

COLLEGE FEEDBACK SYSTEM

PROJECT DOCUMENT

BY

- DHARSHINI S

Entity Relationship Diagram

Problem Statement: College Feedback System

A College Feedback System allows students to provide feedback on the professors they have attended classes with. This system should be able to record feedback for each student and professor and manage various aspects such as students, professors, departments, and feedback.

Task 1: Requirements of the Application

The College Feedback System requires entities for departments, students, professors, feedback, and reports.

Task 2: Entity Set

1. Department
2. Student
3. Professor
4. Feedback
5. Report

Task 3: Map Out the Attributes of the Entities

1. **Department**
 - DeptID (Primary Key)
 - DeptName

2. Student

- StudentID (Primary Key)
- StudentName
- Email
- Phone
- DepartmentID (Foreign Key)

3. Professor

- ProfID (Primary Key)
- ProfName
- Email
- Phone
- DepartmentID (Foreign Key)

4. Feedback

- FeedbackID (Primary Key)
- FeedbackText
- StudentID (Foreign Key)
- ProfID (Foreign Key)

5. Report

- ReportID (Primary Key)
- ReportText
- FeedbackID (Foreign Key)

Task 4: Types of Attributes

Attributes can be classified into different types:

- **Simple Attribute:** An attribute that cannot be divided further

Example:

DeptID, DeptName

StudentID, StudentName

ProfID, ProfName

FeedbackID, FeedbackText

ReportID, ReportText

- **Composite Attribute:** An attribute that can be broken down into smaller parts
- **Derived Attribute:** An attribute whose value can be derived from other attributes
- **Multi-valued Attribute:** An attribute that can have multiple values.

Task 5: Key and Foreign Attributes

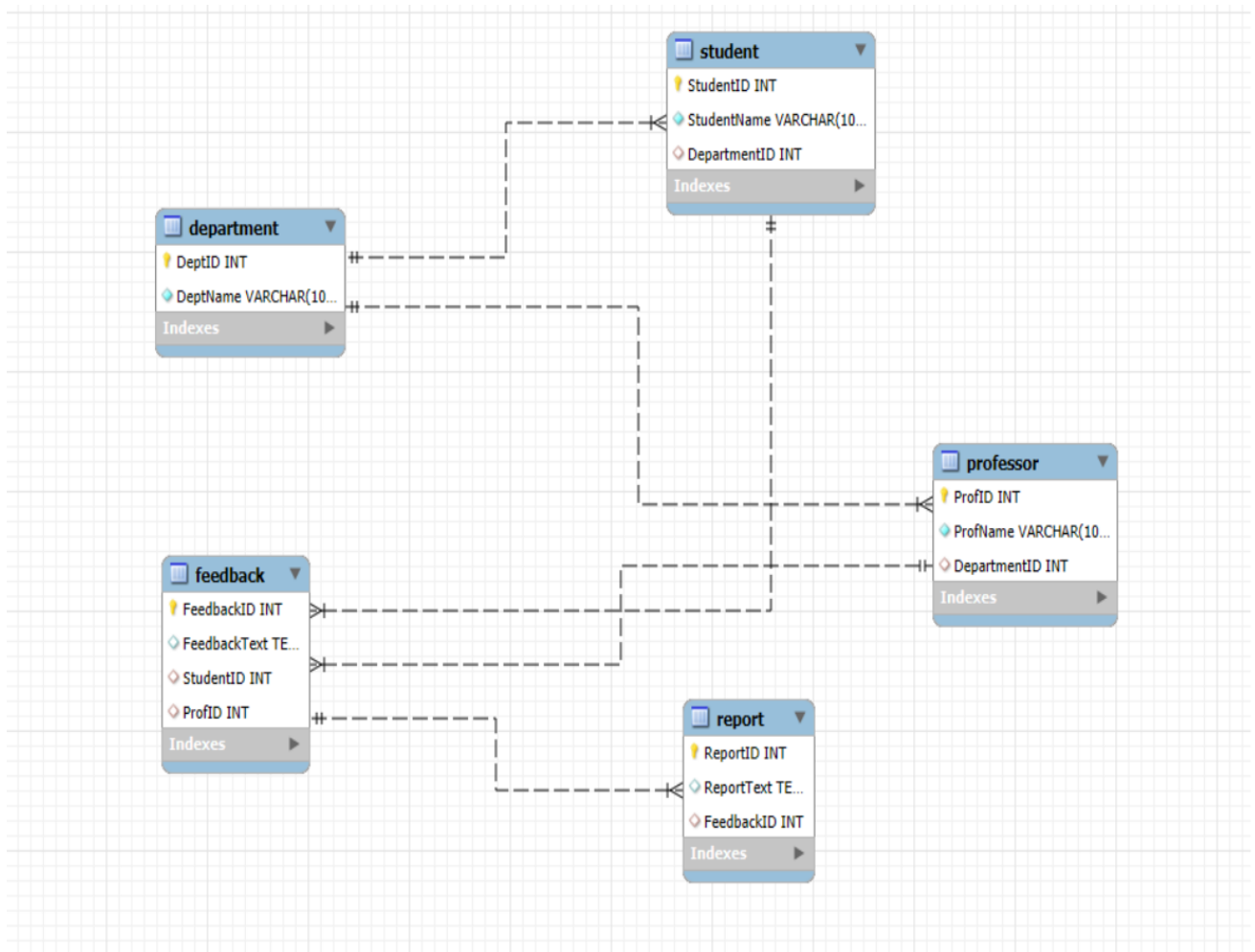
- **Primary Keys:**
 - DeptID in Department Table
 - StudentID in Student Table
 - ProfID in Professor Table
 - FeedbackID in Feedback Table
 - ReportID in Report Table

- **Foreign Keys:**
 - DepartmentID in Student Table and Professor Table references DeptID in Department Table.
 - StudentID in Feedback Table references StudentID in Student Table.
 - ProfID in Feedback Table references ProfID in Professor Table.
 - FeedbackID in Report Table references FeedbackID in Feedback Table.

Task 6: Relationships

- **Department → Student:** One-to-Many (1:N)
- **Department → Professor:** One-to-Many (1:N)
- **Student → Feedback:** One-to-Many (1:N)
- **Professor → Feedback:** One-to-Many (1:N)
- **Feedback → Report:** One-to-One (1:1)

Task 7: ER diagram in Crows'foot.



Normalization

1. Identify First Normal Form (1NF) Issues

Definition of 1NF:

- A table is in 1NF if:
 - All columns contain atomic (indivisible) values.

- There are no repeating groups or multivalued attributes.

Issues in the Feedback Table:

- If the Feedback table has multivalued attributes (e.g., multiple feedback texts for a single FeedbackID) or repeating groups, it violates 1NF.

Example of a violation:

FeedbackID	StudentID	FeedbackText	ProfID
1	101	"Good course", "Well done"	201

Here, FeedbackText contains multiple values in one column, which violates 1NF.

2. Convert to First Normal Form (1NF)

Solution:

To achieve 1NF, separate multivalued attributes into distinct rows:

FeedbackID	StudentID	FeedbackText	ProfID
1	101	"Good course"	201
2	101	"Well done"	201

Now, each cell contains a single atomic value, satisfying 1NF.

3. Identify Second Normal Form (2NF) Issues

Definition of 2NF:

- A table is in 2NF if:
 - It is already in 1NF.
 - There are no **partial dependencies** (a non-prime attribute depending only on part of a composite primary key).

Issues in the Feedback Table:

If the primary key is composite (e.g., FeedbackID + StudentID), attributes like ProfID or FeedbackText might depend only on FeedbackID, not the full compositekey.

Example of a partial dependency:

FeedbackID	StudentID	FeedbackText	ProfID
1	101	"Good course"	201

Here, ProfID depends only on FeedbackID and not on the entire composite key.

4. Convert to Second Normal Form (2NF)

Solution:

To achieve 2NF, remove partial dependencies by creating separate tables.

1. Feedback Table (Feedback-specific details):

FeedbackID	FeedbackText	ProfID
1	"Good course"	201
2	"Well done"	201

2. Student_Feedback Table (Relationship between students and feedback):

FeedbackID	StudentID
1	101
2	101

Now, non-prime attributes depend on the whole key, not part of it.

5. Identify Third Normal Form (3NF) Issues

Definition of 3NF:

- A table is in 3NF if:
 - It is already in 2NF.
 - There are no **transitive dependencies** (non-prime attributes depending on other non-prime attributes).

Issues in the Feedback Table:

If ProfID determines another attribute like DepartmentID, then DepartmentID indirectly depends on the primary key through ProfID.

Example of a transitive dependency:

FeedbackID	FeedbackText	ProfID	DepartmentID
1	"Good course"	"Good course"	10

Here, DepartmentID depends on ProfID, not directly on the primary key (FeedbackID).

6. Convert to Third Normal Form (3NF)

Solution:

To achieve 3NF, remove transitive dependencies by creating separate tables:

1. Feedback Table (Feedback-specific details):

FeedbackID	FeedbackText	ProfID
1	"Good course"	201
2	"Well done"	201

2. Professor Table (Professor-specific details):

ProfID	DepartmentID
201	10

DDL QUERIES

1. Create the Department table:

- o DeptID: The primary key, uniquely identifies each department.
- o DeptName: Stores the name of the department, cannot be null.

