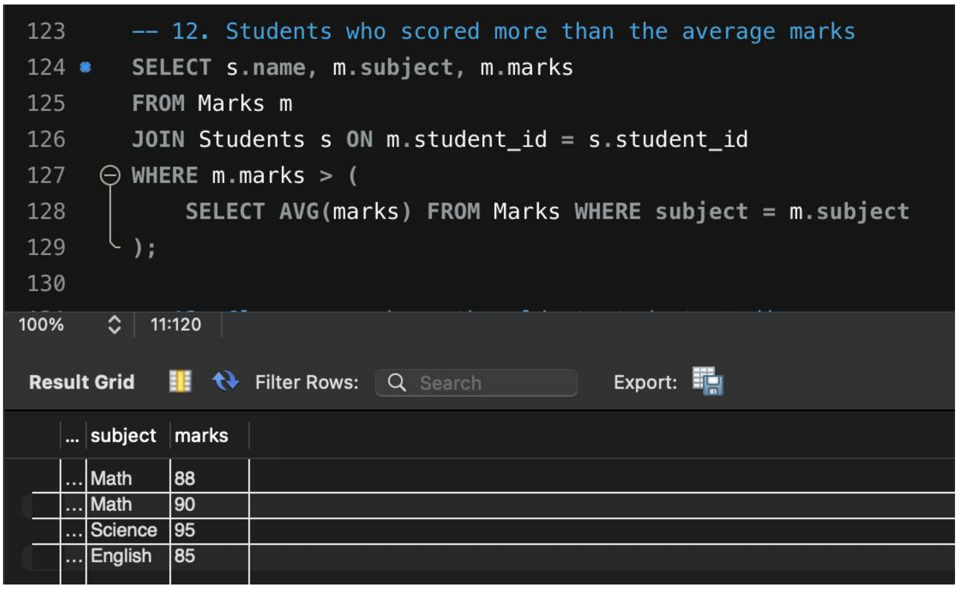
highest_science_ma...
95

Working: Selects and displays only the name column from the Students table.

�� 12. List names of students who scored more than the average marks

Description: Uses a subquery to compare marks above average per subject.

Screenshot below:



```sql
-- 12. Students who scored more than the average marks
SELECT s.name, m.subject, m.marks
FROM Marks m
JOIN Students s ON m.student_id = s.student_id
WHERE m.marks > (
    SELECT AVG(marks) FROM Marks WHERE subject = m.subject
);
```

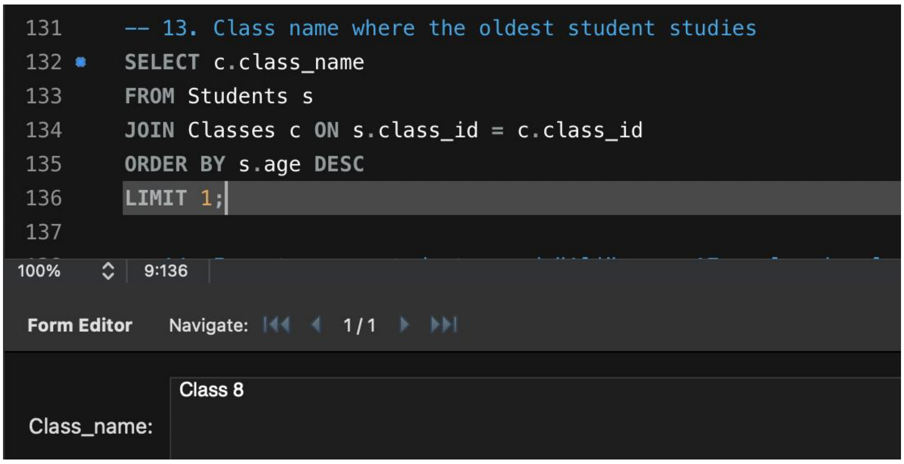
...	subject	marks
...	Math	88
...	Math	90
...	Science	95
...	English	85

Working: Selects and displays only the name column from the Students table.

�� 13. Find the class name where the oldest student studies

Description: Orders students by age descending and picks the top class name.

