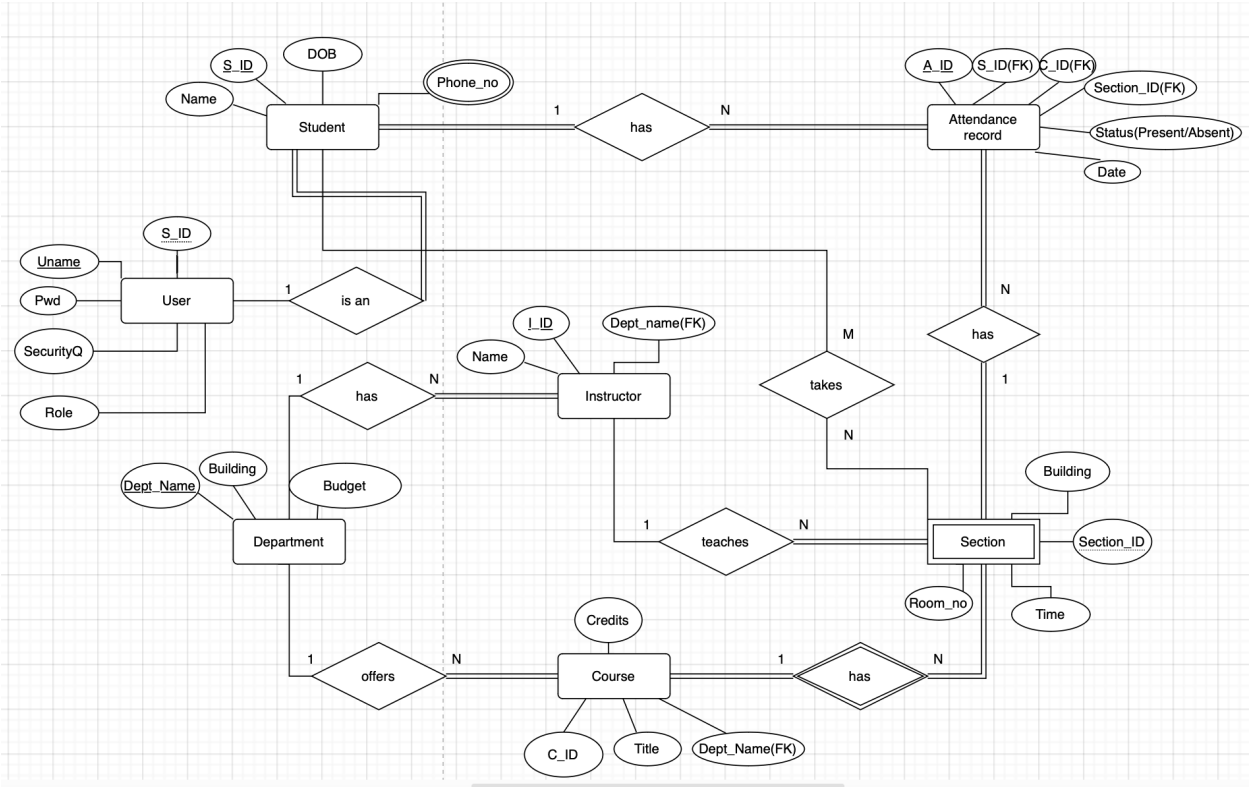
# Documentation for DBS project 2023

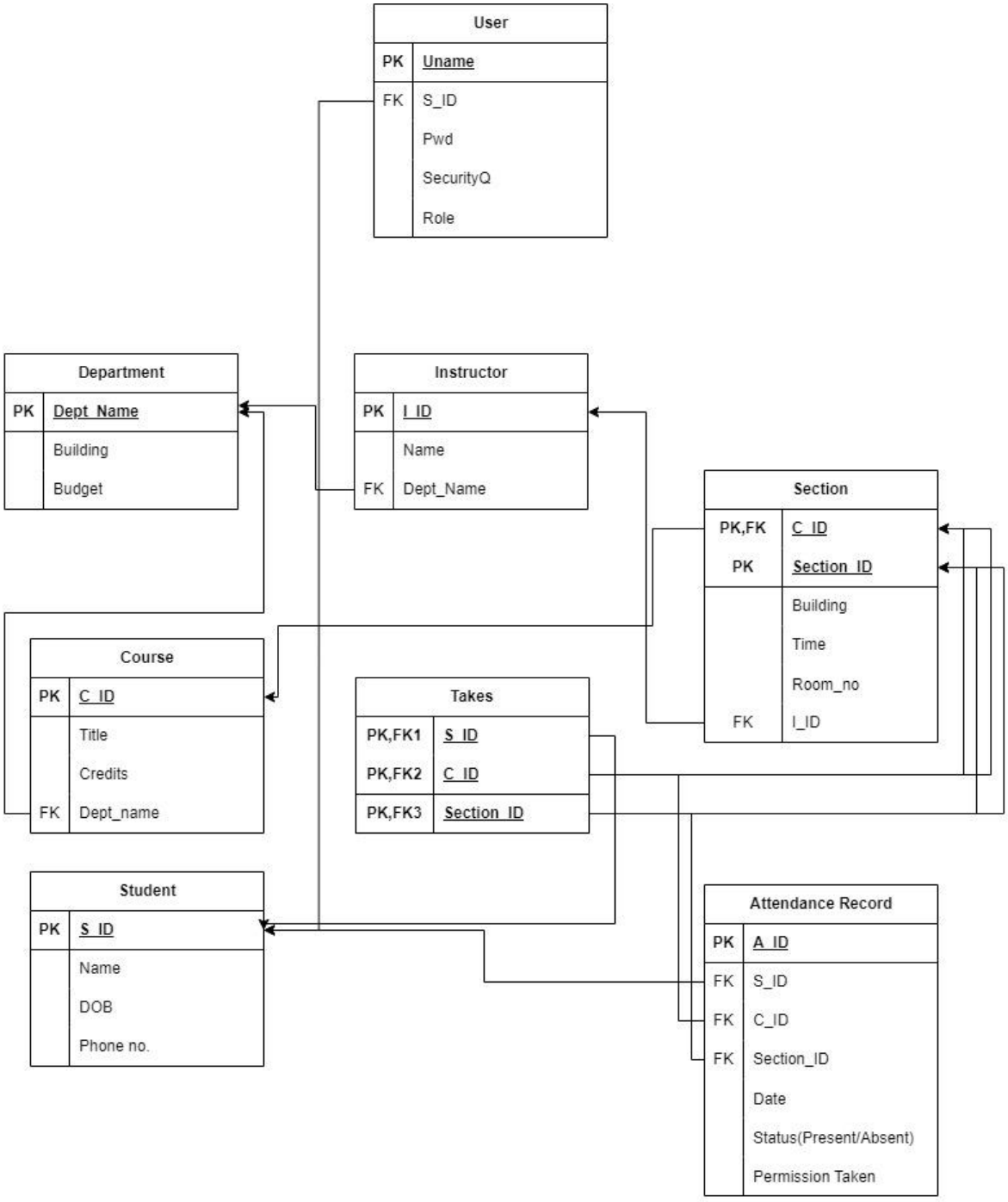
By- Apul Ranjan (2021A7PS2415P)
Peeyush Vatsa (2021A7PS0007P)

## Project 8 - Attendance Management System

# ER Diagram




# Conversion of ER Diagram to Relational Schema along with Normalisation




Normalisation: Student table violates 1NF as it has multi-valued attribute Phone number. To convert it to appropriate normal form, we will decompose the student table as below:

Student 1

| S_ID | Name | DOB |
|---|---|---|

Student 2

| S_ID | Phone no. |
|---|---|

With this normalisation, all the tables are now in 3NF.

Regarding normalisation: The only table which violated 3NF was Student which was broken into two tables Student 1 and Student 2.