Query:

```
CREATE TABLE Department(  
    DeptID INT PRIMARY KEY,  
    DeptName VARCHAR(100) NOT NULL  
);
```

Output:



The screenshot shows a table structure for 'Department'. It has two columns: 'DeptID' and 'DeptName'. 'DeptID' is marked as the primary key with a star icon and is set to 'NULL'. 'DeptName' is also set to 'NULL'.

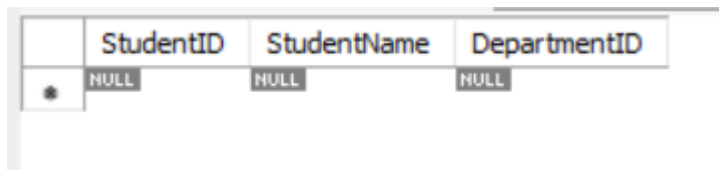
	DeptID	DeptName
*	NULL	NULL

2. Create the Student table:

- o StudentID: The primary key, uniquely identifies each student.
- o StudentName: Stores the student's name, cannot be null.
- o DepartmentID: Foreign key referencing the DeptID in the Department table.

Query:

```
CREATE TABLE Student(  
StudentID INT PRIMARY KEY,  
StudentName VARCHAR(100) NOT NULL,  
DepartmentID INT,  
FOREIGN KEY(DepartmentID) REFERENCES Department(DeptID)  
);
```

Output:

	StudentID	StudentName	DepartmentID
*	NULL	NULL	NULL

3. Create the Professor table:

- o ProfID: The primary key, uniquely identifies each professor.
- o ProfName: Stores the professor's name, cannot be null.
- o DepartmentID: Foreign key referencing the DeptID in the Department table.

Query:

```
CREATE TABLE Professor(  
ProfID INT PRIMARY KEY,  
ProfName VARCHAR(100) NOT NULL,  
DepartmentID INT,  
FOREIGN KEY(DepartmentID) REFERENCES Department(DeptID)  
);
```

Output:

	ProfID	ProfName	DepartmentID
*	NULL	NULL	NULL

4. Create the Feedback table:

- o FeedbackID: The primary key, uniquely identifies each feedback.
- o FeedbackText: Stores the feedback text.
- o StudentID: Foreign key referencing the StudentID in the Student table.
- o ProfID: Foreign key referencing the ProfID in the Professor table.

Query:

```
CREATE TABLE Feedback(
FeedbackID INT PRIMARY KEY,
FeedbackText TEXT,
StudentID INT,
ProfID INT,
FOREIGN KEY(StudentID) REFERENCES Student(StudentID),
FOREIGN KEY(ProfID) REFERENCES Professor(ProfID)
);
```

Output:

	FeedbackID	FeedbackText	StudentID	ProfID
*	NULL	NULL	NULL	NULL

5. Create the Report table:

- o ReportID: The primary key, uniquely identifies each report.
- o ReportText: Stores the report text.

- o FeedbackID: Foreign key referencing the FeedbackID in the Feedback table.

Query:

```
CREATE TABLE Report(  
    ReportID INT PRIMARY KEY,  
    ReportText TEXT,  
    FeedbackID INT,  
    FOREIGN KEY(FeedbackID) REFERENCES Feedback(FeedbackID)  
);
```

Output:

	ReportID	ReportText	FeedbackID
*	NULL	NULL	NULL

DML QUERIES

1. Write a DML query to Insert a new department into the Department table

Query:

```
INSERT INTO Department (DeptID, DeptName)  
  
VALUES  
  
(1, 'Computer Science'),  
  
(2, 'Mathematics'),
```

(3, 'Physics'),
 (4, 'Biology'),
 (5, 'Chemistry'),
 (6, 'Economics'),
 (7, 'History'),
 (8, 'Political Science'),
 (9, 'English Literature'),
 (10, 'Philosophy'),
 (11, 'Art'),
 (12, 'Music'),
 (13, 'Psychology'),
 (14, 'Geology'),
 (15, 'Environmental Science');**Output:**

	DeptID	DeptName
▶	1	Computer Science
	2	Mathematics
	3	Physics
	4	Biology
	5	Chemistry
	6	Economics
	7	History
	8	Political Science
	9	English Literature
	10	Philosophy
	11	Art
	12	Music
	13	Psychology
	14	Geology
	15	Environmental Sc...
•	NULL	NULL

3. Write a DML query to Insert a new student into the Student table

Query:

```
INSERT INTO Student (StudentID, StudentName, DepartmentID)
```

```
VALUES
```

```
(101, 'John Doe', 1),
```

```
(102, 'Alice Johnson', 2),
```

```
(103, 'Bob Brown', 3),
```

```
(104, 'Charlie Davis', 4),
```

```
(105, 'Diana Adams', 5),
```

```
(106, 'Ethan Wright', 6),
```

```
(107, 'Fiona Clark', 7),
```

```
(108, 'George Harris', 8),
```

```
(109, 'Hannah Lewis', 9),
```

```
(110, 'Irene Martinez', 10),
```

```
(111, 'Jack Moore', 11),
```

```
(112, 'Kara Bell', 12),
```

```
(113, 'Liam Parker', 13),
```

```
(114, 'Mia Evans', 14),
```

```
(115, 'Noah Scott', 15);
```

Output:

	StudentID	StudentName	DepartmentID
▶	101	John Doe	1
	102	Alice Johnson	2
	103	Bob Brown	3
	104	Charlie Davis	4
	105	Diana Adams	5
	106	Ethan Wright	6
	107	Fiona Clark	7
	108	George Harris	8
	109	Hannah Lewis	9
	110	Irene Martinez	10
	111	Jack Moore	11
	112	Kara Bell	12
	113	Liam Parker	13
	114	Mia Evans	14
	115	Noah Scott	15
*	NULL	NULL	NULL

3. Write a DML query to Insert a new professor into the Professor table

Query:

```
INSERT INTO Professor (ProfID, ProfName, DepartmentID)
```

```
VALUES
```

```
(201, 'Dr. Smith', 1),
```

```
(202, 'Dr. Taylor', 2),
```

```
(203, 'Dr. Carter', 3),
```

```
(204, 'Dr. Martinez', 4),
```

```
(205, 'Dr. Lee', 5),
```

```
(206, 'Dr. Brown', 6),
```

```
(207, 'Dr. Wilson', 7),
```


(208, 'Dr. White', 8),
(209, 'Dr. Green', 9),
(210, 'Dr. Hall', 10),
(211, 'Dr. Young', 11),
(212, 'Dr. Allen', 12),
(213, 'Dr. Walker', 13),
(214, 'Dr. Hernandez', 14),
(215, 'Dr. King', 15);

Output:

	ProfID	ProfName	DepartmentID
▶	201	Dr. Smith	1
	202	Dr. Taylor	2
	203	Dr. Carter	3
	204	Dr. Martinez	4
	205	Dr. Lee	5
	206	Dr. Brown	6
	207	Dr. Wilson	7
	208	Dr. White	8
	209	Dr. Green	9
	210	Dr. Hall	10
	211	Dr. Young	11
	212	Dr. Allen	12
	213	Dr. Walker	13
	214	Dr. Hernan...	14
	215	Dr. King	15
•	NULL	NULL	NULL

4. Write a DML query to Insert a new feedback into the Feedback table

Query:

INSERT INTO Feedback (FeedbackID, FeedbackText, StudentID, ProfID)

VALUES

(301, 'Great lecture!', 101, 201),
 (302, 'Helpful and clear explanations.', 102, 202),
 (303, 'Engaging lecture style!', 103, 203),
 (304, 'Simplified complex topics.', 104, 204),
 (305, 'Motivational and inspiring.', 105, 205),
 (306, 'Innovative teaching approach.', 106, 206),
 (307, 'Clear and concise delivery.', 107, 207),
 (308, 'Excellent knowledge.', 108, 208),
 (309, 'Very interactive session.', 109, 209),
 (310, 'Effective use of examples.', 110, 210),
 (311, 'Encouraged participation.', 111, 211),
 (312, 'Focused on practical applications.', 112, 212),
 (313, 'Detailed and thorough.', 113, 213),
 (314, 'Engaging visuals used.', 114, 214),
 (315, 'Encouraged critical thinking.', 115, 215);

Output:

	FeedbackID	FeedbackText	StudentID	ProfID
▶	301	Great lecture!	101	201
	302	Helpful and clear explanations.	102	202
	303	Engaging lecture style!	103	203
	304	Simplified complex topics.	104	204
	305	Motivational and inspiring.	105	205
	306	Innovative teaching approach.	106	206
	307	Clear and concise delivery.	107	207
	308	Excellent knowledge.	108	208
	309	Very interactive session.	109	209
	310	Effective use of examples.	110	210
	311	Encouraged participation.	111	211
	312	Focused on practical applicati...	112	212
	313	Detailed and thorough.	113	213
	314	Engaging visuals used.	114	214
	315	Encouraged critical thinking.	115	215
●	NULL	NULL	NULL	NULL

