

Dhaka International University
Department of CSE

Course Code: 0612-304

Course Title: Database Management System Lab

Project Proposal on
Student Result Management System

Submitted by:

Md. Tamim Hossen

Roll: 24

Md. Montasir Hasan *

Roll: 27

Nusrat Jahan

Roll: 38

Mohammed Shahparan

Roll: 39

Nushaiba Kawser Era

Roll: 41

Submitted to:

Md. Ayon Mia

Lecturer,

Department of CSE

Dhaka International University

Contents

Overview.....	2
Objectives.....	2
Problem Statement	2
Entities.....	3
ER Diagram (without relationship):	4
Relationships	4
Data Base	7
Logical Sql Queries.....	16
Outcome	21
Discussion.....	21

Overview

It's a centralized digital platform for result management. Teachers can input marks → System calculates grades → Students can view results securely. Built using database design concepts: entities, relationships, and normalization.

Tables: Student, Teacher, Course, Result, ResultDetails, Enrolls, Conducts, etc.

Objectives

- ❖ Efficiently store and manage students' academic records digitally.
- ❖ Automatically generate accurate result sheets from stored marks.
- ❖ Minimize human errors in result calculation and management.
- ❖ Allow authorized teachers and administrators to input, update, and manage marks.

Problem Statement

Many educational institutions rely on manual or semi-manual methods to store and process student academic results. This leads to several issues such as delayed result generation, human errors in calculations, duplication of data, and difficulty in access for both staff and students. Manual processing is also labor-intensive, time-consuming, and prone to data loss or unauthorized access, negatively affecting the accuracy and timely delivery of student performance information.

Solution Statement

The Student Result Management System will automate the collection, storage, and processing of students' academic data. This centralized digital system will enable teachers to input marks easily, automatically calculate final results and grades, and allow students to securely view their results anytime. The system will enhance data accuracy, reduce administrative workload, and provide faster access to academic performance information.

Entities

✓ Strong Entities:

➤ Student

student_id (PK)
name
email
address
reg_number
department
batch
roll
date_of_birth
age

➤ Course

course_code (PK)
course_title
credit

➤ Teacher

teacher_id (PK)
name
email
department

➤ Result

result_id (PK)
student_id (FK)
course_code (FK)
teacher_id (FK)
marks
cgpa
grade

✓ Weak Entities

➤ ResultDetails

result_id (FK)
marks
course_code (FK)
exam_type
remarks

ER Diagram (without relationship)