Screenshot below:



```sql
-- 13. Class name where the oldest student studies
SELECT c.class_name
FROM Students s
JOIN Classes c ON s.class_id = c.class_id
ORDER BY s.age DESC
LIMIT 1;
```

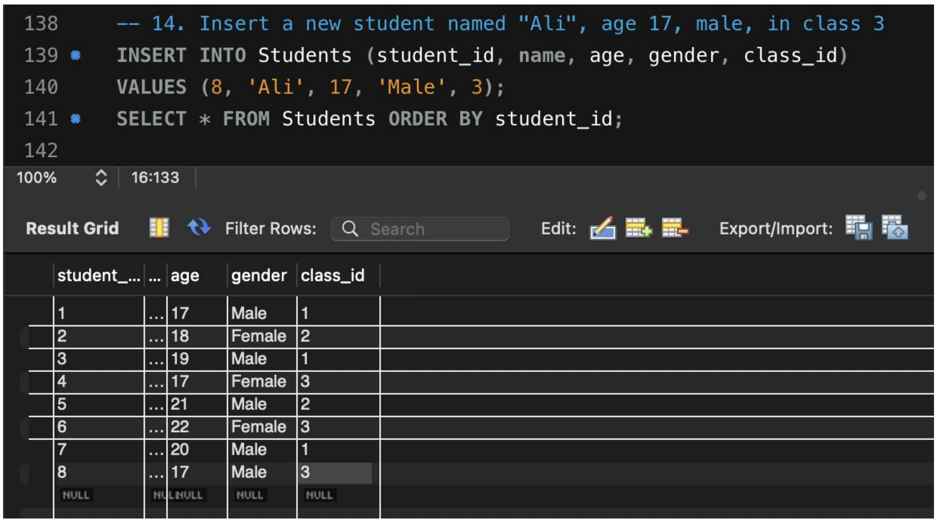
Class_name: Class 8

Working: Selects and displays only the name column from the Students table.

�� 14. Insert a new student named "Ali", age 17, male, in class 3

Description: Adds a new record to `Students`.

Screenshot below:



```sql
-- 14. Insert a new student named "Ali", age 17, male, in class 3
INSERT INTO Students (student_id, name, age, gender, class_id)
VALUES (8, 'Ali', 17, 'Male', 3);
SELECT * FROM Students ORDER BY student_id;
```

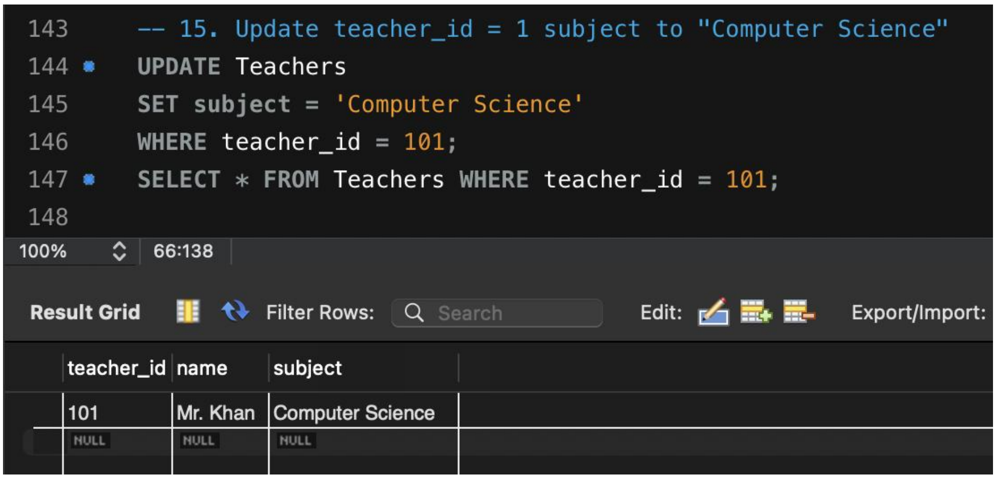
student_...	...	age	gender	class_id
1	...	17	Male	1
2	...	18	Female	2
3	...	19	Male	1
4	...	17	Female	3
5	...	21	Male	2
6	...	22	Female	3
7	...	20	Male	1
8	...	17	Male	3
NULL	NULL	NULL	NULL	NULL

Working: Selects and displays only the name column from the Students table.

