PERIYAR MANIAMMAI INSTITUTE OF SCIENCE & TECHNOLOGY • PMIST •

PERIYAR MANIAMMAI

INSTITUTE OF SCIENCE & TECHNOLOGY

(Deemed to be University)

Established Under Sec. 3 of UGC Act, 1956 • NAAC Accredited

think • innovate • transform

Mini Project

NAME	: HARIDHARAN K (122012012855)
	VISHNU J (122012012894)
	IFFAN A (123012063254)
CLASS	: III – B.Tech CSE – ‘A’
COURSE CODE	: XCS507
COURSE NAME	: DATABASE MANAGEMENT SYSTEM LABORATORY
TOPIC	: STUDENTS INFORMATION SYSTEM

Submitted To

Ms.S.GOMATHI AP/CSE

Students Information System Documentation

1. Introduction

Student information system means the information system for maintaining and providing student information. Some of them are paper based; heavy manual work is involved in managing and maintaining information such as student personal records. However, recently, most schools maintain the record in database. In this system, the user can add, edit, search, delete and view student’s information. All the records are listed below in the system which makes the user, easy to view the information of each and every student. This system is easy to operate and understand by the user.

2. Entities and Attributes

2.1 Login

Attributes:

- user_id (INT, Primary Key): Unique identifier for each user.
- username (VARCHAR(50), UNIQUE): Username for logging in.
- password (VARCHAR(255)): Encrypted password.
- user_role (ENUM): User role or access level (e.g., 'admin' or 'user').

SQL Definition:

```
CREATE TABLE Users (
   user_id INT AUTO_INCREMENT PRIMARY KEY ,
   username VARCHAR(50) UNIQUE NOT NULL ,
   password VARCHAR(255) NOT NULL ,
   user_role ENUM(‘staff ’, ‘student’)
);
```

2.2 Student

Attributes:

- student_id (INT, Primary Key): Unique identifier for each student.
- name (VARCHAR(100)): Full name of the student.
- dob (DATE): Date of birth of the student.
- gender (ENUM): Gender of the student, options 'Male', 'Female', 'Other'.
- address (VARCHAR(255)): Address of the student.
- phone (VARCHAR(15)): Phone number of the student.
- email (VARCHAR(50)): Email address of the student.

SQL Definition:

```
CREATE TABLE Student (
   student_id INT PRIMARY KEY ,
   name VARCHAR(50) ,
   dob DATE ,
   gender ENUM(‘Male, ‘Female’, ‘Others’) ,
   address VARCHAR(255) ,
   phone VARCHAR(15) ,
   email VARCHAR(50)
);
```

2.3 Fees

Attributes:

- fee_id (INT, Primary Key): Unique identifier for each fee record.
- student_id (INT, Foreign Key): References Student(student_id).
- total_fee (DECIMAL(10, 2)): Total fee amount for Student.
- paid_amount (DECIMAL(10, 2)): Amount paid by Student.
- due_date (DATE): Due date for the payment.
- status (ENUM): Fee status (e.g., 'Paid', 'Unpaid').

SQL Definition:

```
CREATE TABLE Fees (
   fee_id INT PRIMARY KEY,
   student_id INT NOT NULL,
   total_fee DECIMAL(10, 2),
   paid_amount DECIMAL(10, 2),
   due_date DATE,
   status ENUM('Paid', 'Unpaid') DEFAULT 'Unpaid',
   FOREIGN KEY (student_id) REFERENCES Students(student_id)
);
```

2.4 Marks

Attributes:

- mark_id (INT, Primary Key): Unique identifier for each mark entry.
- student_id (INT, Foreign Key): References Students(student_id).
- tamil_marks (INT): Tamil marks obtained by the student.
- english_marks (INT): English marks obtained by the student.
- maths_marks (INT): Maths marks obtained by the student.
- science_marks (INT): Science marks obtained by the student.
- social_marks (INT): Social marks obtained by the student.
- total_marks (INT): Total marks obtained by the student.
- grade (CHAR(1)): Grade obtained by the student in subject.

SQL Definition:

```
CREATE TABLE Marks (
   mark_id INT PRIMARY KEY,
   student_id INT NOT NULL,
   tamil_marks INT CHECK (marks_obtained BETWEEN 0 AND 100),
   english_marks INT CHECK (marks_obtained BETWEEN 0 AND 100),
   maths_marks INT CHECK (marks_obtained BETWEEN 0 AND 100),
   science_marks INT CHECK (marks_obtained BETWEEN 0 AND 100),
   social_marks INT CHECK (marks_obtained BETWEEN 0 AND 100),
   total_marks INT,
   grade CHAR(1) ,
   FOREIGN KEY (student_id) REFERENCES Students(student_id) ON DELETE CASCADE
);
```

3. Relationships for Student Information System

3.1 Student to Fees

- **Relationship**: Each student can have multiple fee records, but each fee record is associated with only one student.
- **Foreign Key**: Fees.student_id references Student.student_id

Explanation: This relationship allows each Student entry to be linked with multiple Fees entries, which record the fees paid, total fees, and due amounts for each student over time.

3.2 Student to Marks

- **Relationship**: Each student can have multiple marks entries for different subjects, but each marks entry is linked to only one student.
- **Foreign Key**: Marks.student_id references Student.student_id

Explanation: This relationship connects each Student with multiple Marks entries, allowing the system to record individual subject scores and grades for each student.

3.3 Login for User Authentication

Relationship: The Login table is used for managing system access and does not have direct relationships with other tables in this schema.

Explanation: This table manages user credentials, allowing access control to the system. user_role defines the type of access each user has, either as an 'admin' or a 'user'.

Summary of Relationships :

Table	**Related Table**	**Relational Type**	**Foreign Key**
Student	Fees	One-to-Many	Fees.student_id
Student	Marks	One-to-Many	marks.student_id
Login	--	Standalone	--

4. ER Diagram:

The ER diagram visually represents the entities and their relationships within the Library Management System.