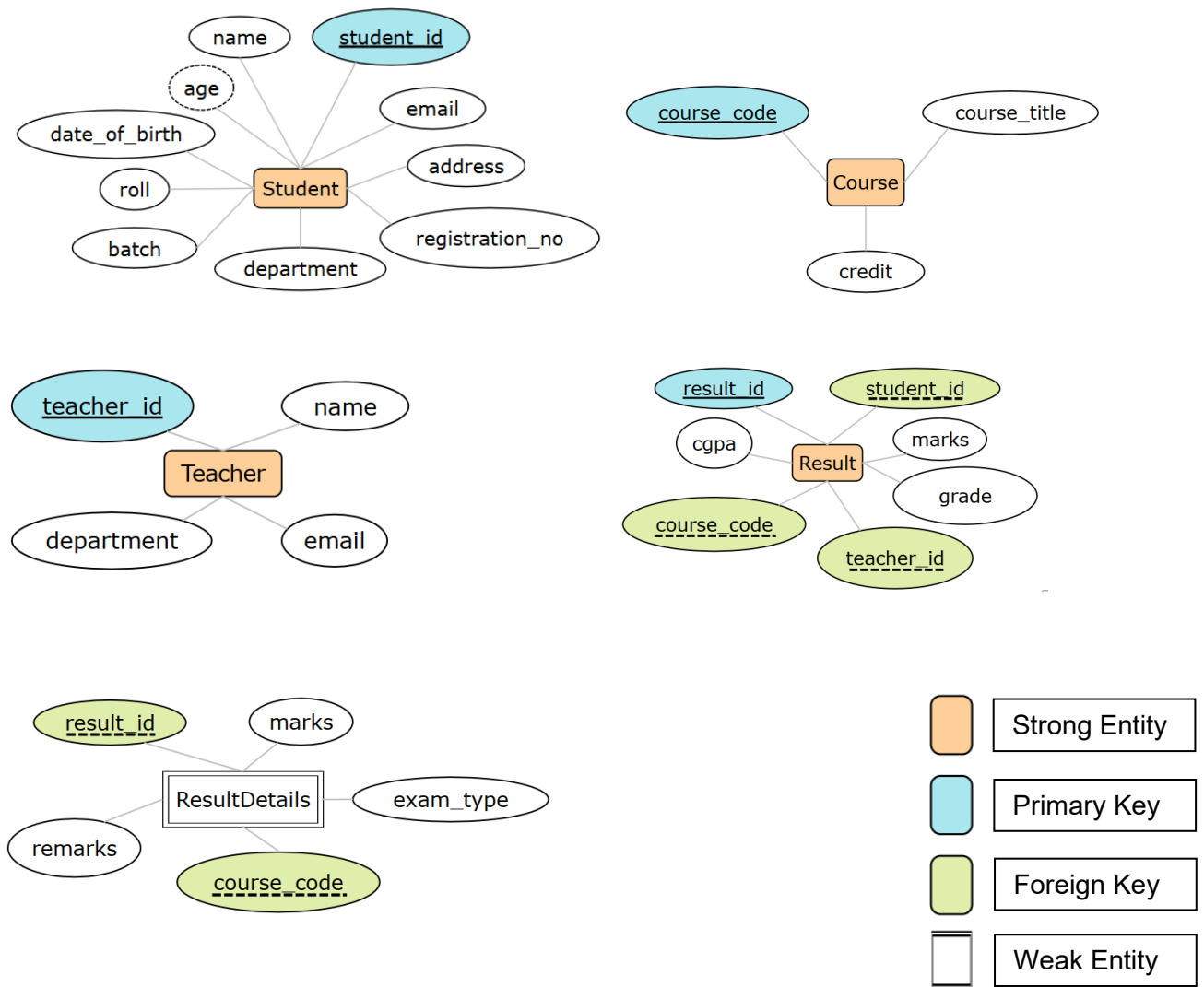


Figure 1: ER diagram of Student Result Management System without relationship

Relationships

- ❖ A student can enroll in many courses, and each course can have many students.
→ **Many-to-Many (M:N)**
- ❖ One teacher can teach many courses, also each course is taught by many teacher.
→ **One-to-Many (N:M)**
- ❖ One student can have many results (for different courses).
→ **One-to-Many (1:M)**
- ❖ One course can have many results (for different students).
→ **One-to-Many (1:M)**
- ❖ Each result has its detailed information like marks, grade, percentage, etc.
→ **One-to-One (1:1)** (if multiple details stored)

ER Diagram (with relationship)