5. Write a DML query to Insert a new report into the Report table

Query:

```
INSERT INTO Report (ReportID, ReportText, FeedbackID)
```

```
VALUES
```

```
(401, 'Student praised teaching clarity.', 301),  
(402, 'Feedback highlighted engaging style.', 302),  
(403, 'Simplification of topics appreciated.', 303),  
(404, 'Motivational delivery appreciated.', 304),  
(405, 'Student liked innovative approach.', 305),  
(406, 'Positive feedback on delivery.', 306),  
(407, 'Encouraging feedback on knowledge.', 307),  
(408, 'Highly interactive session noted.', 308),  
(409, 'Good use of examples noted.', 309),  
(410, 'Participation encouragement appreciated.', 310),  
(411, 'Practical applications were helpful.', 311),  
(412, 'Thorough explanation noted.', 312),  
(413, 'Great use of engaging visuals.', 313),  
(414, 'Critical thinking emphasis appreciated.', 314),  
(415, 'Positive feedback on session.', 315);
```

Output:

	ReportID	ReportText	FeedbackID
▶	401	Student praised teaching clarity.	301
	402	Feedback highlighted engaging style.	302
	403	Simplification of topics appreciated.	303
	404	Motivational delivery appreciated.	304
	405	Student liked innovative approach.	305
	406	Positive feedback on delivery.	306
	407	Encouraging feedback on knowledge.	307
	408	Highly interactive session noted.	308
	409	Good use of examples noted.	309
	410	Participation encouragement appreciated.	310
	411	Practical applications were helpful.	311
	412	Thorough explanation noted.	312
	413	Great use of engaging visuals.	313
	414	Critical thinking emphasis appreciated.	314
	415	Positive feedback on session.	315
•	NULL	NULL	NULL

6. Write a DML query to Delete a department from the Department table

Query:

DELETE FROM Department WHERE DeptID = 1;

Output:

	DeptID	DeptName
▶	2	Mathematics
	3	Physics
	4	Biology
	5	Chemistry
	6	Economics
	7	History
	8	Political Science
	9	English Literature
	10	Philosophy
	11	Art
	12	Music
	13	Psychology
	14	Geology
	15	Environmental S...
•	NULL	NULL

7. Write a DML query to Delete a student from the Student table

Query:

DELETE FROM Student WHERE StudentID = 101;

Output:

	StudentID	StudentName	DepartmentID
▶	102	Alice Johnson	2
	103	Bob Brown	3
	104	Charlie Davis	4
	105	Diana Adams	5
	106	Ethan Wright	6
	107	Fiona Clark	7
	108	George Harris	8
	109	Hannah Lewis	9
	110	Irene Martinez	10
	111	Jack Moore	11
	112	Kara Bell	12
	113	Liam Parker	13
	114	Mia Evans	14
	115	Noah Scott	15
•	NULL	NULL	NULL

8. Write a DML query to Delete a professor from the Professor table

Query:

DELETE FROM Professor WHERE ProfID = 201;

Output:

	ProfID	ProfName	DepartmentID
▶	202	Dr. Taylor	2
	203	Dr. Carter	3
	204	Dr. Martinez	4
	205	Dr. Lee	5
	206	Dr. Brown	6
	207	Dr. Wilson	7
	208	Dr. White	8
	209	Dr. Green	9
	210	Dr. Hall	10
	211	Dr. Young	11
	212	Dr. Allen	12
	213	Dr. Walker	13
	214	Dr. Hernan...	14
	215	Dr. King	15
•	NULL	NULL	NULL

9. Write a DML query to Delete feedback from the Feedback table

Query:

DELETE FROM Feedback WHERE FeedbackID = 301;

Output:

	FeedbackID	FeedbackText	StudentID	ProfID
▶	302	Helpful and clear explanations.	102	202
	303	Engaging lecture style!	103	203
	304	Simplified complex topics.	104	204
	305	Motivational and inspiring.	105	205
	306	Innovative teaching approach.	106	206
	307	Clear and concise delivery.	107	207
	308	Excellent knowledge.	108	208
	309	Very interactive session.	109	209
	310	Effective use of examples.	110	210
	311	Encouraged participation.	111	211
	312	Focused on practical applicati...	112	212
	313	Detailed and thorough.	113	213
	314	Engaging visuals used.	114	214
	315	Encouraged critical thinking.	115	215
•	NULL	NULL	NULL	NULL

10. Write a DML query to Delete a report from the Report table

Query:

DELETE FROM Report WHERE FeedbackID = 301;

Output:

	ReportID	ReportText	FeedbackID
▶	402	Feedback highlighted engaging style.	302
	403	Simplification of topics appreciated.	303
	404	Motivational delivery appreciated.	304
	405	Student liked innovative approach.	305
	406	Positive feedback on delivery.	306
	407	Encouraging feedback on knowledge.	307
	408	Highly interactive session noted.	308
	409	Good use of examples noted.	309
	410	Participation encouragement appreciated.	310
	411	Practical applications were helpful.	311
	412	Thorough explanation noted.	312
	413	Great use of engaging visuals.	313
	414	Critical thinking emphasis appreciated.	314
	415	Positive feedback on session.	315
•	NULL	NULL	NULL

11. Write a DML query to Update the name of a department in the Department table

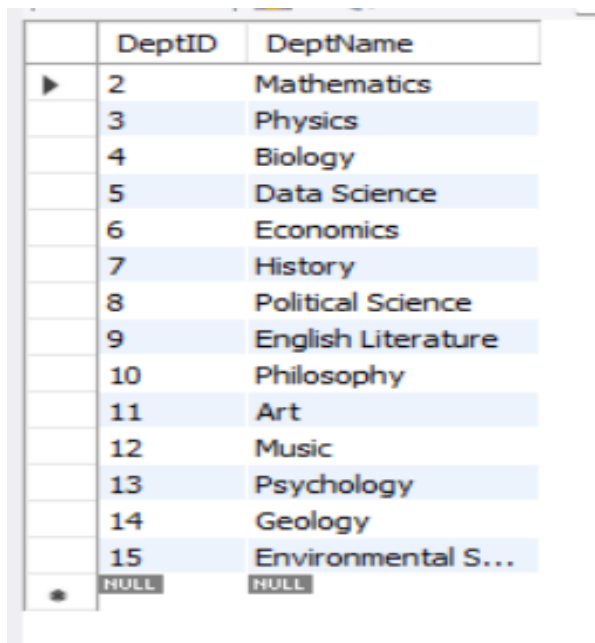
Query:

```
UPDATE Department
```

```
SET DeptName = 'Data Science'
```

```
WHERE DeptID = 5;
```

Output:



The screenshot shows a database table with two columns: DeptID and DeptName. The table contains 15 rows of department names. The row for DeptID 5, 'Data Science', is highlighted in blue, indicating it is the current record being viewed or edited. The other rows are listed in a standard font. The table is displayed in a window with a scroll bar on the left.

DeptID	DeptName
2	Mathematics
3	Physics
4	Biology
5	Data Science
6	Economics
7	History
8	Political Science
9	English Literature
10	Philosophy
11	Art
12	Music
13	Psychology
14	Geology
15	Environmental S...
HULL	HULL

12. Write a DML query to Update the name of a student in the Student table

Query:

```
UPDATE Student
```

```
SET StudentName = 'Johnathan Doe'
```

```
WHERE StudentID = 104;
```

Output:

	StudentID	StudentName	DepartmentID
▶	102	Alice Johnson	2
	103	Bob Brown	3
	104	Johnathan Doe	4
	105	Diana Adams	5
	106	Ethan Wright	6
	107	Fiona Clark	7
	108	George Harris	8
	109	Hannah Lewis	9
	110	Irene Martinez	10
	111	Jack Moore	11
	112	Kara Bell	12
	113	Liam Parker	13
	114	Mia Evans	14
	115	Noah Scott	15
✱	NULL	NULL	NULL

13. Write a DML query to Update the department of a professor in the Professor table

Query:

UPDATE Professor

SET DepartmentID = 2

WHERE ProfID = 210;

Output:

	ProfID	ProfName	DepartmentID
▶	202	Dr. Taylor	2
	203	Dr. Carter	3
	204	Dr. Martinez	4
	205	Dr. Lee	5
	206	Dr. Brown	6
	207	Dr. Wilson	7
	208	Dr. White	8
	209	Dr. Green	9
	210	Dr. Hall	2
	211	Dr. Young	11
	212	Dr. Allen	12
	213	Dr. Walker	13
	214	Dr. Hernan...	14
	215	Dr. King	15
✱	NULL	NULL	NULL

14. Write a DML query to Update the feedback text in the Feedback table

Query:

```
UPDATE Feedback
```

```
SET FeedbackText = 'Exceptional teaching style!'
```

```
WHERE FeedbackID = 309;
```

Output:

	FeedbackID	FeedbackText	StudentID	ProfID
▶	302	Helpful and clear explanations.	102	202
	303	Engaging lecture style!	103	203
	304	Simplified complex topics.	104	204
	305	Motivational and inspiring.	105	205
	306	Innovative teaching approach.	106	206
	307	Clear and concise delivery.	107	207
	308	Excellent knowledge.	108	208
	309	Exceptional teaching style!	109	209
	310	Effective use of examples.	110	210
	311	Encouraged participation.	111	211
	312	Focused on practical applicati...	112	212
	313	Detailed and thorough.	113	213
	314	Engaging visuals used.	114	214
	315	Encouraged critical thinking.	115	215
•	NULL	NULL	NULL	NULL

15. Write a DML query to Update the report text in the Report table

Query:

```
UPDATE Report
```

```
SET ReportText = 'Student praised for detailed explanation.'
```

```
WHERE ReportID = 405;
```

Output:

	ReportID	ReportText	FeedbackID
▶	402	Feedback highlighted engaging style.	302
	403	Simplification of topics appreciated.	303
	404	Motivational delivery appreciated.	304
	405	Student praised for detailed explanation.	305
	406	Positive feedback on delivery.	306
	407	Encouraging feedback on knowledge.	307
	408	Highly interactive session noted.	308
	409	Good use of examples noted.	309
	410	Participation encouragement appreciated.	310
	411	Practical applications were helpful.	311
	412	Thorough explanation noted.	312
	413	Great use of engaging visuals.	313
	414	Critical thinking emphasis appreciated.	314
	415	Positive feedback on session.	315
●	NULL	NULL	NULL

DQL QUERIES

1. Retrieve All Departments

Query:

```
SELECT DeptName FROM Department;
```

Output:

	DeptName
▶	Mathematics
	Physics
	Biology
	Data Science
	Economics
	History
	Political Science
	English Literature
	Philosophy
	Art
	Music
	Psychology
	Geology
	Environmental S...