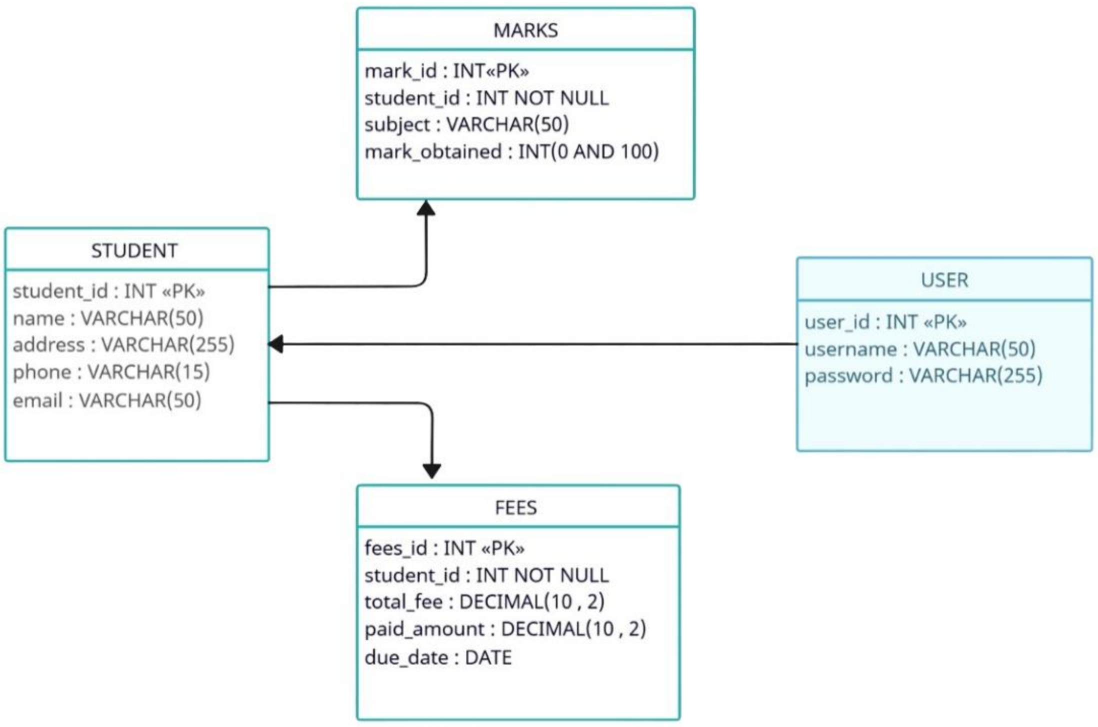
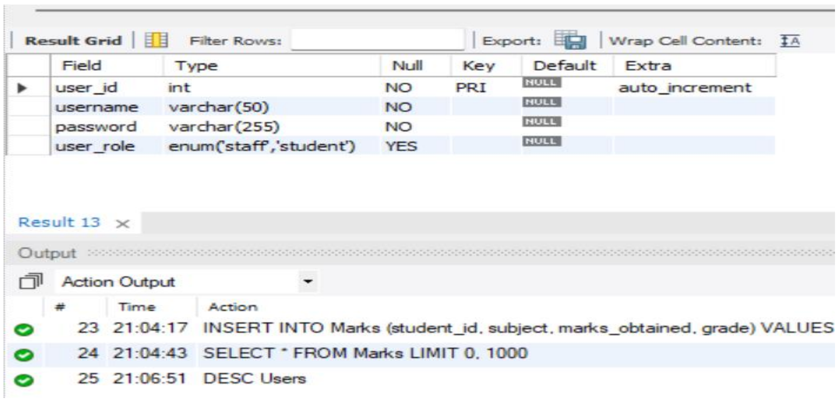


Table 1 : Users

Query :

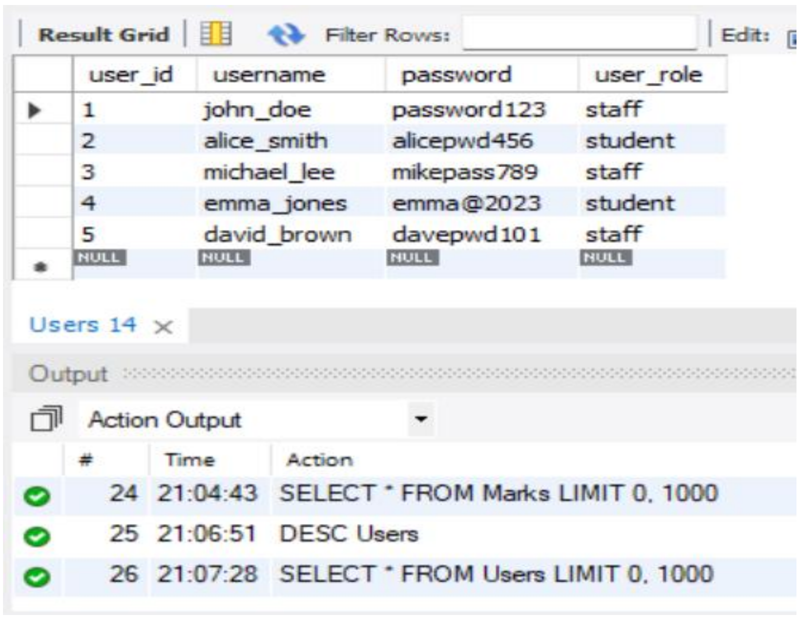
```
CREATE TABLE Users (
   user_id INT AUTO_INCREMENT PRIMARY KEY ,
   username VARCHAR(50) NOT NULL ,
   password VARCHAR(255) NOT NULL ,
   user_role ENUM(‘staff ’, ‘student’)
);

INSERT INTO Users ( user_id, username, password, user_role )
VALUES
(001, 'john_doe', 'password123', 'staff'),
(002, 'alice_smith', 'alicepwd456', 'student'),
(003, 'michael_lee', 'mikepass789', 'staff'),
(004, 'emma_jones', 'emma@2023', 'student'),
(005, 'david_brown', 'davepwd101', 'staff');

SELECT * FROM Users;
```

Output :



Field	Type	Null	Key	Default	Extra
user_id	int	NO	PRI	NULL	auto_increment
username	varchar(50)	NO		NULL	
password	varchar(255)	NO		NULL	
user_role	enum('staff','student')	YES		NULL	

```
#   Time      Action
23  21:04:17  INSERT INTO Marks (student_id, subject, marks_obtained, grade) VALUES
24  21:04:43  SELECT * FROM Marks LIMIT 0, 1000
25  21:06:51  DESC Users
```



user_id	username	password	user_role
1	john_doe	password123	staff
2	alice_smith	alicepwd456	student
3	michael_lee	mikepass789	staff
4	emma_jones	emma@2023	student
5	david_brown	davepwd101	staff
NULL	NULL	NULL	NULL

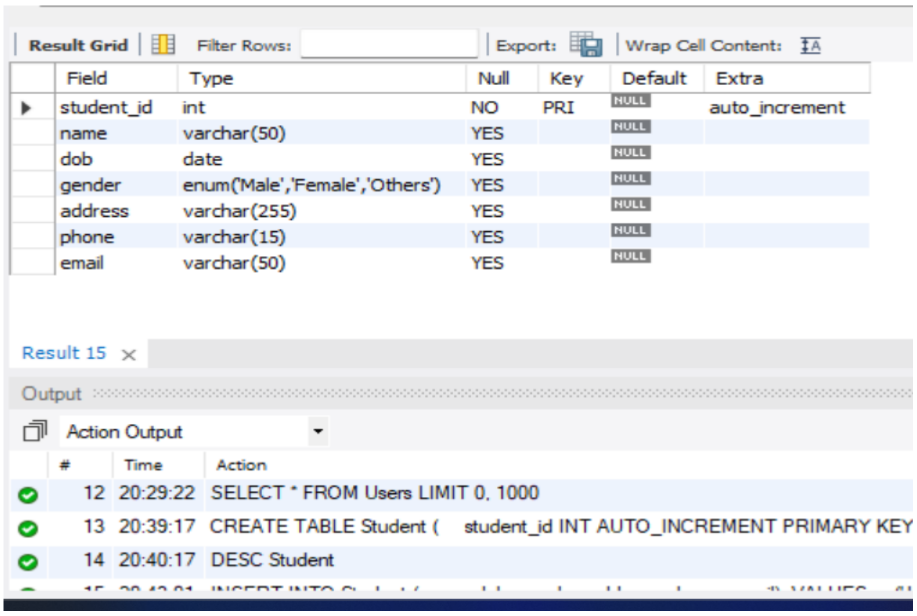
```
#   Time      Action
24  21:04:43  SELECT * FROM Marks LIMIT 0, 1000
25  21:06:51  DESC Users
26  21:07:28  SELECT * FROM Users LIMIT 0, 1000
```