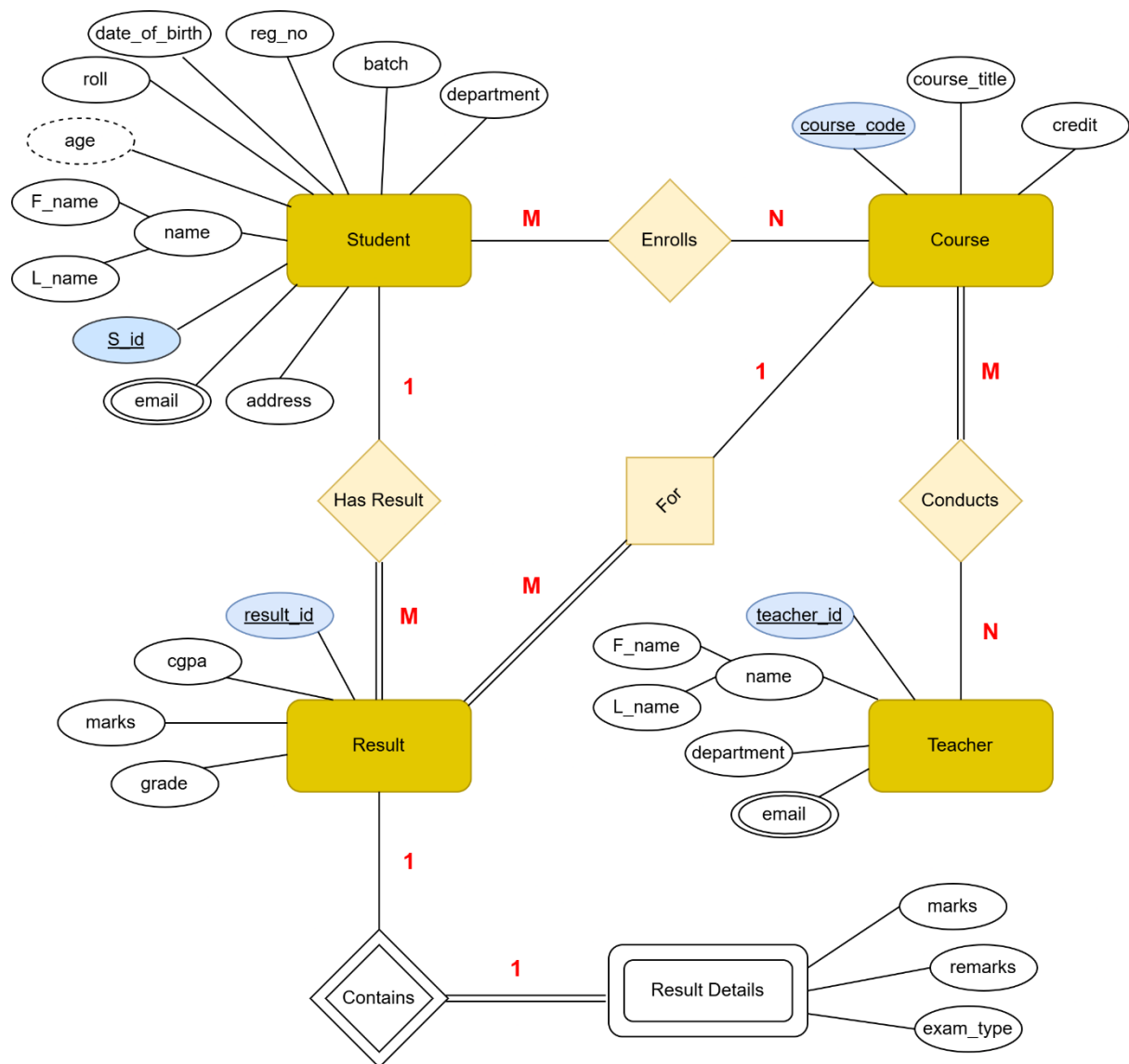


Figure 2: ER diagram of Student Result Management System with relationship

Mapping

1. Student:

<u>Student_id</u>	F_name	L_name	roll	address	Reg_no	DoB	Batch	Department
-------------------	--------	--------	------	---------	--------	-----	-------	------------

2. Student Email:

<u>Student_id</u>	S_Email
-------------------	---------

3. Course:

<u>Course_id</u>	Course_title	credit	Teacher_id
------------------	--------------	--------	------------

4. Teacher:

<u>Teacher_id</u>	F_name	L_name	Department
-------------------	--------	--------	------------

5. Teacher Email:

<u>Teacher_id</u>	Teacher_Email
-------------------	---------------

6. Result:

<u>Result_id</u>	cgpa	grade	marks	<u>Course_id</u>	<u>Student_id</u>
------------------	------	-------	-------	------------------	-------------------

7. Result Details:

<u>Result_id</u>	remarks	Exam_type
------------------	---------	-----------

8. Enrolls:

<u>Student_id</u>	<u>Course_id</u>
-------------------	------------------

9. Conducts:

<u>Course_id</u>	<u>Teacher_id</u>
------------------	-------------------

Data Base

1. Create Database:

```
create Database student_result_management_system_d87;
```

2. Student Table:

```
2
3 • CREATE TABLE Student (
4     Student_id INT PRIMARY KEY,
5     F_name VARCHAR(30) NOT NULL,
6     L_name VARCHAR(30) NOT NULL,
7     roll INT UNIQUE NOT NULL,
8     address VARCHAR(100),
9     Reg_no VARCHAR(20) UNIQUE NOT NULL,
10    DoB DATE NOT NULL,
11    Batch VARCHAR(10) NOT NULL,
12    Department VARCHAR(30) NOT NULL
13 );
14
15 • INSERT INTO Student VALUES
16 (1, 'Nusrat', 'Jahan', 101, 'Badda, Dhaka', 'REG2025001', '2004-03-15', '56', 'CSE'),
17 (2, 'Rafi', 'Hasan', 102, 'Mirpur, Dhaka', 'REG2025002', '2003-07-22', '56', 'CSE'),
18 (3, 'Mitu', 'Akter', 103, 'Uttara, Dhaka', 'REG2025003', '2004-01-10', '57', 'EEE'),
19 (4, 'Sami', 'Rahman', 104, 'Banani, Dhaka', 'REG2025004', '2003-04-09', '57', 'CSE'),
20 (5, 'Tanha', 'Khan', 105, 'Dhanmondi, Dhaka', 'REG2025005', '2004-02-05', '58', 'BBA'),
21 (6, 'Arif', 'Hossain', 106, 'Badda, Dhaka', 'REG2025006', '2003-12-12', '56', 'EEE'),
22 (7, 'Farzana', 'Noor', 107, 'Mirpur, Dhaka', 'REG2025007', '2004-05-15', '57', 'CSE'),
23 (8, 'Imran', 'Ali', 108, 'Uttara, Dhaka', 'REG2025008', '2003-06-22', '58', 'CSE'),
24 (9, 'Sadiah', 'Rahim', 109, 'Banani, Dhaka', 'REG2025009', '2004-07-09', '59', 'BBA'),
25 (10, 'Naeem', 'Islam', 110, 'Dhanmondi, Dhaka', 'REG2025010', '2003-09-21', '56', 'EEE');
```