2. List All Students

Query:

```
SELECT StudentName FROM Student;
```

Output:

	StudentName
▶	Alice Johnson
	Bob Brown
	Johnathan Doe
	Diana Adams
	Ethan Wright
	Fiona Clark
	George Harris
	Hannah Lewis
	Irene Martinez
	Jack Moore
	Kara Bell
	Liam Parker
	Mia Evans
	Noah Scott

3. Display All Professors

Query:

```
SELECT ProfName FROM Professor;
```

Output:

	ProfName
▶	Dr. Taylor
	Dr. Carter
	Dr. Martinez
	Dr. Lee
	Dr. Brown
	Dr. Wilson
	Dr. White
	Dr. Green
	Dr. Hall
	Dr. Young
	Dr. Allen
	Dr. Walker
	Dr. Hernan...
	Dr. King

4. Retrieve All Feedback

Query:

```
SELECT FeedbackText FROM Feedback;
```

Output:

	FeedbackText
▶	Helpful and clear explanations.
	Engaging lecture style!
	Simplified complex topics.
	Motivational and inspiring.
	Innovative teaching approach.
	Clear and concise delivery.
	Excellent knowledge.
	Exceptional teaching style!
	Effective use of examples.
	Encouraged participation.
	Focused on practical applicati...
	Detailed and thorough.
	Engaging visuals used.
	Encouraged critical thinking.

5. Show All Reports

Query:

```
SELECT ReportText FROM Report;
```

Output:

	ReportText
▶	Feedback highlighted engaging style.
	Simplification of topics appreciated.
	Motivational delivery appreciated.
	Student praised for detailed explanation.
	Positive feedback on delivery.
	Encouraging feedback on knowledge.
	Highly interactive session noted.
	Good use of examples noted.
	Participation encouragement appreciated.
	Practical applications were helpful.
	Thorough explanation noted.
	Great use of engaging visuals.
	Critical thinking emphasis appreciated.
	Positive feedback on session.

6. List All Feedback with Approval Status

Query:

```
SELECT FeedbackText FROM Feedback;
```

Output:

	FeedbackText
▶	Helpful and clear explanations.
	Engaging lecture style!
	Simplified complex topics.
	Motivational and inspiring.
	Innovative teaching approach.
	Clear and concise delivery.
	Excellent knowledge.
	Exceptional teaching style!
	Effective use of examples.
	Encouraged participation.
	Focused on practical applicati...
	Detailed and thorough.
	Engaging visuals used.
	Encouraged critical thinking.

7. Display All Feedback by Students

Query:

```
SELECT FeedbackText
```

```
FROM Feedback
```

```
JOIN Student ON Feedback.StudentID = Student.StudentID;
```

Output:

	FeedbackText
▶	Helpful and clear explanations.
	Engaging lecture style!
	Simplified complex topics.
	Motivational and inspiring.
	Innovative teaching approach.
	Clear and concise delivery.
	Excellent knowledge.
	Exceptional teaching style!
	Effective use of examples.
	Encouraged participation.
	Focused on practical applicati...
	Detailed and thorough.
	Engaging visuals used.
	Encouraged critical thinking.

8. List All Feedback by Professors

Query:

```
SELECT FeedbackText
```

```
FROM Feedback
```

```
JOIN Professor ON Feedback.ProfID = Professor.ProfID;
```

Output:

	FeedbackText
▶	Helpful and clear explanations.
	Engaging lecture style!
	Simplified complex topics.
	Motivational and inspiring.
	Innovative teaching approach.
	Clear and concise delivery.
	Excellent knowledge.
	Exceptional teaching style!
	Effective use of examples.
	Encouraged participation.
	Focused on practical applicati...
	Detailed and thorough.
	Engaging visuals used.
	Encouraged critical thinking.

9. Retrieve Student Names with Their Department

Query:

```
SELECT Student.StudentName, Department.DeptName
```

```
FROM Student
```

```
JOIN Department ON Student.DepartmentID = Department.DeptID;
```

Output:

	StudentName	DeptName
▶	Alice Johnson	Mathematics
	Bob Brown	Physics
	Johnathan Doe	Biology
	Diana Adams	Data Science
	Ethan Wright	Economics
	Fiona Clark	History
	George Harris	Political Science
	Hannah Lewis	English Literature
	Irene Martinez	Philosophy
	Jack Moore	Art
	Kara Bell	Music
	Liam Parker	Psychology
	Mia Evans	Geology
	Noah Scott	Environmental S...

10. Display Feedbacks without Associated Reports

Query:

```
SELECT FeedbackText ,ReportText
```

```
FROM Feedback
```

```
join Report ON Feedback.FeedbackID = Report.FeedbackID
```

Output:

	FeedbackText	ReportText
▶	Helpful and clear explanations.	Feedback highlighted engaging style.
	Engaging lecture style!	Simplification of topics appreciated.
	Simplified complex topics.	Motivational delivery appreciated.
	Motivational and inspiring.	Student praised for detailed explanation.
	Innovative teaching approach.	Positive feedback on delivery.
	Clear and concise delivery.	Encouraging feedback on knowledge.
	Excellent knowledge.	Highly interactive session noted.
	Exceptional teaching style!	Good use of examples noted.
	Effective use of examples.	Participation encouragement appreciated.
	Encouraged participation.	Practical applications were helpful.
	Focused on practical applicati...	Thorough explanation noted.
	Detailed and thorough.	Great use of engaging visuals.
	Engaging visuals used.	Critical thinking emphasis appreciated.
	Encouraged critical thinking.	Positive feedback on session.

Aggregate Functions

1. Write a SQL query to Retrieve the DeptName of all departments in uppercase.

Query:

```
SELECT UPPER(DeptName) AS DeptName_Uppercase
```

```
FROM Department;
```


Output:

	DeptName_Uppercase
▶	MATHEMATICS
	PHYSICS
	BIOLOGY
	DATA SCIENCE
	ECONOMICS
	HISTORY
	POLITICAL SCIENCE
	ENGLISH LITERATURE
	PHILOSOPHY
	ART
	MUSIC
	PSYCHOLOGY
	GEOLOGY
	ENVIRONMENTAL SCI...

2. Write a SQL query to Concatenate the FirstName and LastName of students, separated by a space.

Query:

```
SELECT  
  
StudentID,  
  
SUBSTRING_INDEX(StudentName, ' ', 1) AS FirstName,  
SUBSTRING_INDEX(StudentName, ' ', -1) AS LastName,  
  
CONCAT(SUBSTRING_INDEX(StudentName, ' ', 1), ' ',  
SUBSTRING_INDEX(StudentName, ' ', -1)) AS FullName  
FROM Student;
```

Output:

	StudentID	FirstName	LastName	FullName
▶	102	Alice	Johnson	Alice Johnson
	103	Bob	Brown	Bob Brown
	104	Johnathan	Doe	Johnathan Doe
	105	Diana	Adams	Diana Adams
	106	Ethan	Wright	Ethan Wright
	107	Fiona	Clark	Fiona Clark
	108	George	Harris	George Harris
	109	Hannah	Lewis	Hannah Lewis
	110	Irene	Martinez	Irene Martinez
	111	Jack	Moore	Jack Moore
	112	Kara	Bell	Kara Bell
	113	Liam	Parker	Liam Parker
	114	Mia	Evans	Mia Evans
	115	Noah	Scott	Noah Scott

3. Write a SQL query to Extract the first 5 characters of FeedbackText for each feedback.

Query:

```
SELECT LEFT(FeedbackText, 5) AS FeedbackSnippet  
FROM Feedback;
```

Output:

	FeedbackSnippet
▶	Helpf
	Engag
	Simpl
	Motiv
	Innov
	Clear
	Excel
	Excep
	Effec
	Encou
	Focus
	Detai
	Engag
	Encou

4. Write a SQL query to Calculate the length of the DeptName for all departments.

Query:

```
SELECT DeptName, LENGTH(DeptName) AS DeptNameLength  
FROM Department;
```

Output:

	DeptName	DeptNameLength
▶	Mathematics	11
	Physics	7
	Biology	7
	Data Science	12
	Economics	9
	History	7
	Political Science	17
	English Literature	18
	Philosophy	10
	Art	3
	Music	5
	Psychology	10
	Geology	7
	Environmental S...	21