Table 2 : Student

Query :

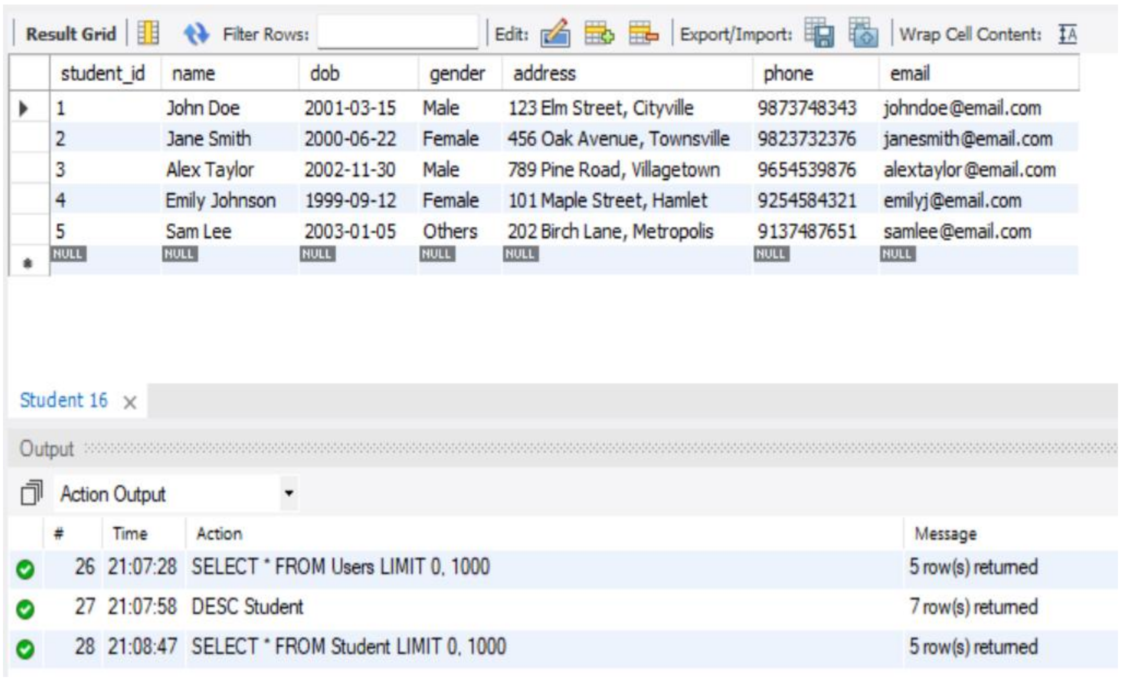
```
CREATE TABLE Student (
   student_id INT AUTO_INCREMENT PRIMARY KEY ,
   name VARCHAR(50) ,
   dob DATE ,
   gender ENUM(‘Male, ‘Female’, ‘Others’) ,
   address VARCHAR(255) ,
   phone VARCHAR(15) ,
   email VARCHAR(50)
);


INSERT INTO Student (name, dob, gender, address, phone, email)
VALUES
 ('John Doe', '2001-03-15', 'Male', '123 Elm Street, Cityville', '555-1234', 'johndoe@email.com'),
('Jane Smith', '2000-06-22', 'Female', '456 Oak Avenue, Townsville', '555-5678',
'janesmith@email.com'),
 ('Alex Taylor', '2002-11-30', 'Male', '789 Pine Road, Villagetown', '555-9876',
'alextaylor@email.com'),
('Emily Johnson', '1999-09-12', 'Female', '101 Maple Street, Hamlet', '555-4321',
'emilyj@email.com'),
('Sam Lee', '2003-01-05', 'Others', '202 Birch Lane, Metropolis', '555-8765', 'samlee@email.com');
```

Output :



Field	Type	Null	Key	Default	Extra
student_id	int	NO	PRI	NULL	auto_increment
name	varchar(50)	YES		NULL	
dob	date	YES		NULL	
gender	enum('Male','Female','Others')	YES		NULL	
address	varchar(255)	YES		NULL	
phone	varchar(15)	YES		NULL	
email	varchar(50)	YES		NULL	

```
     #   Time      Action
     12  20:29:22  SELECT * FROM Users LIMIT 0, 1000
     13  20:39:17  CREATE TABLE Student (    student_id INT AUTO_INCREMENT PRIMARY KEY
     14  20:40:17  DESC Student
     15  [illegible]  INSERT INTO Student ( [illegible]
```



	student_id	name	dob	gender	address	phone	email
▶	1	John Doe	2001-03-15	Male	123 Elm Street, Cityville	9873748343	johndoe@email.com
	2	Jane Smith	2000-06-22	Female	456 Oak Avenue, Townsville	9823732376	janesmith@email.com
	3	Alex Taylor	2002-11-30	Male	789 Pine Road, Villagetown	9654539876	alextaylor@email.com
	4	Emily Johnson	1999-09-12	Female	101 Maple Street, Hamlet	9254584321	emilyj@email.com
	5	Sam Lee	2003-01-05	Others	202 Birch Lane, Metropolis	9137487651	samlee@email.com
*	NULL	NULL	NULL	NULL	NULL	NULL	NULL

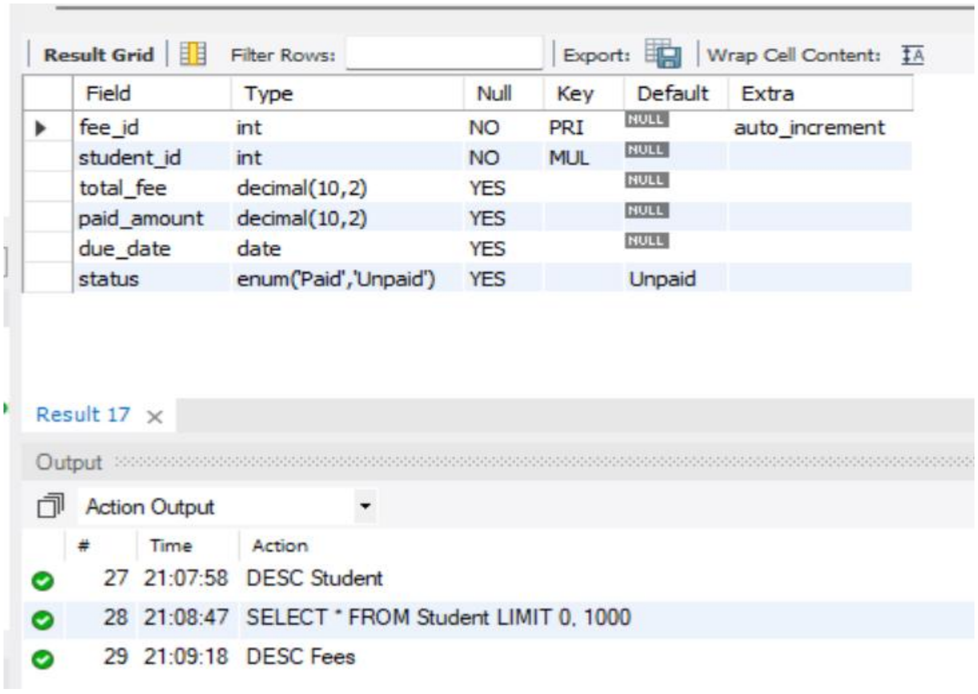
```
#   Time      Action                                Message
26  21:07:28  SELECT * FROM Users LIMIT 0, 1000     5 row(s) returned
27  21:07:58  DESC Student                          7 row(s) returned
28  21:08:47  SELECT * FROM Student LIMIT 0, 1000   5 row(s) returned
```