Student Table (Output):

Result Grid Filter Rows: Edit: Export/Import: Wrap Cell Content:									
	Student_id	F_name	L_name	roll	address	Reg_no	DoB	Batch	Department
▶	1	Nusrat	Jahan	101	Badda, Dhaka	REG2025001	2004-03-15	56	CSE
	2	Rafi	Hasan	102	Mirpur, Dhaka	REG2025002	2003-07-22	56	CSE
	3	Mitu	Akter	103	Uttara, Dhaka	REG2025003	2004-01-10	57	EEE
	4	Sami	Rahman	104	Banani, Dhaka	REG2025004	2003-04-09	57	CSE
	5	Tanha	Khan	105	Dhanmondi, Dhaka	REG2025005	2004-02-05	58	BBA
	6	Arif	Hossain	106	Badda, Dhaka	REG2025006	2003-12-12	56	EEE
	7	Farzana	Noor	107	Mirpur, Dhaka	REG2025007	2004-05-15	57	CSE
	8	Imran	Ali	108	Uttara, Dhaka	REG2025008	2003-06-22	58	CSE
	9	Sadia	Rahim	109	Banani, Dhaka	REG2025009	2004-07-09	59	BBA
	10	Naeem	Islam	110	Dhanmondi, Dhaka	REG2025010	2003-09-21	56	EEE
•	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

3. Student Email Table:

```
28 • CREATE TABLE Student_Email (  
29     Student_id INT NOT NULL,  
30     S_Email VARCHAR(50) UNIQUE NOT NULL,  
31     FOREIGN KEY (Student_id) REFERENCES Student(Student_id)  
32 );  
33  
34 • INSERT INTO Student_Email VALUES  
35     (1, 'nusrat@gmail.com'),  
36     (2, 'rafi@gmail.com'),  
37     (3, 'mitu@gmail.com'),  
38     (4, 'sami@gmail.com'),  
39     (5, 'tanha@gmail.com'),  
40     (6, 'arif@gmail.com'),  
41     (7, 'farzana@gmail.com'),  
42     (8, 'imran@gmail.com'),  
43     (9, 'sadia@gmail.com'),  
44     (10, 'naeem@gmail.com');
```

Student Email Table (Output):

Result Grid		
Filter Rows:		
Edit: Export/Import: Wrap Cell Content:		
	Student_id	S_Email
▶	1	nusrat@gmail.com
	2	rafi@gmail.com
	3	mitu@gmail.com
	4	sami@gmail.com
	5	tanha@gmail.com
	6	arif@gmail.com
	7	farzana@gmail.com
	8	imran@gmail.com
	9	sadia@gmail.com
	10	naeem@gmail.com
•	NULL	NULL

4. Teacher Table:

```
47 • CREATE TABLE Teacher (  
48     Teacher_id INT PRIMARY KEY,  
49     F_name VARCHAR(30) NOT NULL,  
50     L_name VARCHAR(30) NOT NULL,  
51     Department VARCHAR(30) NOT NULL  
52 );  
53  
54 • INSERT INTO Teacher VALUES  
55     (1, 'Shamim', 'Ahmed', 'CSE'),  
56     (2, 'Fahim', 'Rahman', 'CSE'),  
57     (3, 'Joya', 'Akter', 'EEE'),  
58     (4, 'Sumon', 'Khan', 'BBA'),  
59     (5, 'Afsana', 'Nahar', 'EEE'),  
60     (6, 'Mahfuz', 'Alam', 'CSE'),  
61     (7, 'Farid', 'Uddin', 'CSE'),  
62     (8, 'Tahmid', 'Hasan', 'CSE'),  
63     (9, 'Mira', 'Begum', 'EEE'),  
64     (10, 'Rashid', 'Hossain', 'CSE');
```