�� 15. Update the subject of teacher with teacher_id = 101

Description: Modifies the subject of a teacher to "Computer Science".

Screenshot below:



```
-- 15. Update teacher_id = 1 subject to "Computer Science"
UPDATE Teachers
SET subject = 'Computer Science'
WHERE teacher_id = 101;
SELECT * FROM Teachers WHERE teacher_id = 101;

```

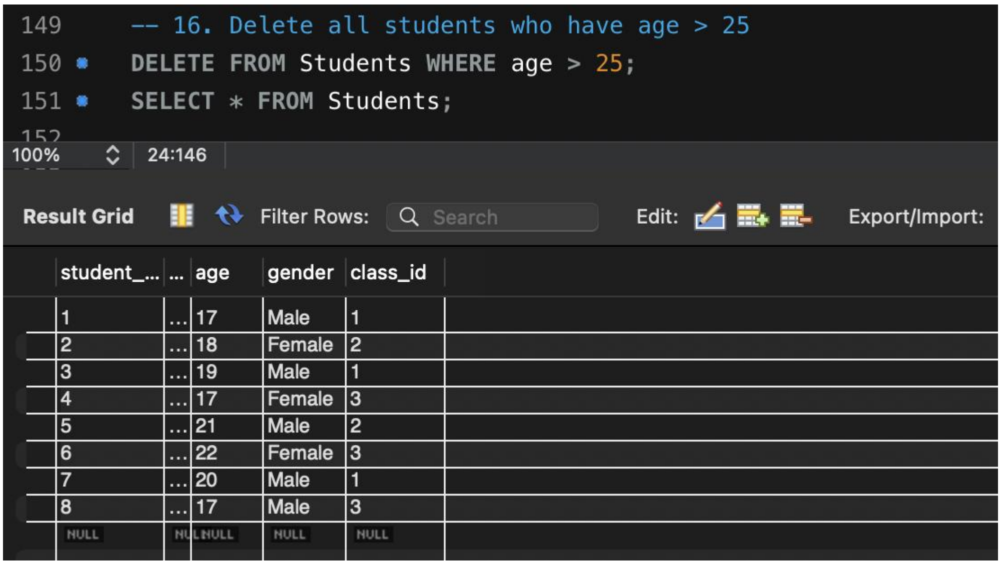
teacher_id	name	subject
101	Mr. Khan	Computer Science
NULL	NULL	NULL

Working: Selects and displays only the name column from the Students table.

�� 16. Delete all students who have age > 25

Description: Deletes student records where age exceeds 25.

Screenshot below:



```
-- 16. Delete all students who have age > 25
DELETE FROM Students WHERE age > 25;
SELECT * FROM Students;
```

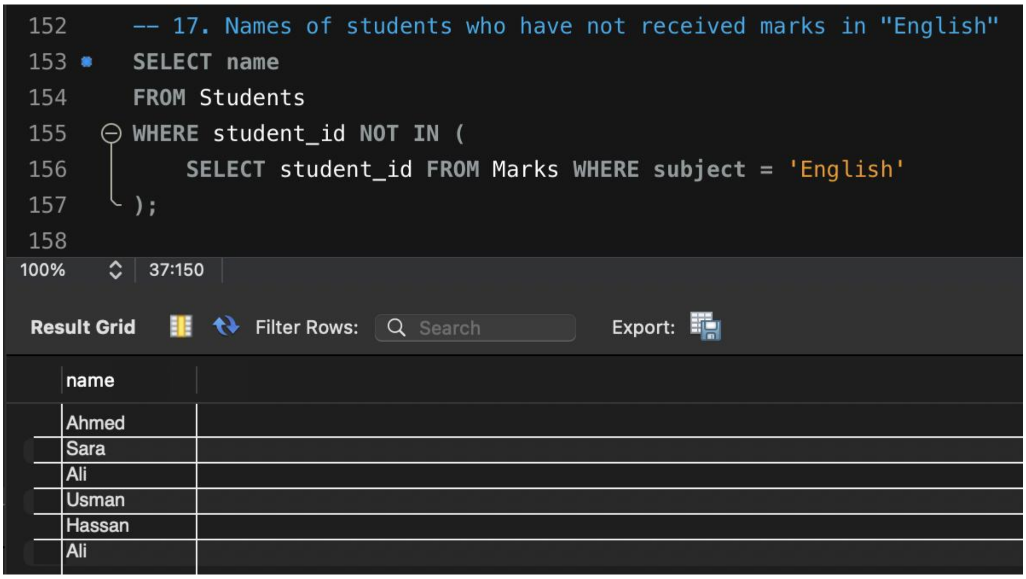
student_...	...	age	gender	class_id
1	...	17	Male	1
2	...	18	Female	2
3	...	19	Male	1
4	...	17	Female	3
5	...	21	Male	2
6	...	22	Female	3
7	...	20	Male	1
8	...	17	Male	3
NULL	NU	NULL	NULL	NULL