5. Write a SQL query to Count the total number of feedback entries.

Query:

```
SELECT COUNT(*) AS TotalFeedbackCount  
FROM Feedback;
```

Output:

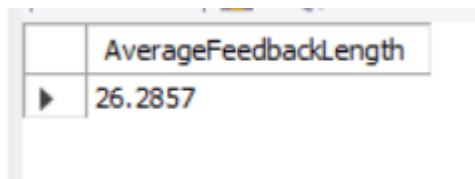
	TotalFeedbackCount
▶	14

6. Write a SQL query to Calculate the average length of feedback text.

Query:

```
SELECT AVG(LENGTH(FeedbackText)) AS AverageFeedbackLength  
FROM Feedback;
```

Output:



AverageFeedbackLength
26.2857

7. Write a SQL query to Retrieve the FeedbackText and the year it was given.

Alter & Update Query:

```
UPDATE Feedback
```

```
SET FeedbackDate = CASE
```

```
    WHEN FeedbackID = 302 THEN '2023-01-10'
```

```
    WHEN FeedbackID = 303 THEN '2023-02-15'
```

```
    WHEN FeedbackID = 304 THEN '2023-03-20'
```

```
    WHEN FeedbackID = 305 THEN '2023-04-25'
```

```
    WHEN FeedbackID = 306 THEN '2023-05-10'
```

```
    WHEN FeedbackID = 307 THEN '2023-06-15'
```

```
    WHEN FeedbackID = 308 THEN '2023-07-20'
```

```
    WHEN FeedbackID = 309 THEN '2023-08-25'
```

```

WHEN FeedbackID = 310 THEN '2023-09-10'

WHEN FeedbackID = 311 THEN '2023-10-15'

WHEN FeedbackID = 312 THEN '2023-11-20'

WHEN FeedbackID = 313 THEN '2023-12-25'

WHEN FeedbackID = 314 THEN '2024-02-15'

WHEN FeedbackID = 315 THEN '2024-03-20'

ELSE FeedbackDate

END;

```

Query:

```

SELECT

FeedbackText,

YEAR(FeedbackDate) AS FeedbackYear

FROM

Feedback;

```

Output:

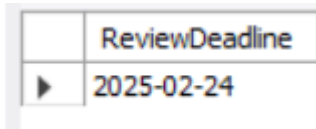
	FeedbackText	FeedbackYear
▶	Helpful and clear explanations.	2023
	Engaging lecture style!	2023
	Simplified complex topics.	2023
	Motivational and inspiring.	2023
	Innovative teaching approach.	2023
	Clear and concise delivery.	2023
	Excellent knowledge.	2023
	Exceptional teaching style!	2023
	Effective use of examples.	2023
	Encouraged participation.	2023
	Focused on practical applicati...	2023
	Detailed and thorough.	2023
	Engaging visuals used.	2024
	Encouraged critical thinking.	2024

8. Write a SQL query to Add 30 days to the current date to find the review deadline.

Query:

```
SELECT CURRENT_DATE + INTERVAL 30 DAY AS ReviewDeadline;
```

Output:



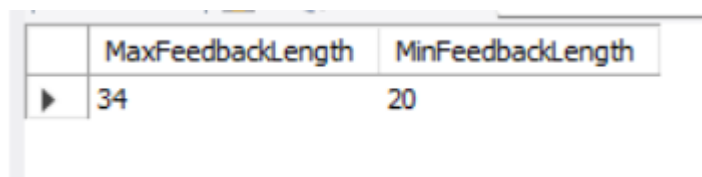
ReviewDeadline
2025-02-24

9. Write a SQL query to Find the maximum and minimum lengths of FeedbackText.

Query:

```
SELECT  
  
MAX(LENGTH(FeedbackText)) AS MaxFeedbackLength,  
  
MIN(LENGTH(FeedbackText)) AS MinFeedbackLength  
  
FROM Feedback;
```

Output:



MaxFeedbackLength	MinFeedbackLength
34	20

10. Write a SQL query to Calculate the total number of students in each department.

Query:

```
SELECT Department.DeptName, COUNT(Student.StudentID) AS  
TotalStudents
```

FROM Student

JOIN Department ON Student.DepartmentID = Department.DeptID

GROUP BY Department.DeptName;

Output:

	DeptName	TotalStudents
▶	Mathematics	1
	Physics	1
	Biology	1
	Data Science	1
	Economics	1
	History	1
	Political Science	1
	English Literature	1
	Philosophy	1
	Art	1
	Music	1
	Psychology	1
	Geology	1
	Environmental S...	1

11. Write a SQL query to Display 'Long Feedback' if the length of FeedbackText is greater than 100 characters, otherwise 'Short Feedback'.

Query:

SELECT

FeedbackText,

CASE

WHEN LENGTH(FeedbackText) > 100 THEN 'Long Feedback'

ELSE 'Short Feedback'

END AS FeedbackLengthCategory

FROM Feedback;

Output:

	FeedbackText	FeedbackLengthCategory
►	Helpful and clear explanations.	Short Feedback
	Engaging lecture style!	Short Feedback
	Simplified complex topics.	Short Feedback
	Motivational and inspiring.	Short Feedback
	Innovative teaching approach.	Short Feedback
	Clear and concise delivery.	Short Feedback
	Excellent knowledge.	Short Feedback
	Exceptional teaching style!	Short Feedback
	Effective use of examples.	Short Feedback
	Encouraged participation.	Short Feedback
	Focused on practical applicati...	Short Feedback
	Detailed and thorough.	Short Feedback
	Engaging visuals used.	Short Feedback
	Encouraged critical thinking.	Short Feedback

12. Write a SQL query to Provide the feedback status based on text length using IF.

Query:

SELECT

FeedbackText,

IF(LENGTH(FeedbackText) > 100, 'Long Feedback', 'Short Feedback')

AS FeedbackStatus

FROM Feedback;

Output:

	FeedbackText	FeedbackStatus
►	Helpful and clear explanations.	Short Feedback
	Engaging lecture style!	Short Feedback
	Simplified complex topics.	Short Feedback
	Motivational and inspiring.	Short Feedback
	Innovative teaching approach.	Short Feedback
	Clear and concise delivery.	Short Feedback
	Excellent knowledge.	Short Feedback
	Exceptional teaching style!	Short Feedback
	Effective use of examples.	Short Feedback
	Encouraged participation.	Short Feedback
	Focused on practical applicati...	Short Feedback
	Detailed and thorough.	Short Feedback
	Engaging visuals used.	Short Feedback
	Encouraged critical thinking.	Short Feedback

13. Write a SQL query to Replace NULL in ReportText with 'No Report'.

Query:

SELECT

COALESCE(ReportText, 'No Report') AS UpdatedReportText

FROM Report;

Output:

	UpdatedReportText
►	Feedback highlighted engaging style.
	Simplification of topics appreciated.
	Motivational delivery appreciated.
	Student praised for detailed explanation.
	Positive feedback on delivery.
	Encouraging feedback on knowledge.
	Highly interactive session noted.
	Good use of examples noted.
	Participation encouragement appreciated.
	Practical applications were helpful.
	Thorough explanation noted.
	Great use of engaging visuals.
	Critical thinking emphasis appreciated.
	Positive feedback on session.

14. Write a SQL query to Check for the presence of NULL in FeedbackText and provide a default message.

Query:

```
SELECT  
  
FeedbackText,  
  
CASE  
  
WHEN FeedbackText IS NULL THEN 'No Feedback Provided'  
  
ELSE FeedbackText  
  
END AS FeedbackMessage  
  
FROM Feedback;
```

Output:

	FeedbackText	FeedbackMessage
▶	Helpful and clear explanations.	Helpful and clear explanations.
	Engaging lecture style!	Engaging lecture style!
	Simplified complex topics.	Simplified complex topics.
	Motivational and inspiring.	Motivational and inspiring.
	Innovative teaching approach.	Innovative teaching approach.
	Clear and concise delivery.	Clear and concise delivery.
	Excellent knowledge.	Excellent knowledge.
	Exceptional teaching style!	Exceptional teaching style!
	Effective use of examples.	Effective use of examples.
	Encouraged participation.	Encouraged participation.
	Focused on practical applicati...	Focused on practical applicati...
	Detailed and thorough.	Detailed and thorough.
	Engaging visuals used.	Engaging visuals used.
	Encouraged critical thinking.	Encouraged critical thinking.