Teacher Table (Output):

Result Grid

Filter Rows:

Edit:

Export/Import:

Wr

	Teacher_id	F_name	L_name	Department
▶	1	Shamim	Ahmed	CSE
	2	Fahim	Rahman	CSE
	3	Joya	Akter	EEE
	4	Sumon	Khan	BBA
	5	Afsana	Nahar	EEE
	6	Mahfuz	Alam	CSE
	7	Farid	Uddin	CSE
	8	Tahmid	Hasan	CSE
	9	Mira	Begum	EEE
	10	Rashid	Hossain	CSE
✱	NULL	NULL	NULL	NULL

5. Teacher Email Table:

```
67 • CREATE TABLE Teacher_Email (  
68     Teacher_id INT NOT NULL,  
69     Teacher_Email VARCHAR(50) UNIQUE NOT NULL,  
70     FOREIGN KEY (Teacher_id) REFERENCES Teacher(Teacher_id)  
71 );  
72  
73 • INSERT INTO Teacher_Email VALUES  
74     (1, 'shamim@univ.com'),  
75     (2, 'fahim@univ.com'),  
76     (3, 'joya@univ.com'),  
77     (4, 'sumon@univ.com'),  
78     (5, 'afsana@univ.com'),  
79     (6, 'mahfuz@univ.com'),  
80     (7, 'farid@univ.com'),  
81     (8, 'tahmid@univ.com'),  
82     (9, 'mira@univ.com'),  
83     (10, 'rashid@univ.com');  
84
```

Teacher Email Table (Output):



Result Grid		Filter Rows:	Edit:	Export/Import:	Wrap Cell
	Teacher_id	Teacher_Email			
▶	1	shamim@univ.com			
	2	fahim@univ.com			
	3	joya@univ.com			
	4	sumon@univ.com			
	5	afsana@univ.com			
	6	mahfuz@univ.com			
	7	farid@univ.com			
	8	tahmid@univ.com			
	9	mira@univ.com			
	10	rashid@univ.com			
•	NULL	NULL			




6. Course Table:



```
86 • CREATE TABLE Course (  
87     Course_id INT PRIMARY KEY,  
88     Course_title VARCHAR(50) NOT NULL,  
89     credit INT DEFAULT 3,  
90     Teacher_id INT NOT NULL,  
91     FOREIGN KEY (Teacher_id) REFERENCES Teacher(Teacher_id)  
92 );  
93  
94 • INSERT INTO Course VALUES  
95     (101, 'Database Systems', 3, 1),  
96     (102, 'Digital Logic', 3, 2),  
97     (103, 'Data Structures', 3, 3),  
98     (104, 'Accounting', 2, 4),  
99     (105, 'Electronics', 3, 5),  
100    (106, 'AI Fundamentals', 3, 6),  
101    (107, 'Web Development', 3, 7),  
102    (108, 'Networking', 3, 8),  
103    (109, 'Microprocessor', 3, 9),  
104    (110, 'Machine Learning', 3, 10);
```

Course Table (Output):

Result Grid

  Filter Rows:

Edit:   

Export/Import:  

Wrap Cell

	Course_id	Course_title	credit	Teacher_id
▶	101	Database Systems	3	1
	102	Digital Logic	3	2
	103	Data Structures	3	3
	104	Accounting	2	4
	105	Electronics	3	5
	106	AI Fundamentals	3	6
	107	Web Development	3	7
	108	Networking	3	8
	109	Microprocessor	3	9
	110	Machine Learning	3	10
•	NULL	NULL	NULL	NULL