Working: Selects and displays only the name column from the Students table.

�� 17. Get names of students who have not received marks in "English"

Description: Finds students who aren’t in the English subject marks table.

Screenshot below:



```sql
-- 17. Names of students who have not received marks in "English"
SELECT name
FROM Students
WHERE student_id NOT IN (
    SELECT student_id FROM Marks WHERE subject = 'English'
);
```

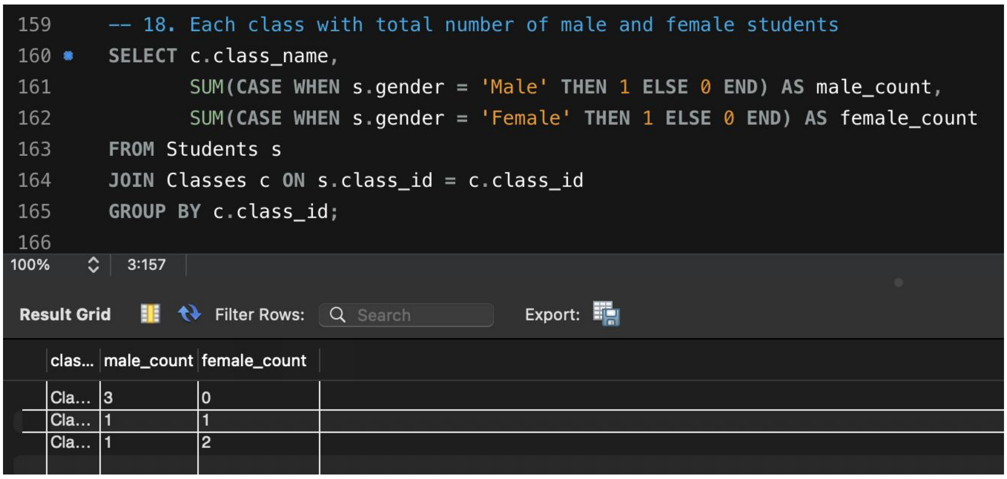
name
Ahmed
Sara
Ali
Usman
Hassan
Ali

Working: Selects and displays only the name column from the Students table.

�� 18. Display class name with male and female student count

Description: Uses `CASE WHEN` with `GROUP BY` for gender-wise counts per class.

Screenshot below:



```sql
-- 18. Each class with total number of male and female students
SELECT c.class_name,
       SUM(CASE WHEN s.gender = 'Male' THEN 1 ELSE 0 END) AS male_count,
       SUM(CASE WHEN s.gender = 'Female' THEN 1 ELSE 0 END) AS female_count
FROM Students s
JOIN Classes c ON s.class_id = c.class_id
GROUP BY c.class_id;
```