SET OPERATIONS

1. Combining Students and Professors

SQL query to Get a combined list of all StudentName and ProfName in the college, ensuring each name appears only once.

```
SELECT StudentName AS Name FROM Student
```

```
UNION
```

```
SELECT ProfName AS Name FROM Professor;
```

Output:

	Name
▶	Alice Johnson
	Bob Brown
	Johnathan Doe
	Diana Adams
	Ethan Wright
	Fiona Clark
	George Harris
	Hannah Lewis
	Irene Martinez
	Jack Moore
	Kara Bell
	Liam Parker
	Mia Evans
	Noah Scott
	Dr. Taylor
	Dr. Carter
	Dr. Martinez
	Dr. Lee
	Dr. Brown

Dr. Wilson
Dr. White
Dr. Green
Dr. Hall
Dr. Young
Dr. Allen
Dr. Walker
Dr. Hernandez
Dr. King

2. Combining Students and Professors

SQL query to Get a combined list of all StudentName and ProfName in the college, including duplicates.

```
SELECT StudentName AS Name FROM Student
```

```
UNION ALL
```

```
SELECT ProfName AS Name FROM Professor;
```

Output:

	Name
▶	Alice Johnson
	Bob Brown
	Johnathan Doe
	Diana Adams
	Ethan Wright
	Fiona Clark
	George Harris
	Hannah Lewis
	Irene Martinez
	Jack Moore
	Kara Bell
	Liam Parker
	Mia Evans
	Noah Scott
	Dr. Taylor
	Dr. Carter
	Dr. Martinez
	Dr. Lee
	Dr. Brown
	Dr. Wilson
	Dr. White
	Dr. Green

	Dr. Hall
	Dr. Young
	Dr. Allen
	Dr. Walker
	Dr. Hernandez
	Dr. King

3. Common Departments Between Students and Professors

SQL query to Find the DepartmentID that is common to both students and professors.

```
SELECT DISTINCT DepartmentID
```

```
FROM Student
```

```
WHERE DepartmentID IN (SELECT DepartmentID FROM Professor);
```

Output:

	DepartmentID
▶	2
	3
	4
	5
	6
	7
	8
	9
	11
	12
	13
	14
	15

4. Departments with Students but No Professors

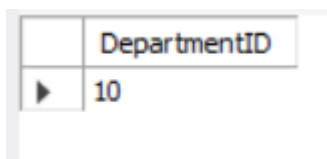
SQL query to Find the DepartmentID that have students but no professors.

```
SELECT DISTINCT DepartmentID
```

```
FROM Student
```

```
WHERE DepartmentID NOT IN (SELECT DepartmentID FROM  
Professor);
```

Output:



	DepartmentID
▶	10

5. Feedback and Reports without a Matching Feedback

SQL query to Get all FeedbackText and ReportText combined into one list, showing which are from feedback and which are from reports, including unmatched entries.

```
SELECT FeedbackText AS Text, 'Feedback' AS Source
```

```
FROM Feedback
```

```
UNION ALL
```

```
SELECT ReportText AS Text, 'Report' AS Source
```

```
FROM Report;
```

Output:

	Text	Source
▶	Helpful and clear explanations.	Feedback
	Engaging lecture style!	Feedback
	Simplified complex topics.	Feedback
	Motivational and inspiring.	Feedback
	Innovative teaching approach.	Feedback
	Clear and concise delivery.	Feedback
	Excellent knowledge.	Feedback
	Exceptional teaching style!	Feedback
	Effective use of examples.	Feedback
	Encouraged participation.	Feedback
	Focused on practical applicati...	Feedback
	Detailed and thorough.	Feedback
	Engaging visuals used.	Feedback
	Encouraged critical thinking.	Feedback
	Feedback highlighted engagin...	Report
	Simplification of topics appreci...	Report
	Motivational delivery apprecia...	Report
	Student praised for detailed e...	Report
	Positive feedback on delivery.	Report
	Encouraging feedback on kno...	Report
	Highly interactive session noted.	Report
	Good use of examples noted.	Report
	Participation encouragement ...	Report
	Practical applications were hel...	Report
	Thorough explanation noted.	Report
	Great use of engaging visuals.	Report
	Critical thinking emphasis appr...	Report
	Positive feedback on session.	Report

6. Find All Unique Texts from Feedback and Reports

SQL query to Get all unique texts from both FeedbackText and ReportText.

```
SELECT FeedbackText AS Text FROM Feedback
```

```
UNION
```

SELECT ReportText AS Text FROM Report;

Output:

	Text
▶	Helpful and clear explanations.
	Engaging lecture style!
	Simplified complex topics.
	Motivational and inspiring.
	Innovative teaching approach.
	Clear and concise delivery.
	Excellent knowledge.
	Exceptional teaching style!
	Effective use of examples.
	Encouraged participation.
	Focused on practical applicati...
	Detailed and thorough.
	Engaging visuals used.
	Encouraged critical thinking.
	Feedback highlighted engagin...
	Simplification of topics appreci...
	Motivational delivery apprecia...
	Student praised for detailed e...
	Positive feedback on delivery.
	Encouraging feedback on kno...
	Highly interactive session noted.
	Good use of examples noted.
	Participation encouragement ...
	Practical applications were hel...
	Thorough explanation noted.
	Great use of engaging visuals.
	Critical thinking emphasis appr...
	Positive feedback on session.

7. Feedbacks with Matching Reports

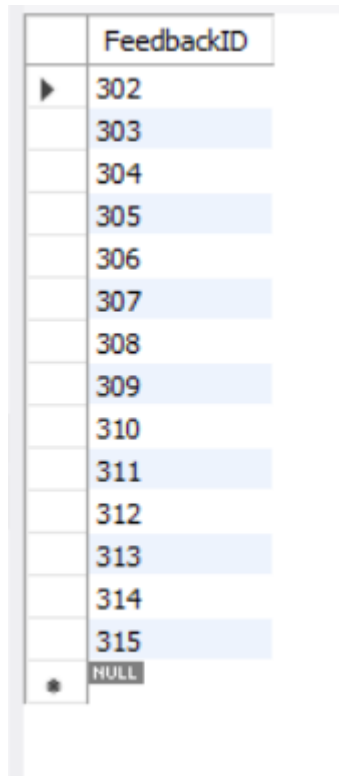
SQL query to Find all FeedbackID which have a matching report in the Report table.

SELECT FeedbackID

FROM Feedback

WHERE FeedbackID IN (SELECT FeedbackID FROM Report);

Output:



FeedbackID
302
303
304
305
306
307
308
309
310
311
312
313
314
315
NULL

8. Students without Feedback

SQL query to Find all StudentID from the Student table who have not given any feedback.

Alter Query:

INSERT INTO Department (DeptID, DeptName)

VALUES (16, 'Computer Science');

INSERT INTO Student (StudentID, StudentName, DepartmentID)

VALUES (116, 'Sophia Morgan', 16);

```
INSERT INTO Professor (ProfID, ProfName, DepartmentID)
```

```
VALUES (216, 'Dr. Anderson', 16);
```

```
INSERT INTO Feedback (FeedbackID, FeedbackText, StudentID,  
ProfID)
```

```
VALUES (316, NULL, 116, 216);
```

Alter Feedback output:

	FeedbackID	FeedbackText	StudentID	ProfID	FeedbackDate
▶	302	Helpful and clear explanations.	102	202	2023-01-10
	303	Engaging lecture style!	103	203	2023-02-15
	304	Simplified complex topics.	104	204	2023-03-20
	305	Motivational and inspiring.	105	205	2023-04-25
	306	Innovative teaching approach.	106	206	2023-05-10
	307	Clear and concise delivery.	107	207	2023-06-15
	308	Excellent knowledge.	108	208	2023-07-20
	309	Exceptional teaching style!	109	209	2023-08-25
	310	Effective use of examples.	110	210	2023-09-10
	311	Encouraged participation.	111	211	2023-10-15
	312	Focused on practical applicati...	112	212	2023-11-20
	313	Detailed and thorough.	113	213	2023-12-25
	314	Engaging visuals used.	114	214	2024-02-15
	315	Encouraged critical thinking.	115	215	2024-03-20
	316	NULL	116	216	NULL

Query:

```
SELECT StudentID
```

```
FROM Student
```

```
WHERE StudentID NOT IN (SELECT DISTINCT StudentID FROM  
Feedback);
```

Output:

	StudentID	StudentName
▶	116	Sophia Morgan
•	NULL	NULL

9. Professors without Feedback

SQL query to Find all ProfID from the Professor table who have not received any feedback.

```
SELECT ProfID
```

```
FROM Professor
```

```
WHERE ProfID NOT IN (SELECT DISTINCT ProfID FROM  
Feedback);
```

Output:

	ProfID	ProfName
▶	216	Dr. Anderson
•	NULL	NULL

10. Professors and Students without Feedback

SQL query to Find all ProfID and StudentID who are either professors or students but have not been involved in any feedback.

```
SELECT ProfID AS ID, 'Professor' AS Role
```

```
FROM Professor
```

```
WHERE ProfID NOT IN (SELECT DISTINCT ProfID FROM  
Feedback)
```

```
UNION ALL
```

```
SELECT StudentID AS ID, 'Student' AS Role
```

```
FROM Student
```

WHERE StudentID NOT IN (SELECT DISTINCT StudentID FROM Feedback);

Output:

	StudentID	StudentName
▶	116	Sophia Morgan
	216	Dr. Anderson

JOINS

1. Inner Join

Query to Retrieve all feedback along with the names of the students who gave them.

SELECT

F.FeedbackID,

F.FeedbackText,

S.StudentName

FROM

Feedback F

INNER JOIN

Student S

ON

F.StudentID = S.StudentID;

Output:

	FeedbackID	FeedbackText	StudentName
▶	302	Helpful and clear explanations.	Alice Johnson
	303	Engaging lecture style!	Bob Brown
	304	Simplified complex topics.	Johnathan Doe
	305	Motivational and inspiring.	Diana Adams
	306	Innovative teaching approach.	Ethan Wright
	307	Clear and concise delivery.	Fiona Clark
	308	Excellent knowledge.	George Harris
	309	Exceptional teaching style!	Hannah Lewis
	310	Effective use of examples.	Irene Martinez
	311	Encouraged participation.	Jack Moore
	312	Focused on practical applicati...	Kara Bell
	313	Detailed and thorough.	Liam Parker
	314	Engaging visuals used.	Mia Evans
	315	Encouraged critical thinking.	Noah Scott

2. Left Join

Query to Display all departments along with the names of students (if any) enrolled in each department.