7. Result Table:

```
108 • CREATE TABLE Result (  
109     Result_id INT PRIMARY KEY,  
110     cgpa FLOAT CHECK (cgpa BETWEEN 0.00 AND 4.00) NOT NULL,  
111     grade CHAR(2) NOT NULL,  
112     marks INT CHECK (marks BETWEEN 0 AND 100) NOT NULL,  
113     Course_id INT NOT NULL,  
114     Student_id INT NOT NULL,  
115     FOREIGN KEY (Course_id) REFERENCES Course(Course_id),  
116     FOREIGN KEY (Student_id) REFERENCES Student(Student_id)  
117 );  
118  
119 • INSERT INTO Result VALUES  
120     (1, 3.80, 'A', 85, 101, 1),  
121     (2, 3.50, 'A-', 78, 102, 2),  
122     (3, 3.60, 'A-', 81, 103, 3),  
123     (4, 3.90, 'A', 88, 104, 4),  
124     (5, 3.70, 'A-', 83, 105, 5),  
125     (6, 3.20, 'B+', 75, 106, 6),  
126     (7, 3.00, 'B', 70, 107, 7),  
127     (8, 3.85, 'A', 86, 108, 8),  
128     (9, 3.40, 'A-', 79, 109, 9),  
129     (10, 3.95, 'A+', 90, 110, 10);  
130
```

Result Table (Output):

Result Grid						
		Filter Rows:		Edit:		Export/Import:
	Result_id	cgpa	grade	marks	Course_id	Student_id
▶	1	3.8	A	85	101	1
	2	3.5	A-	78	102	2
	3	3.6	A-	81	103	3
	4	3.9	A	88	104	4
	5	3.7	A-	83	105	5
	6	3.2	B+	75	106	6
	7	3	B	70	107	7
	8	3.85	A	86	108	8
	9	3.4	A-	79	109	9
	10	3.95	A+	90	110	10
✱	NULL	NULL	NULL	NULL	NULL	NULL

8. Result Details Table:

```
132 • CREATE TABLE Result_Details (  
133     Result_id INT NOT NULL,  
134     remarks VARCHAR(50),  
135     Exam_type VARCHAR(20) NOT NULL,  
136     FOREIGN KEY (Result_id) REFERENCES Result(Result_id)  
137 );  
138  
139 • INSERT INTO Result_Details VALUES  
140 (1, 'Excellent', 'Final'),  
141 (2, 'Good', 'Mid'),  
142 (3, 'Very Good', 'Final'),  
143 (4, 'Outstanding', 'Final'),  
144 (5, 'Satisfactory', 'Mid'),  
145 (6, 'Average', 'Mid'),  
146 (7, 'Improved', 'Final'),  
147 (8, 'Excellent', 'Final'),  
148 (9, 'Good', 'Mid'),  
149 (10, 'Top Score', 'Final');  
150
```

Result Details Table (Output):

Result Grid	Filter Rows:	Export:	Wrap Cell Content:
Result_id	remarks	Exam_type	
1	Excellent	Final	
2	Good	Mid	
3	Very Good	Final	
4	Outstanding	Final	
5	Satisfactory	Mid	
6	Average	Mid	
7	Improved	Final	
8	Excellent	Final	
9	Good	Mid	
10	Top Score	Final	

9. Enrolls Table:

```
152 • CREATE TABLE Enrolls (  
153     Student_id INT NOT NULL,  
154     Course_id INT NOT NULL,  
155     PRIMARY KEY (Student_id, Course_id),  
156     FOREIGN KEY (Student_id) REFERENCES Student(Student_id),  
157     FOREIGN KEY (Course_id) REFERENCES Course(Course_id)  
158 );  
159  
160 • INSERT INTO Enrolls VALUES  
161     (1, 101),  
162     (2, 102),  
163     (3, 103),  
164     (4, 104),  
165     (5, 105),  
166     (6, 106),  
167     (7, 107),  
168     (8, 108),  
169     (9, 109),  
170     (10, 110);
```

Enrolls Table (Output):

Result Grid			Filter Rows:	Edit:	Exp
	Student_id	Course_id			
▶	1	101			
	2	102			
	3	103			
	4	104			
	5	105			
	6	106			
	7	107			
	8	108			
	9	109			
	10	110			
✱	NULL	NULL			

10. Conducts Table:

```
172
173 • CREATE TABLE Conducts (
174     Course_id INT NOT NULL,
175     Teacher_id INT NOT NULL,
176     PRIMARY KEY (Course_id, Teacher_id),
177     FOREIGN KEY (Course_id) REFERENCES Course(Course_id),
178     FOREIGN KEY (Teacher_id) REFERENCES Teacher(Teacher_id)
179 );
180
181 • INSERT INTO Conducts VALUES
182     (101, 1),
183     (102, 2),
184     (103, 3),
185     (104, 4),
186     (105, 5),
187     (106, 6),
188     (107, 7),
189     (108, 8),
190     (109, 9),
191     (110, 10);
```

Conducts Table (Output):