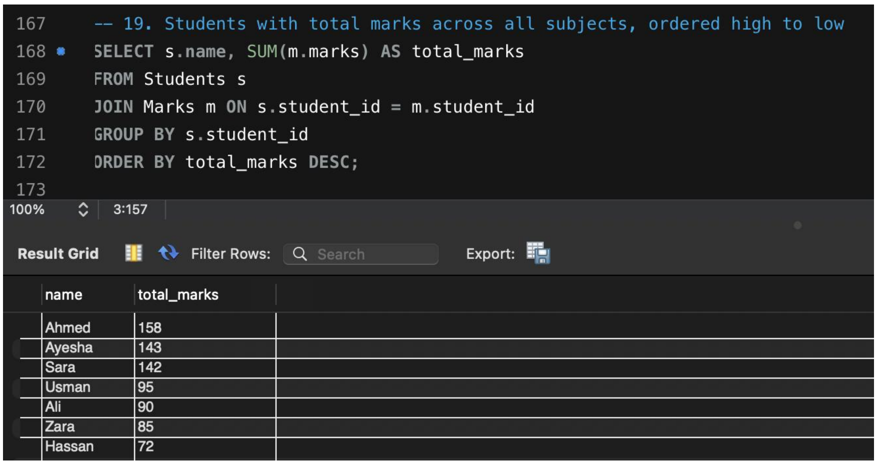
clas...	male_count	female_count
Cla...	3	0
Cla...	1	1
Cla...	1	2

Working: Selects and displays only the name column from the Students table.

�� 19. Get total marks for each student ordered from high to low

Description: Sums marks for each student and orders them descending.

Screenshot below:



```
-- 19. Students with total marks across all subjects, ordered high to low
SELECT s.name, SUM(m.marks) AS total_marks
FROM Students s
JOIN Marks m ON s.student_id = m.student_id
GROUP BY s.student_id
ORDER BY total_marks DESC;
```

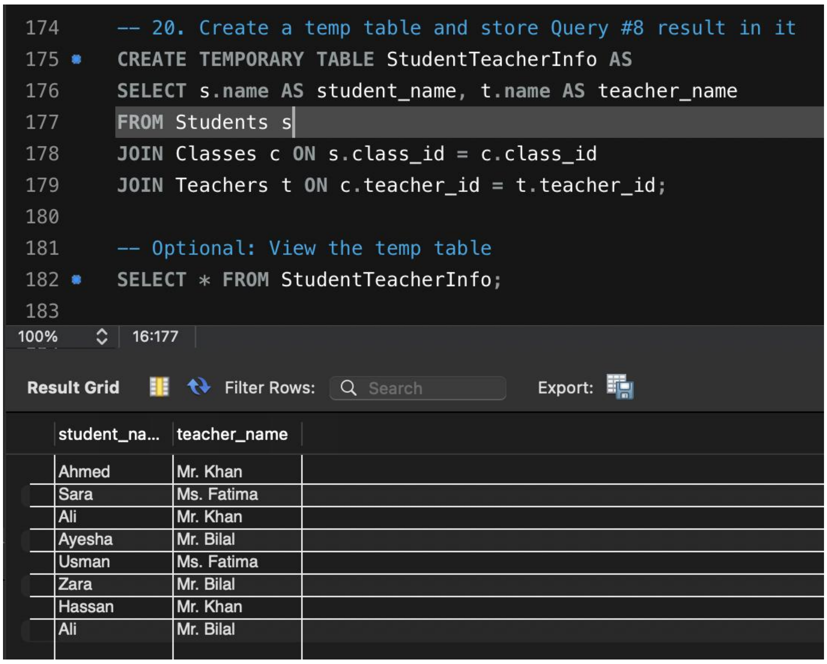
name	total_marks
Ahmed	158
Ayesha	143
Sara	142
Usman	95
Ali	90
Zara	85
Hassan	72

Working: Selects and displays only the name column from the Students table.

�� 20. Create a temporary table with student-teacher pairs

Description: Stores the output of Query #8 in a temporary table.

Screenshot below:



```
-- 20. Create a temp table and store Query #8 result in it
CREATE TEMPORARY TABLE StudentTeacherInfo AS
SELECT s.name AS student_name, t.name AS teacher_name
FROM Students s
JOIN Classes c ON s.class_id = c.class_id
JOIN Teachers t ON c.teacher_id = t.teacher_id;

-- Optional: View the temp table
SELECT * FROM StudentTeacherInfo;
```

student_na...	teacher_name
Ahmed	Mr. Khan
Sara	Ms. Fatima
Ali	Mr. Khan
Ayesha	Mr. Bilal
Usman	Ms. Fatima
Zara	Mr. Bilal
Hassan	Mr. Khan
Ali	Mr. Bilal

Working: Selects and displays only the name column from the Students table.