SELECT

D.DeptID,

D.DeptName,

S.StudentName

FROM

Department D

LEFT JOIN

Student S

ON

D.DeptID = S.DepartmentID;

Output:

	DeptID	DeptName	StudentName
▶	2	Mathematics	Alice Johnson
	3	Physics	Bob Brown
	4	Biology	Johnathan Doe
	5	Data Science	Diana Adams
	6	Economics	Ethan Wright
	7	History	Fiona Clark
	8	Political Science	George Harris
	9	English Literature	Hannah Lewis
	10	Philosophy	Irene Martinez
	11	Art	Jack Moore
	12	Music	Kara Bell
	13	Psychology	Liam Parker
	14	Geology	Mia Evans
	15	Environmental S...	Noah Scott

3. Right Join

Query to List all professors along with the names of their departments.

SELECT

P.ProfID,

P.ProfName,

D.DeptName

FROM

Department D

RIGHT JOIN

Professor P

ON

D.DeptID = P.DepartmentID;

Output:

	ProfID	ProfName	DeptName
▶	202	Dr. Taylor	Mathematics
	203	Dr. Carter	Physics
	204	Dr. Martinez	Biology
	205	Dr. Lee	Data Science
	206	Dr. Brown	Economics
	207	Dr. Wilson	History
	208	Dr. White	Political Science
	209	Dr. Green	English Literature
	210	Dr. Hall	Mathematics
	211	Dr. Young	Art
	212	Dr. Allen	Music
	213	Dr. Walker	Psychology
	214	Dr. Hernan...	Geology
	215	Dr. King	Environmental S...

4. Full Outer Join

Query to Get a list of all feedback along with the names of students and professors involved, if available.

SELECT

F.FeedbackID,

F.FeedbackText,

S.StudentName,

P.ProfName

FROM

Feedback F

LEFT JOIN

Student S

ON

F.StudentID = S.StudentID

LEFT JOIN

Professor P

ON

F.ProfID = P.ProfID

UNION

SELECT

F.FeedbackID,

F.FeedbackText,

S.StudentName,

P.ProfName

FROM

Feedback F

RIGHT JOIN

Student S

ON

F.StudentID = S.StudentID

RIGHT JOIN

Professor P

ON

F.ProfID = P.ProfID;

Output:

	FeedbackID	FeedbackText	StudentName	ProfName
▶	302	Helpful and clear explanations.	Alice Johnson	Dr. Taylor
	303	Engaging lecture style!	Bob Brown	Dr. Carter
	304	Simplified complex topics.	Johnathan Doe	Dr. Martinez
	305	Motivational and inspiring.	Diana Adams	Dr. Lee
	306	Innovative teaching approach.	Ethan Wright	Dr. Brown
	307	Clear and concise delivery.	Fiona Clark	Dr. Wilson
	308	Excellent knowledge.	George Harris	Dr. White
	309	Exceptional teaching style!	Hannah Lewis	Dr. Green
	310	Effective use of examples.	Irene Martinez	Dr. Hall
	311	Encouraged participation.	Jack Moore	Dr. Young
	312	Focused on practical applicati...	Kara Bell	Dr. Allen
	313	Detailed and thorough.	Liam Parker	Dr. Walker
	314	Engaging visuals used.	Mia Evans	Dr. Hernan...
	315	Encouraged critical thinking.	Noah Scott	Dr. King

5. Cross Join

Query to Generate all possible combinations of departments and students.

```
SELECT D.DeptName,S.StudentName
```

```
FROM Department D
```

```
CROSS JOIN Student S;
```

Output:

	DeptName	StudentName
▶	Environmental Science	Alice Johnson
	Geology	Alice Johnson
	Psychology	Alice Johnson
	Music	Alice Johnson
	Art	Alice Johnson
	Philosophy	Alice Johnson
	English Literature	Alice Johnson
	Political Science	Alice Johnson
	History	Alice Johnson
	Economics	Alice Johnson
	Data Science	Alice Johnson
	Biology	Alice Johnson
	Physics	Alice Johnson
	Mathematics	Alice Johnson
	Environmental Science	Bob Brown
	Geology	Bob Brown
	Psychology	Bob Brown
	Music	Bob Brown
	Art	Bob Brown
	Philosophy	Bob Brown
	English Literature	Bob Brown
	Political Science	Bob Brown
	History	Bob Brown
	Economics	Bob Brown

	Data Science	Bob Brown
	Biology	Bob Brown
	Physics	Bob Brown
	Mathematics	Bob Brown
	Environmental Science	Johnathan Doe
	Geology	Johnathan Doe
	Psychology	Johnathan Doe
	Music	Johnathan Doe
	Art	Johnathan Doe
	Philosophy	Johnathan Doe
	English Literature	Johnathan Doe
	Political Science	Johnathan Doe
	History	Johnathan Doe
	Economics	Johnathan Doe
	Data Science	Johnathan Doe
	Biology	Johnathan Doe
	Physics	Johnathan Doe
	Mathematics	Johnathan Doe
	Environmental Science	Diana Adams
	Geology	Diana Adams
	Psychology	Diana Adams
	Music	Diana Adams
	Art	Diana Adams

	Philosophy	Diana Adams
	English Literature	Diana Adams
	Political Science	Diana Adams
	History	Diana Adams
	Economics	Diana Adams
	Data Science	Diana Adams
	Biology	Diana Adams
	Physics	Diana Adams
	Mathematics	Diana Adams
	Environmental Science	Ethan Wright
	Geology	Ethan Wright
	Psychology	Ethan Wright
	Music	Ethan Wright
	Art	Ethan Wright
	Philosophy	Ethan Wright
	English Literature	Ethan Wright
	Political Science	Ethan Wright
	History	Ethan Wright
	Economics	Ethan Wright
	Data Science	Ethan Wright
	Biology	Ethan Wright
	Physics	Ethan Wright
	Mathematics	Ethan Wright

Environmental Science	Fiona Clark
Geology	Fiona Clark
Psychology	Fiona Clark
Music	Fiona Clark
Art	Fiona Clark
Philosophy	Fiona Clark
English Literature	Fiona Clark
Political Science	Fiona Clark
History	Fiona Clark
Economics	Fiona Clark
Data Science	Fiona Clark
Biology	Fiona Clark
Physics	Fiona Clark
Mathematics	Fiona Clark
Environmental Science	George Harris
Geology	George Harris
Psychology	George Harris
Music	George Harris
Art	George Harris
Philosophy	George Harris
English Literature	George Harris
Political Science	George Harris
History	George Harris

Economics	George Harris
Data Science	George Harris
Biology	George Harris
Physics	George Harris
Mathematics	George Harris
Environmental Science	Hannah Lewis
Geology	Hannah Lewis
Psychology	Hannah Lewis
Music	Hannah Lewis
Art	Hannah Lewis
Philosophy	Hannah Lewis
English Literature	Hannah Lewis
Political Science	Hannah Lewis
History	Hannah Lewis
Economics	Hannah Lewis
Data Science	Hannah Lewis
Biology	Hannah Lewis
Physics	Hannah Lewis
Mathematics	Hannah Lewis
Environmental Science	Irene Martinez
Geology	Irene Martinez
Psychology	Irene Martinez
Music	Irene Martinez

Art	Irene Martinez
Philosophy	Irene Martinez
English Literature	Irene Martinez
Political Science	Irene Martinez
History	Irene Martinez
Economics	Irene Martinez
Data Science	Irene Martinez
Biology	Irene Martinez
Physics	Irene Martinez
Mathematics	Irene Martinez
Environmental Science	Jack Moore
Geology	Jack Moore
Psychology	Jack Moore
Music	Jack Moore
Art	Jack Moore
Philosophy	Jack Moore
English Literature	Jack Moore
Political Science	Jack Moore
History	Jack Moore
Economics	Jack Moore
Data Science	Jack Moore
Biology	Jack Moore
Physics	Jack Moore

Mathematics	Jack Moore
Environmental Science	Kara Bell
Geology	Kara Bell
Psychology	Kara Bell
Music	Kara Bell
Art	Kara Bell
Philosophy	Kara Bell
English Literature	Kara Bell
Political Science	Kara Bell
History	Kara Bell
Economics	Kara Bell
Data Science	Kara Bell
Biology	Kara Bell
Physics	Kara Bell
Mathematics	Kara Bell
Environmental Science	Liam Parker
Geology	Liam Parker
Psychology	Liam Parker
Music	Liam Parker
Art	Liam Parker
Philosophy	Liam Parker
English Literature	Liam Parker
Political Science	Liam Parker

History	Liam Parker
Economics	Liam Parker
Data Science	Liam Parker
Biology	Liam Parker
Physics	Liam Parker
Mathematics	Liam Parker
Environmental Science	Mia Evans
Geology	Mia Evans
Psychology	Mia Evans
Music	Mia Evans
Art	Mia Evans
Philosophy	Mia Evans
English Literature	Mia Evans
Political Science	Mia Evans
History	Mia Evans
Economics	Mia Evans
Data Science	Mia Evans
Biology	Mia Evans
Physics	Mia Evans
Mathematics	Mia Evans
Environmental Science	Noah Scott
Geology	Noah Scott
Psychology	Noah Scott