Result Grid			Filter Rows:	Edit:	Export/Import:	
	Course_id	Teacher_id				
▶	101	1				
	102	2				
	103	3				
	104	4				
	105	5				
	106	6				
	107	7				
	108	8				
	109	9				
	110	10				
✱	NULL	NULL				

Logical Sql Queries

1. Show all students with their email addresses:

1. Show all students with their email addresses

```
SELECT S.Student_id, CONCAT(S.F_name, ' ', S.L_name) AS Student_Name, SE.S_Email
FROM Student S
JOIN Student_Email SE ON S.Student_id = SE.Student_id;
```

Output:

Result Grid	Filter Rows:	Export:	Wrap Cell Content:
Student_id	Student_Name	S_Email	
1	Nusrat Jahan	nusrat@gmail.com	
2	Rafi Hasan	rafi@gmail.com	
3	Mitu Akter	mitu@gmail.com	
4	Sami Rahman	sami@gmail.com	
5	Tanha Khan	tanha@gmail.com	
6	Arif Hossain	arif@gmail.com	
7	Farzana Noor	farzana@gmail.com	
8	Imran Ali	imran@gmail.com	
9	Sadia Rahim	sadia@gmail.com	
10	Naeem Islam	naeem@gmail.com	



2. List all students with their department and batch:

2. List all students with their department and batch

```
SELECT F_name, L_name, Department, Batch
FROM Student;
```

Output:

Result Grid



Filter Rows:

Export:

	F_name	L_name	Department	Batch
▶	Nusrat	Jahan	CSE	56
	Rafi	Hasan	CSE	56
	Mitu	Akter	EEE	57
	Sami	Rahman	CSE	57
	Tanha	Khan	BBA	58
	Arif	Hossain	EEE	56
	Farzana	Noor	CSE	57
	Imran	Ali	CSE	58
	Sadia	Rahim	BBA	59
	Naeem	Islam	EEE	56

57



3. Find students who scored more than 3.80 CGPA:

3. Find students who scored more than 3.80 CGPA

```
SELECT S.F_name, S.L_name, R.cgpa, R.grade
FROM Student S
JOIN Result R ON S.Student_id = R.Student_id
WHERE R.cgpa > 3.80;
```

Output:

Result Grid



Filter Rows:

	F_name	L_name	cgpa	grade
▶	Sami	Rahman	3.9	A
	Imran	Ali	3.85	A
	Naeem	Islam	3.95	A+

4. Find which teacher conducts which course:

4. Find which teacher conducts which course

```
SELECT T.F_name AS Teacher, C.Course_title
FROM Teacher T
JOIN Conducts Co ON T.Teacher_id = Co.Teacher_id
JOIN Course C ON Co.Course_id = C.Course_id;
```

Output:

Result Grid	Filter Rows:	Export:	Wrap Cell Conte
Teacher	Course_title		
▶ Shamim	Database Systems		
Fahim	Digital Logic		
Joya	Data Structures		
Sumon	Accounting		
Afsana	Electronics		
Mahfuz	AI Fundamentals		
Farid	Web Development		
Tahmid	Networking		
Mira	Microprocessor		
Rashid	Machine Learning		

5. Show all results with remarks and exam type:

5. Show all results with remarks and exam type

```
SELECT R.Result_id, S.F_name, S.L_name, R.grade, R.cgpa, RD.remarks, RD.Exam_type
FROM Result R
JOIN Student S ON R.Student_id = S.Student_id
JOIN Result_Details RD ON R.Result_id = RD.Result_id;
```

Output:

Result Grid		Filter Rows:			Export:	Wrap Cell Content:	
	Result_id	F_name	L_name	grade	cgpa	remarks	Exam_type
▶	1	Nusrat	Jahan	A	3.8	Excellent	Final
	2	Rafi	Hasan	A-	3.5	Good	Mid
	3	Mitu	Akter	A-	3.6	Very Good	Final
	4	Sami	Rahman	A	3.9	Outstanding	Final
	5	Tanha	Khan	A-	3.7	Satisfactory	Mid
	6	Arif	Hossain	B+	3.2	Average	Mid
	7	Farzana	Noor	B	3	Improved	Final
	8	Imran	Ali	A	3.85	Excellent	Final
	9	Sadia	Rahim	A-	3.4	Good	Mid
	10	Naeem	Islam	A+	3.95	Top Score	Final