# SQL Queries

```
-- Query 1) Insert records for administrators
INSERT INTO User (Uname, S_ID, Pwd, Role)
   VALUES ('admin1', NULL, SHA2('password123', 256), 'admin');
INSERT INTO User (Uname, S_ID, Pwd, Role)
   VALUES ('admin2', NULL, SHA2('adminpassword', 256), 'admin');
```

```
mysql> INSERT INTO User (Uname, S_ID, Pwd, Role)
    ->     VALUES ('admin1', NULL, SHA2('password123', 256), 'admin');
Query OK, 1 row affected (0.00 sec)

mysql> INSERT INTO User (Uname, S_ID, Pwd, Role)
    ->     VALUES ('admin2', NULL, SHA2('adminpassword', 256), 'admin');
Query OK, 1 row affected (0.00 sec)

mysql> select * from user;
+-------------+------+------------------------------------------------------------------+-------+
| Uname       | S_ID | Pwd                                                              | Role  |
+-------------+------+------------------------------------------------------------------+-------+
| admin1      | NULL | ef92b778bafe771e89245b89ecbc08a44a4e166c06659911881f383d4473e94f | admin |
| admin2      | NULL | 749f09bade8aca755660eeb17792da880218d4fbdc4e25fbec279d7fe9f65d70 | admin |
| alice       |    1 | 5e884898da28047151d0e56f8dc6292773603d0d6aabbdd62a11ef721d1542d8 | user  |
| awesomeuser |    8 | 27cc6994fc1c01ce6659c6bddca9b69c4c6a9418065e612c69d110b3f7b11f8a | user  |
| bob         |    2 | 2bb80d537b1da3e38bd30361aa855686bde0eacd7162fef6a25fe97bf527a25b | user  |
| charlie     |    3 | 8d969eef6ecad3c29a3a629280e686cf0c3f5d5a86aff3ca12020c923adc6c92 | user  |
| cooluser    |    7 | 0b14d501a594442a01c6859541bcb3e8164d183d32937b851835442f69d5c94e | user  |
| david       |    4 | 65e84be33532fb784c48129675f9eff3a682b27168c0ea744b2cf58ee02337c5 | user  |
| eliteuser   |    9 | 0fd205965ce169b5c023282bb5fa2e239b6716726db5defaa8ceff225be805dc | user  |
| fancyuser   |    5 | 89e01536ac207279409d4de1e5253e01f4a1769e696db0d6062ca9b8f56767c8 | user  |
| superuser   |   10 | ef92b778bafe771e89245b89ecbc08a44a4e166c06659911881f383d4473e94f | user  |
| uniqueuser  |    6 | 1c8bfe8f801d79745c4631d09fff36c82aa37fc4cce4fc946683d7b336b63032 | user  |
+-------------+------+------------------------------------------------------------------+-------+
12 rows in set (0.00 sec)

mysql>
```

```
-- Query 2) Admin creating profiles for students and instructors
-- Create profile for a student
INSERT INTO Student (S_ID, Name, DOB)
   VALUES (1001, 'John Doe', '2000-05-01');
 -- Create profile for an instructor
INSERT INTO Instructor (I_ID, Name, Dept_Name)
   VALUES (2001, 'Jane Smith', 'Computer Science');
```

```
mysql> INSERT INTO Student (S_ID, Name, DOB)
    ->     VALUES (1001, 'John Doe', '2000-05-01');
Query OK, 1 row affected (0.01 sec)

mysql> select * from student;
+------+------------------+------------+
| S_ID | Name             | DOB        |
+------+------------------+------------+
|    1 | Ava Johnson      | 2002-06-03 |
|    2 | Ethan Lee        | 2003-08-12 |
|    3 | Olivia Chen      | 2005-10-21 |
|    4 | Liam Kim         | 2004-07-07 |
|    5 | Emily Brown      | 2001-01-15 |
|    6 | Noah Patel       | 2002-12-29 |
|    7 | Chloe Davis      | 2006-04-10 |
|    8 | William Hernandez | 2007-09-05 |
|    9 | Sophia Singh     | 2004-05-26 |
|   10 | Michael Jones    | 2005-11-11 |
| 1001 | John Doe         | 2000-05-01 |
+------+------------------+------------+
11 rows in set (0.00 sec)

mysql>
```

```
mysql> INSERT INTO Instructor (I_ID, Name, Dept_Name)
    ->     VALUES (2001, 'Jane Smith', 'Computer Science');
Query OK, 1 row affected (0.00 sec)

mysql> select * from instructor;
+------+------------------+------------------+
| I_ID | Name             | Dept_Name        |
+------+------------------+------------------+
|    1 | John Doe         | Mathematics      |
|    2 | Jane Smith       | Biology          |
|    3 | Bob Johnson      | Computer Science |
|    4 | Tom Williams     | Psychology       |
|    5 | Emily Davis      | English          |
|    6 | Michael Lee      | Physics          |
|    7 | Sarah Kim        | Chemistry        |