Music	Noah Scott
Art	Noah Scott
Philosophy	Noah Scott
English Literature	Noah Scott
Political Science	Noah Scott
History	Noah Scott
Economics	Noah Scott
Data Science	Noah Scott
Biology	Noah Scott
Physics	Noah Scott
Mathematics	Noah Scott

6. Self Join

Query to Find all students who are enrolled in the same department.

```
SELECT S1.StudentName AS Student1,S2.StudentName AS Student2,
S1.DepartmentID
FROM Student S1
INNER JOIN Student S2
ON S1.DepartmentID = S2.DepartmentID
AND S1.StudentID <> S2.StudentID;
```

Output:

StudentName
Alice Johnson

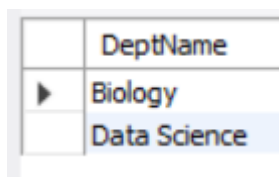
7. Self Join with Aggregation

Query to Find departments with more than one student enrolled.

```
SELECT S1.DepartmentID,COUNT(S1.StudentID) AS StudentCount
```

```
FROM Student S1  
  
INNER JOIN Student S2  
  
ON S1.DepartmentID = S2.DepartmentID  
  
GROUP BY S1.DepartmentID  
  
HAVING COUNT(S1.StudentID) > 1;
```

Output:



	DeptName
▶	Biology
	Data Science

8. Inner Join with Multiple Conditions

Query to Display all feedback along with the names of students and professors involved, considering feedback given by students only.

```
SELECT F.FeedbackID,F.FeedbackText,S.StudentName,P.ProfName  
  
FROM Feedback F  
  
INNER JOIN Student S  
  
ON F.StudentID = S.StudentID  
  
INNER JOIN Professor P  
  
ON F.ProfID = P.ProfID;
```

Output:

	FeedbackID	FeedbackText	StudentName	ProfName
▶	302	Helpful and clear explanations.	Alice Johnson	Dr. Taylor
	303	Engaging lecture style!	Bob Brown	Dr. Carter
	304	Simplified complex topics.	Johnathan Doe	Dr. Martinez
	305	Motivational and inspiring.	Diana Adams	Dr. Lee
	306	Innovative teaching approach.	Ethan Wright	Dr. Brown
	307	Clear and concise delivery.	Fiona Clark	Dr. Wilson
	308	Excellent knowledge.	George Harris	Dr. White
	309	Exceptional teaching style!	Hannah Lewis	Dr. Green
	310	Effective use of examples.	Irene Martinez	Dr. Hall
	311	Encouraged participation.	Jack Moore	Dr. Young
	312	Focused on practical applicati...	Kara Bell	Dr. Allen
	313	Detailed and thorough.	Liam Parker	Dr. Walker
	314	Engaging visuals used.	Mia Evans	Dr. Hernan...
	315	Encouraged critical thinking.	Noah Scott	Dr. King

9. Left Join with Conditional Filtering

Query to List all professors along with their departments, including those who are not assigned to any department.

```
SELECT P.ProfID,P.ProfName,D.DeptName
```

```
FROM Professor P
```

```
LEFT JOIN Department D
```

```
ON P.DepartmentID = D.DeptID;
```

Output:

	ProfID	ProfName	DeptName
▶	202	Dr. Taylor	Mathematics
	203	Dr. Carter	Physics
	204	Dr. Martinez	Biology
	205	Dr. Lee	Data Science
	206	Dr. Brown	Economics
	207	Dr. Wilson	History
	208	Dr. White	Political Science
	209	Dr. Green	English Literature
	210	Dr. Hall	Mathematics
	211	Dr. Young	Art
	212	Dr. Allen	Music
	213	Dr. Walker	Psychology
	214	Dr. Hernan...	Geology
	215	Dr. King	Environmental S...

10. Cross Join with Filtering

Query to Generate all possible combinations of students and professors who are in the same department.

```
SELECT S.StudentName,P.ProfName,S.DepartmentID
```

```
FROM Student S
```

```
CROSS JOIN Professor P
```

```
WHERE S.DepartmentID = P.DepartmentID;
```

Output:

	StudentName	ProfName	DepartmentID
▶	Alice Johnson	Dr. Taylor	2
	Alice Johnson	Dr. Hall	2
	Bob Brown	Dr. Carter	3
	Johnathan Doe	Dr. Martinez	4
	Diana Adams	Dr. Lee	5
	Ethan Wright	Dr. Brown	6
	Fiona Clark	Dr. Wilson	7
	George Harris	Dr. White	8
	Hannah Lewis	Dr. Green	9
	Jack Moore	Dr. Young	11
	Kara Bell	Dr. Allen	12
	Liam Parker	Dr. Walker	13
	Mia Evans	Dr. Hernan...	14
	Noah Scott	Dr. King	15

SUB QUERIES

1. Subquery to Find Students with No Feedback

Query:

-- to Find Students with No Feedback

```
SELECT StudentID, StudentName
```

```
FROM Student
```



```

WHERE StudentID NOT IN (

SELECT DISTINCT StudentID

FROM Feedback

);

```

Output:

	StudentID	StudentName
▶	116	Sophia Morgan
★	NULL	NULL

2. Subquery to Calculate Average Feedback Length

Query:

-- to Calculate Average Feedback Length

```

SELECT StudentID, AVG(LENGTH(FeedbackText)) AS
AverageFeedbackLength

FROM Feedback

GROUP BY StudentID;

```

Output:

	StudentID	AverageFeedbackLength
▶	102	31.0000
	103	23.0000
	104	26.0000
	105	27.0000
	106	29.0000
	107	27.0000
	108	20.0000
	109	27.0000
	110	26.0000
	111	25.0000
	112	34.0000
	113	22.0000
	114	22.0000
	115	29.0000

3. Subquery to Determine Busiest Professor

Query:

-- to Determine Busiest Professor

```
SELECT ProfID, COUNT(FeedbackID) AS FeedbackCount
```

```
FROM Feedback
```

```
GROUP BY ProfID
```

```
ORDER BY FeedbackCount DESC
```

```
LIMIT 1;
```

Output:

	ProfID	FeedbackCount
►	202	1

4. Subquery to Identify Departments with No Professors

Query:

-- to Identify Departments with No Professors

```
SELECT DeptID, DeptName
```

```
FROM Department
```

```
WHERE DeptID NOT IN (
```

```
SELECT DISTINCT DepartmentID
```

```
FROM Professor
```

```
);
```

Output:

	DeptID	DeptName
▶	10	Philosophy
•	NULL	NULL

5. Subquery to Find Feedback with the Longest Text

Query:

-- to Find Feedback with the Longest Text

```
SELECT FeedbackID, FeedbackText
```

```
FROM Feedback
```

```
WHERE LENGTH(FeedbackText) = (
```

```
SELECT MAX(LENGTH(FeedbackText))
```

```
FROM Feedback
```

```
);
```

Output:

	FeedbackID	FeedbackText
▶	312	Focused on practical applications.
•	NULL	NULL

6. Subquery to Determine Departments with the Most Students

Query:

-- to Determine Departments with the Most Students

```
SELECT DepartmentID, COUNT(StudentID) AS StudentCount
```

FROM Student

GROUP BY DepartmentID

ORDER BY StudentCount DESC

LIMIT 1;

Output:

	DepartmentID	StudentCount
▶	2	1

7. Subquery to Retrieve Feedback Given by Top Student

Query:

-- to Retrieve Feedback Given by Top Student

SELECT FeedbackID, FeedbackText

FROM Feedback

WHERE StudentID = (

SELECT StudentID

FROM Feedback

GROUP BY StudentID

ORDER BY AVG(LENGTH(FeedbackText)) DESC

LIMIT 1

);

Output:

	FeedbackID	FeedbackText
▶	312	Focused on practical applications.
•	NULL	NULL

8. Subquery to Find Professors Teaching Most Departments

Query:

-- to Find Professors Teaching Most Departments

```
SELECT ProfID, COUNT(DISTINCT DepartmentID) AS  
DepartmentCount  
FROM Professor  
GROUP BY ProfID  
ORDER BY DepartmentCount DESC  
LIMIT 1;
```

Output:

	ProfID	DepartmentCount
▶	202	1

9. Subquery to Retrieve Feedback with Approved Reports

Query:

-- to Retrieve Feedback with Approved Reports

```
SELECT FeedbackID, FeedbackText  
FROM Feedback
```

```
WHERE FeedbackID IN (  
  
SELECT FeedbackID  
  
FROM Report  
  
WHERE ReportText LIKE '%Approved%'  
  
);
```

Output:

	FeedbackText
▶	Motivational and inspiring.

10. Subquery to Find Students with Longest Feedback

Query:

-- to Find Students with Longest Feedback

```
SELECT StudentID, StudentName  
  
FROM Student  
  
WHERE StudentID IN (  
  
SELECT StudentID  
  
FROM Feedback  
  
WHERE LENGTH(FeedbackText) > (  
  
SELECT AVG(LENGTH(FeedbackText))  
  
FROM Feedback  
  
)
```

);

Output:

	StudentID	StudentName
▶	102	Alice Johnson
	105	Diana Adams
	106	Ethan Wright
	107	Fiona Clark
	109	Hannah Lewis
	112	Kara Bell
	115	Noah Scott
✱	NULL	NULL