6. List all students with their enrolled courses and course teacher:

6. List all students with their enrolled courses and course teacher

```
SELECT S.F_name AS Student, C.Course_title, T.F_name AS Teacher
FROM Enrolls E
JOIN Student S ON E.Student_id = S.Student_id
JOIN Course C ON E.Course_id = C.Course_id
JOIN Teacher T ON C.Teacher_id = T.Teacher_id;
```

Output:

Result Grid	Filter Rows:	Export:	Wrap Cel
Student	Course_title	Teacher	
Nusrat	Database Systems	Shamim	
Rafi	Digital Logic	Fahim	
Mitu	Data Structures	Joya	
Sami	Accounting	Sumon	
Tanha	Electronics	Afsana	
Arif	Tanha Fundamentals	Mahfuz	
Farzana	Web Development	Farid	
Imran	Networking	Tahmid	
Sadia	Microprocessor	Mira	
Naeem	Machine Learning	Rashid	

7. Find total number of students in each department:

```
7. Find total number of students in each department
SELECT Department, COUNT(*) AS Total_Students
FROM Student
GROUP BY Department;
```

Output:


Result Grid	Filter Rows:	Export:	Wrap Cell Content
Department	Total_Students		
CSE	5		
EEE	3		
BBA	2		

8. Find the highest CGPA and corresponding student name:

```
8. Find the highest CGPA and corresponding student name
SELECT S.F_name, S.L_name, R.cgpa
FROM Student S
JOIN Result R ON S.Student_id = R.Student_id
WHERE R.cgpa = (SELECT MAX(cgpa) FROM Result);
```

Output:

Result Grid



Filter Rows:

	F_name	L_name	cgpa
▶	Naeem	Islam	3.95

9. Show list of teachers from CSE department with their email:

9. Show list of teachers from CSE department with their email

```
SELECT T.F_name, T.L_name, TE.Teacher_Email  
FROM Teacher T  
JOIN Teacher_Email TE ON T.Teacher_id = TE.Teacher_id  
WHERE T.Department = 'CSE';
```

Output:

Result Grid	Filter Rows:	Export:	Wrap Cell Co
F_name	L_name	Teacher_Email	
Shamim	Ahmed	shamim@univ.com	
Fahim	Shamim	fahim@univ.com	
Mahfuz	Alam	mahfuz@univ.com	
Farid	Uddin	farid@univ.com	
Tahmid	Hasan	tahmid@univ.com	
Rashid	Hossain	rashid@univ.com	

10. Calculate average CGPA per department:

10. Calculate average CGPA per department

```
SELECT S.Department, ROUND(AVG(R.cgpa), 2) AS Avg_CGPA  
FROM Result R  
JOIN Student S ON R.Student_id = S.Student_id  
GROUP BY S.Department;
```

Output:

Result Grid	Filter Rows:	Export:	V
Department	Avg_CGPA		
CSE	3.61		
EEE	3.58		
BBA	3.55		

Outcome

- ❖ Error minimization and faster result processing.
- ❖ Improved data security and accuracy.
- ❖ Enhanced user accessibility via different result viewing formats.
- ❖ Reduced administrative workload and paper consumption.

Discussion

The Student Result Management System helps automate result processing and reduce human errors. It ensures accurate data handling and easy access for both teachers and students. By using a well-structured database with proper relationships, the system maintains data consistency and improves overall efficiency in managing academic results.

References

1. Silberschatz, A., Galvin, P. B., & Gagne, G. (2020). *Operating System Concepts* (10th ed.). Wiley.
2. Elmasri, R., & Navathe, S. B. (2016). *Fundamentals of Database Systems* (7th ed.). Pearson.
3. Ramakrishnan, R., & Gehrke, J. (2003). *Database Management Systems* (3rd ed.). McGraw-Hill.
4. W3Schools. (n.d.). *SQL Tutorial*. Retrieved from <https://www.w3schools.com/sql/>
5. TutorialsPoint. (n.d.). *DBMS Tutorial*. Retrieved from <https://www.tutorialspoint.com/dbms/>
6. GeeksforGeeks. (n.d.). *Student Result Management System Project*. Retrieved from <https://www.geeksforgeeks.org/>
7. GitHub. (n.d.). *Student Result Management System Projects*. Retrieved from <https://github.com/>
8. IEEE Xplore Digital Library. (n.d.). *Research Papers on Student Result Management Systems*. Retrieved from <https://ieeexplore.ieee.org/>

Git Hub Links

<https://github.com/tamim65k>

<https://github.com/Montasirpeal>

<https://github.com/paransha1>

<https://github.com/nusrattasfi>