Table 3 : Fees

Query :

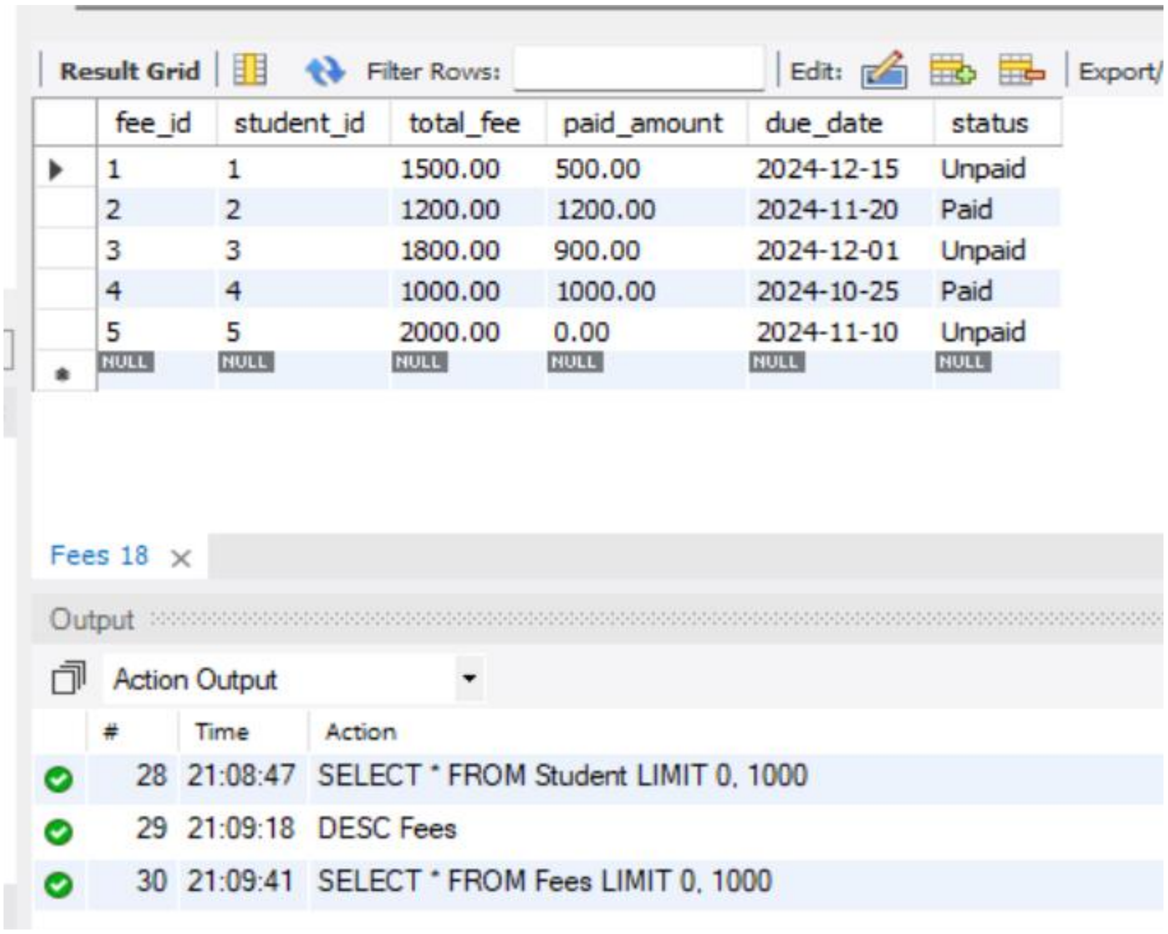
```
CREATE TABLE Fees (
   fee_id INT AUTO_INCREMENT PRIMARY KEY,
   student_id INT NOT NULL,
   total_fee DECIMAL(10, 2),
   paid_amount DECIMAL(10, 2),
   due_date DATE,
   status ENUM('Paid', 'Unpaid') DEFAULT 'Unpaid',
   FOREIGN KEY (student_id) REFERENCES Student(student_id)
);

INSERT INTO Fees (student_id, total_fee, paid_amount, due_date, status)
VALUES
 (1, 1500.00, 500.00, '2024-12-15', 'Unpaid'),
 (2, 1200.00, 1200.00, '2024-11-20', 'Paid'),
 (3, 1800.00, 900.00, '2024-12-01', 'Unpaid'),
 (4, 1000.00, 1000.00, '2024-10-25', 'Paid'),
 (5, 2000.00, 0.00, '2024-11-10', 'Unpaid');
```

Output :



Field	Type	Null	Key	Default	Extra
fee_id	int	NO	PRI	NULL	auto_increment
student_id	int	NO	MUL	NULL	
total_fee	decimal(10,2)	YES		NULL	
paid_amount	decimal(10,2)	YES		NULL	
due_date	date	YES		NULL	
status	enum('Paid','Unpaid')	YES		Unpaid	

```
#   Time      Action
27  21:07:58  DESC Student
28  21:08:47  SELECT * FROM Student LIMIT 0, 1000
29  21:09:18  DESC Fees
```



fee_id	student_id	total_fee	paid_amount	due_date	status
1	1	1500.00	500.00	2024-12-15	Unpaid
2	2	1200.00	1200.00	2024-11-20	Paid
3	3	1800.00	900.00	2024-12-01	Unpaid
4	4	1000.00	1000.00	2024-10-25	Paid
5	5	2000.00	0.00	2024-11-10	Unpaid
NULL	NULL	NULL	NULL	NULL	NULL

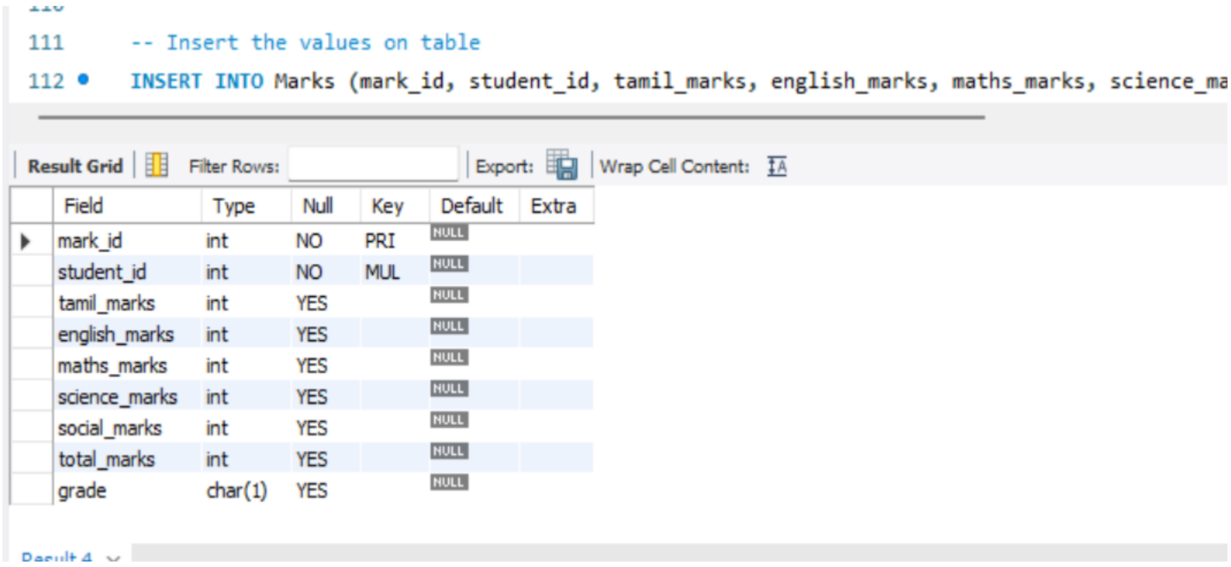
```
#   Time      Action
28  21:08:47  SELECT * FROM Student LIMIT 0, 1000
29  21:09:18  DESC Fees
30  21:09:41  SELECT * FROM Fees LIMIT 0, 1000
```

Table 4 : Marks

Query :

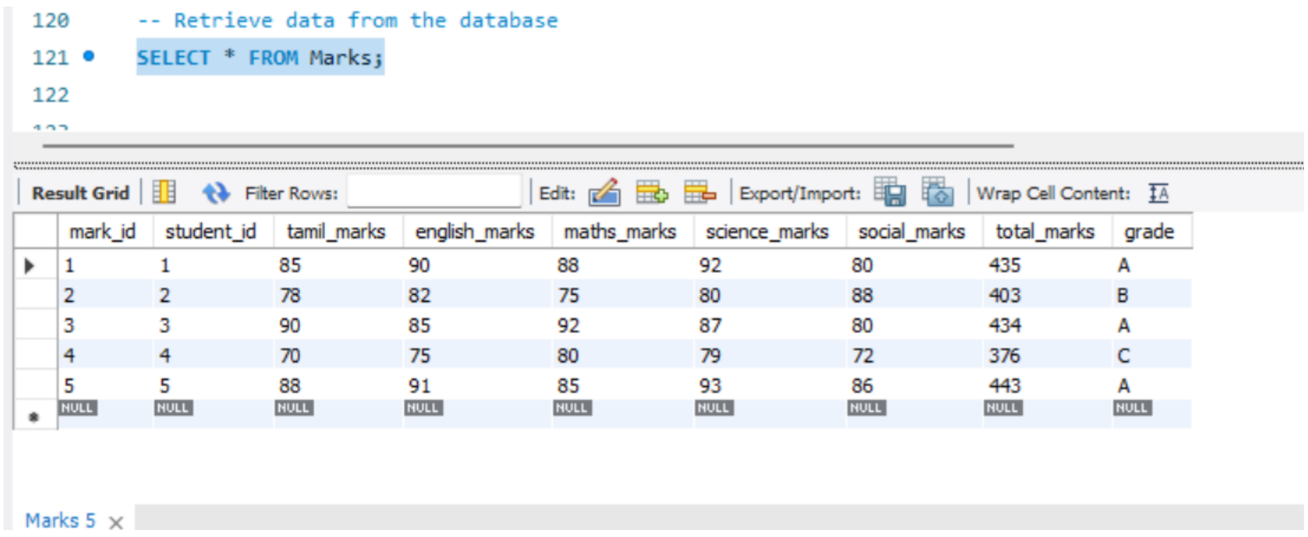
```
CREATE TABLE Marks (
   mark_id INT PRIMARY KEY,
   student_id INT NOT NULL,
   tamil_marks INT CHECK (marks_obtained BETWEEN 0 AND 100),
   english_marks INT CHECK (marks_obtained BETWEEN 0 AND 100),
   maths_marks INT CHECK (marks_obtained BETWEEN 0 AND 100),
   science_marks INT CHECK (marks_obtained BETWEEN 0 AND 100),
   social_marks INT CHECK (marks_obtained BETWEEN 0 AND 100),
   total_marks INT,
   grade CHAR(1) ,
   FOREIGN KEY (student_id) REFERENCES Students(student_id) ON DELETE CASCADE
);

INSERT INTO Marks (mark_id, student_id, tamil_marks, english_marks, maths_marks,
            science_marks, social_marks, total_marks, grade)
VALUES
            (1, 1, 85, 90, 88, 92, 80, 435, 'A'),
            (2, 2, 78, 82, 75, 80, 88, 403, 'B'),
            (3, 3, 90, 85, 92, 87, 80, 434, 'A'),
            (4, 4, 70, 75, 80, 79, 72, 376, 'C'),
            (5, 5, 88, 91, 85, 93, 86, 443, 'A');
```

Output :



```
-- Insert the values on table
INSERT INTO Marks (mark_id, student_id, tamil_marks, english_marks, maths_marks, science_ma
```

Field	Type	Null	Key	Default	Extra
mark_id	int	NO	PRI	NULL	
student_id	int	NO	MUL	NULL	
tamil_marks	int	YES		NULL	
english_marks	int	YES		NULL	
maths_marks	int	YES		NULL	
science_marks	int	YES		NULL	
social_marks	int	YES		NULL	
total_marks	int	YES		NULL	
grade	char(1)	YES		NULL	


```
-- Retrieve data from the database
SELECT * FROM Marks;
```

mark_id	student_id	tamil_marks	english_marks	maths_marks	science_marks	social_marks	total_marks	grade
1	1	85	90	88	92	80	435	A
2	2	78	82	75	80	88	403	B
3	3	90	85	92	87	80	434	A
4	4	70	75	80	79	72	376	C
5	5	88	91	85	93	86	443	A
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Operations based on the current Student Information System design to:

1. **Add a New Student**
2. **Update a Student details**
3. **Remove a Student**
4. **Mark Entry**
5. **View Entire Student Details**

1. **Add a New Student :**

Query:

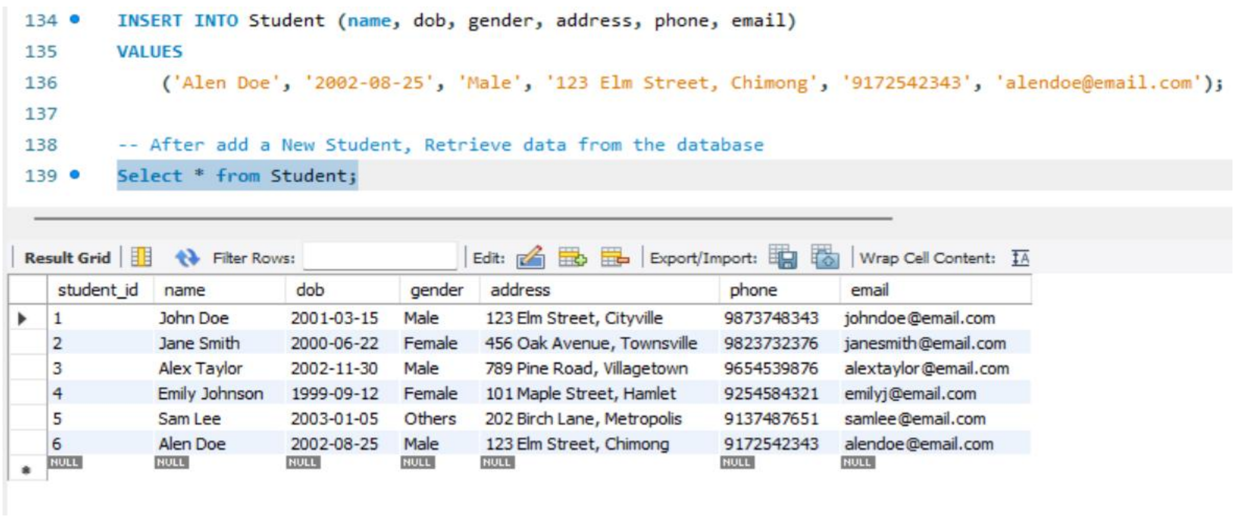
INSERT INTO Student (name, dob, gender, address, phone, email)

VALUES

('Alen Doe', '2002-08-25', 'Male', '123 Elm Street, Chimong', '9172542343', 'alendoe@email.com');

Select * from Student;

Output :



```
INSERT INTO Student (name, dob, gender, address, phone, email)
VALUES
    ('Alen Doe', '2002-08-25', 'Male', '123 Elm Street, Chimong', '9172542343', 'alendoe@email.com');

-- After add a New Student, Retrieve data from the database
Select * from Student;
```

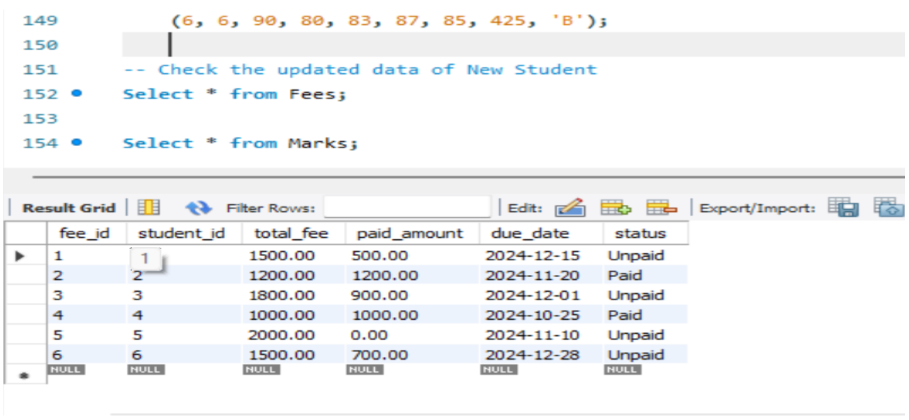
student_id	name	dob	gender	address	phone	email
1	John Doe	2001-03-15	Male	123 Elm Street, Cityville	9873748343	johndoe@email.com
2	Jane Smith	2000-06-22	Female	456 Oak Avenue, Townsville	9823732376	janesmith@email.com
3	Alex Taylor	2002-11-30	Male	789 Pine Road, Villagetown	9654539876	alextaylor@email.com
4	Emily Johnson	1999-09-12	Female	101 Maple Street, Hamlet	9254584321	emilyj@email.com
5	Sam Lee	2003-01-05	Others	202 Birch Lane, Metropolis	9137487651	samlee@email.com
6	Alen Doe	2002-08-25	Male	123 Elm Street, Chimong	9172542343	alendoe@email.com
NULL	NULL	NULL	NULL	NULL	NULL	NULL

2. Update a Student details :

Query:

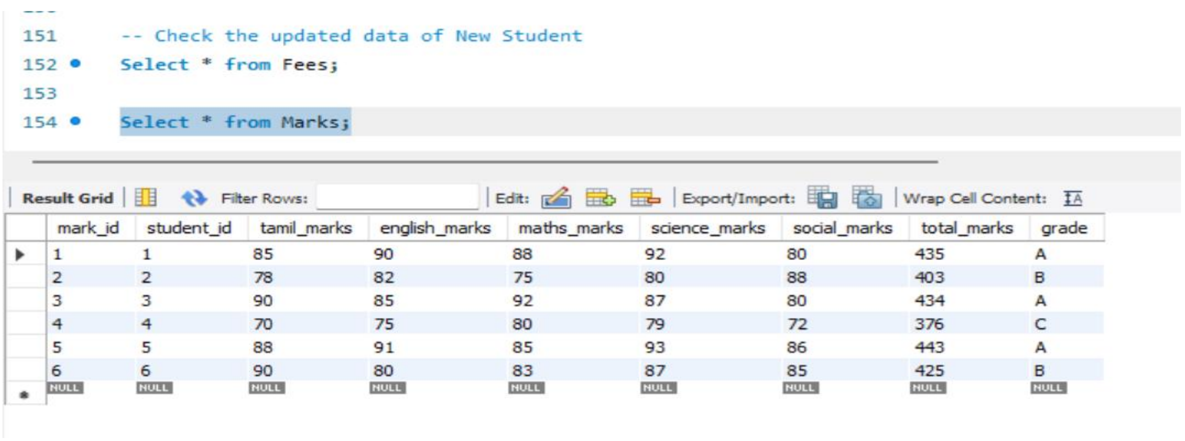
```
            INSERT INTO Fees (student_id, total_fee, paid_amount, due_date, status)
            VALUES
                        (6, 1500.00, 700.00, '2024-12-28', 'Unpaid');


            INSERT INTO Marks (mark_id, student_id, tamil_marks, english_marks, maths_marks,
science_marks, social_marks, total_marks, grade)
            VALUES
                        (6, 6, 90, 80, 83, 87, 85, 425, 'B');
            Select * from Fees;
            Select * from Marks;
```

Output:



```
        (6, 6, 90, 80, 83, 87, 85, 425, 'B');

-- Check the updated data of New Student
Select * from Fees;

Select * from Marks;
```

fee_id	student_id	total_fee	paid_amount	due_date	status
1	1	1500.00	500.00	2024-12-15	Unpaid
2	2	1200.00	1200.00	2024-11-20	Paid
3	3	1800.00	900.00	2024-12-01	Unpaid
4	4	1000.00	1000.00	2024-10-25	Paid
5	5	2000.00	0.00	2024-11-10	Unpaid
6	6	1500.00	700.00	2024-12-28	Unpaid
NULL	NULL	NULL	NULL	NULL	NULL



```
-- Check the updated data of New Student
Select * from Fees;

Select * from Marks;
```

mark_id	student_id	tamil_marks	english_marks	maths_marks	science_marks	social_marks	total_marks	grade
1	1	85	90	88	92	80	435	A
2	2	78	82	75	80	88	403	B
3	3	90	85	92	87	80	434	A
4	4	70	75	80	79	72	376	C
5	5	88	91	85	93	86	443	A
6	6	90	80	83	87	85	425	B
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

3. Remove a Student :

Query:

DELETE FROM Student WHERE student_id = 2;

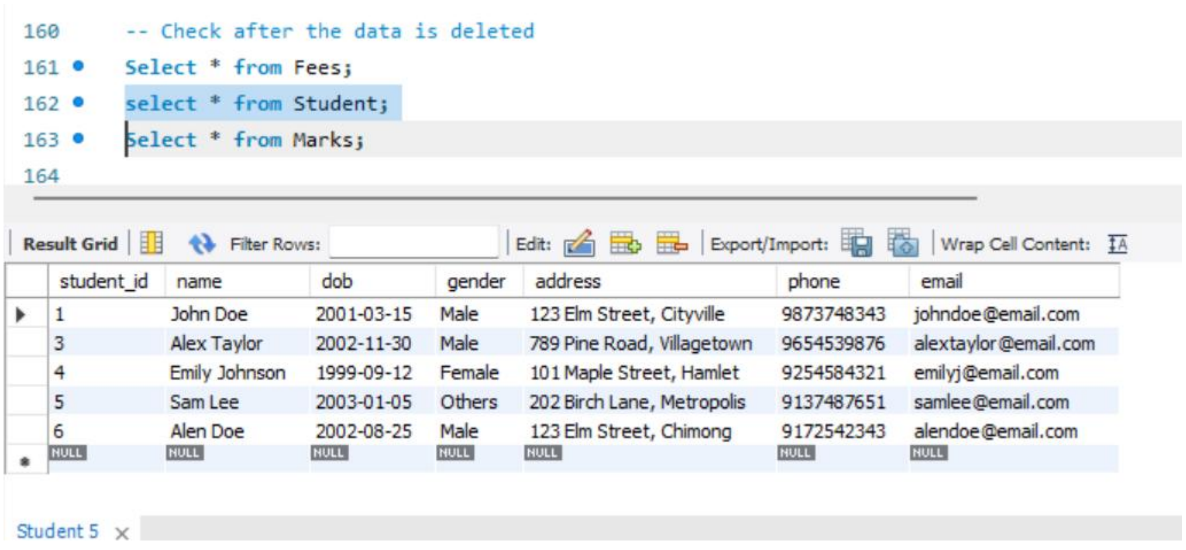
DELETE FROM Fees WHERE student_id = 2;

Select * from Fees;

select * from Student;

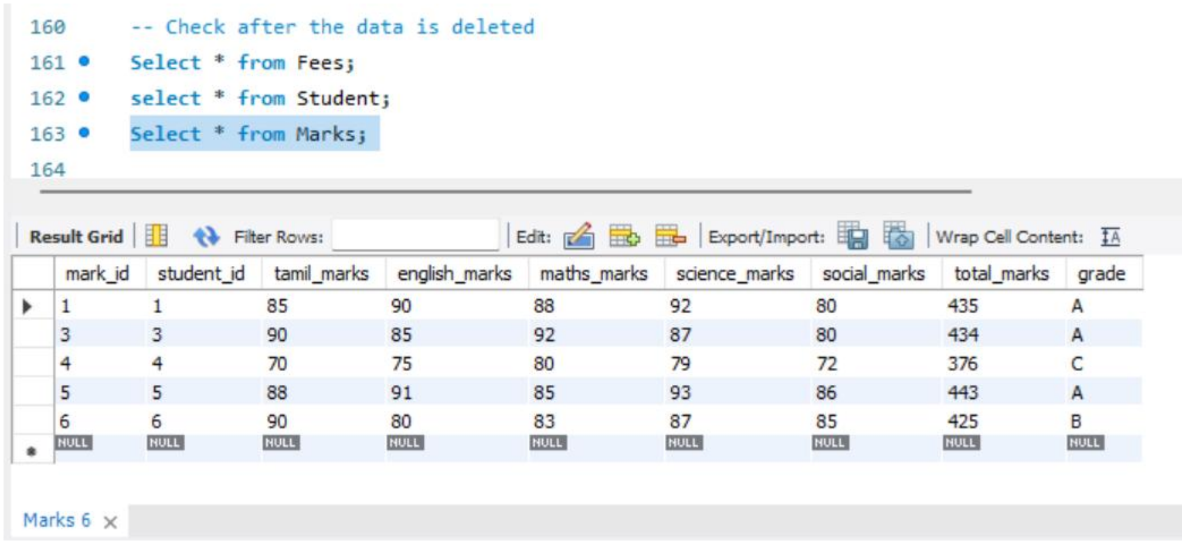
Select * from Marks;

Output:



```
-- Check after the data is deleted
Select * from Fees;
select * from Student;
Select * from Marks;
```

	student_id	name	dob	gender	address	phone	email
▶	1	John Doe	2001-03-15	Male	123 Elm Street, Cityville	9873748343	johndoe@email.com
	3	Alex Taylor	2002-11-30	Male	789 Pine Road, Villagetown	9654539876	alextaylor@email.com
	4	Emily Johnson	1999-09-12	Female	101 Maple Street, Hamlet	9254584321	emilyj@email.com
	5	Sam Lee	2003-01-05	Others	202 Birch Lane, Metropolis	9137487651	samlee@email.com
	6	Alen Doe	2002-08-25	Male	123 Elm Street, Chimong	9172542343	alendoe@email.com
*	NULL	NULL	NULL	NULL	NULL	NULL	NULL



```
-- Check after the data is deleted
Select * from Fees;
select * from Student;
Select * from Marks;
```

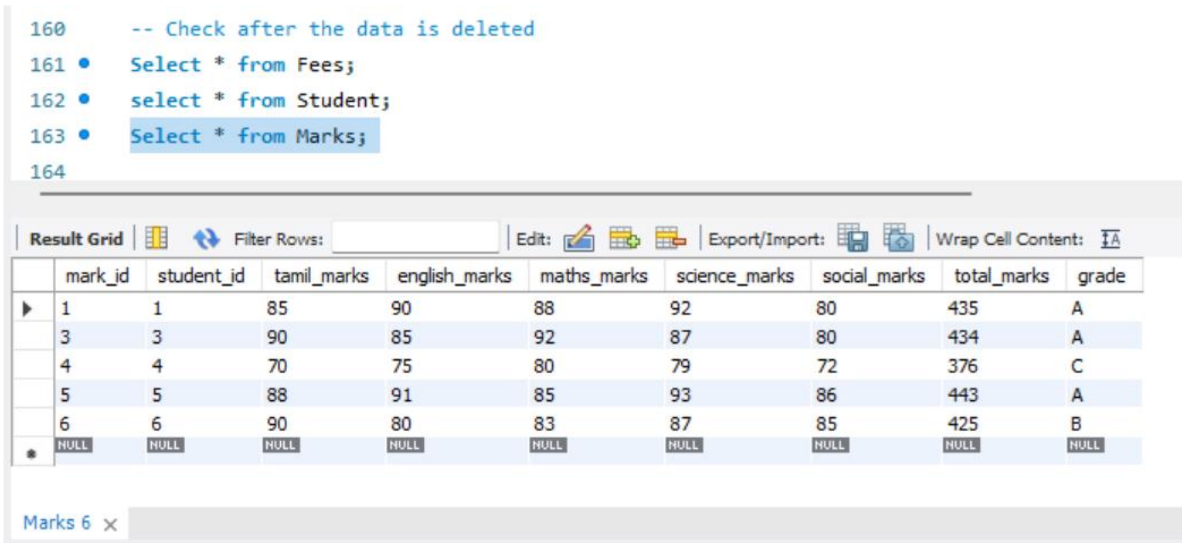
	mark_id	student_id	tamil_marks	english_marks	maths_marks	science_marks	social_marks	total_marks	grade
▶	1	1	85	90	88	92	80	435	A
	3	3	90	85	92	87	80	434	A
	4	4	70	75	80	79	72	376	C
	5	5	88	91	85	93	86	443	A
	6	6	90	80	83	87	85	425	B
*	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

4. Mark Entry for New Student :

Query:

```
INSERT INTO Marks (mark_id, student_id, tamil_marks, english_marks, maths_marks,
science_marks, social_marks, total_marks, grade)
VALUES
    (6, 6, 90, 80, 83, 87, 85, 425, 'B');
Select * from Marks;
```

Output:



```
-- Check after the data is deleted
Select * from Fees;
select * from Student;
Select * from Marks;
```

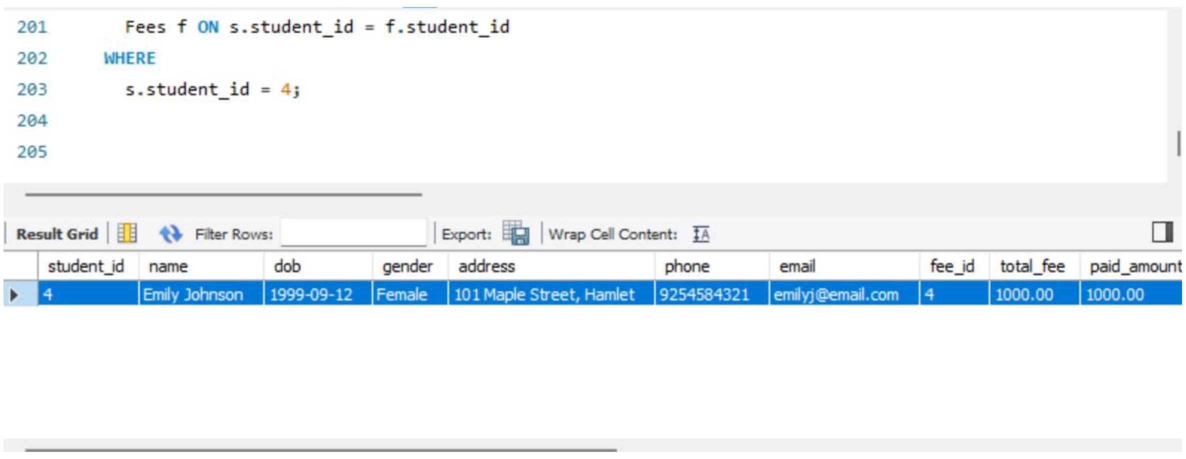
mark_id	student_id	tamil_marks	english_marks	maths_marks	science_marks	social_marks	total_marks	grade
1	1	85	90	88	92	80	435	A
3	3	90	85	92	87	80	434	A
4	4	70	75	80	79	72	376	C
5	5	88	91	85	93	86	443	A
6	6	90	80	83	87	85	425	B
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

5. View Entire Student Details

Query :

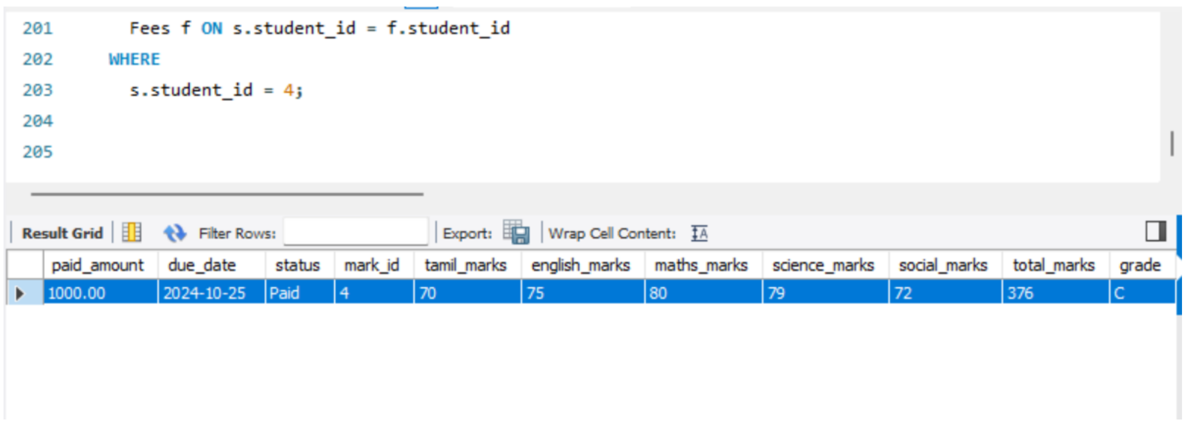
```
SELECT
        s.student_id,
        s.name,
        s.dob,
        s.gender,
        s.address,
        s.phone,
        s.email
        f.fee_id,
        f.total_fee,
        f.paid_amount,
        f.due_date,
        f.status,
        m.mark_id,
        m.tamil_marks,
        m.english_marks,
        m.maths_marks,
        m.science_marks,
        m.social_marks,
        m.total_marks,
        m.grade
FROM
        Student s
JOIN
        Marks m ON s.student_id = m.student_id
JOIN
        Fees f ON s.student_id = f.student_id
WHERE
        s.student_id = 4;
```

Output :



```
    Fees f ON s.student_id = f.student_id
  WHERE
    s.student_id = 4;
```

student_id	name	dob	gender	address	phone	email	fee_id	total_fee	paid_amount
4	Emily Johnson	1999-09-12	Female	101 Maple Street, Hamlet	9254584321	emilyj@email.com	4	1000.00	1000.00



```
    Fees f ON s.student_id = f.student_id
  WHERE
    s.student_id = 4;
```

paid_amount	due_date	status	mark_id	tamil_marks	english_marks	maths_marks	science_marks	social_marks	total_marks	grade
1000.00	2024-10-25	Paid	4	70	75	80	79	72	376	C