|    8 | David Chen       | Economics        |
|    9 | Jessica Brown    | History          |
|   10 | Brian Wilson     | Sociology        |
|   11 | Karen Taylor     | Mathematics      |
|   12 | Jason Anderson   | Biology          |
|   13 | Rachel Kim       | Computer Science |
|   14 | Samuel Lee       | Psychology       |
|   15 | Laura Davis      | English          |
|   16 | Daniel Park      | Physics          |
|   17 | Kevin Brown      | Economics        |
|   18 | Eric Wilson      | History          |
|   19 | Catherine Taylor | Sociology        |
|   20 | Amanda Johnson   | Mathematics      |
|   21 | James Anderson   | Biology          |
|   22 | Alex Kim         | Computer Science |
|   23 | Mary Lee         | Psychology       |
|   24 | Olivia Davis     | English          |
|   25 | Andrew Park      | Physics          |
|   26 | Stephanie Brown  | Economics        |
|   27 | Jacob Wilson     | History          |
|   28 | Natalie Taylor   | Sociology        |
| 2001 | Jane Smith       | Computer Science |
+------+------------------+------------------+
29 rows in set (0.00 sec)
```

```sql
-- Query 3) Admin updating details of students and instructors
-- Update phone number of a student
UPDATE Student_Phone
SET Phone_no = '987-654-3211'
WHERE S_ID = 2;
-- Update ID of an instructor
UPDATE Instructor
SET Name = 'Jane Doe'
WHERE I_ID = 2;
```

```
mysql> UPDATE Student_Phone
    -> SET Phone_no = '987-654-3211'
    -> WHERE S_ID = 2;
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> select * from Student_phone;
+------+--------------+
| S_ID | Phone_no     |
+------+--------------+
|    1 | 123-456-7890 |
|    1 | 555-555-5555 |
|    2 | 987-654-3211 |
|    3 | 111-222-3333 |
|    4 | 444-555-6666 |
|    4 | 777-888-9999 |
|    5 | 123-123-1234 |
|    6 | 555-1212     |
|    6 | 555-1313     |
|    7 | 555-1414     |
|    8 | 555-1515     |
|    9 | 555-1616     |
|   10 | 555-1717     |
|   10 | 555-1818     |
+------+--------------+
14 rows in set (0.00 sec)

mysql>
```

```
mysql> UPDATE Instructor
    -> SET Name = 'Jane Doe'
    -> WHERE I_ID = 2;
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> select * from instructor;
+------+------------------+------------------+
| I_ID | Name             | Dept_Name        |
+------+------------------+------------------+
|    1 | John Doe         | Mathematics      |
|    2 | Jane Doe         | Biology          |
|    3 | Bob Johnson      | Computer Science |
|    4 | Tom Williams     | Psychology       |
|    5 | Emily Davis      | English          |
|    6 | Michael Lee      | Physics          |
|    7 | Sarah Kim        | Chemistry        |
|    8 | David Chen       | Economics        |
|    9 | Jessica Brown    | History          |
|   10 | Brian Wilson     | Sociology        |
|   11 | Karen Taylor     | Mathematics      |
|   12 | Jason Anderson   | Biology          |
|   13 | Rachel Kim       | Computer Science |
|   14 | Samuel Lee       | Psychology       |
|   15 | Laura Davis      | English          |
|   16 | Daniel Park      | Physics          |
|   17 | Kevin Brown      | Economics        |
|   18 | Eric Wilson      | History          |
|   19 | Catherine Taylor | Sociology        |
|   20 | Amanda Johnson   | Mathematics      |
|   21 | James Anderson   | Biology          |
|   22 | Alex Kim         | Computer Science |
|   23 | Mary Lee         | Psychology       |
|   24 | Olivia Davis     | English          |
|   25 | Andrew Park      | Physics          |
|   26 | Stephanie Brown  | Economics        |
|   27 | Jacob Wilson     | History          |
|   28 | Natalie Taylor   | Sociology        |
| 2001 | Jane Smith       | Computer Science |
+------+------------------+------------------+
29 rows in set (0.00 sec)

mysql>
```

```
-- Query 4) Signin authentication to verify if provided credentials by students are correct
SELECT COUNT(*) FROM User
WHERE Uname = 'alice' AND Pwd = SHA2('password', 256);
 -- returns 1 if valid else return 0
```

```
mysql> SELECT COUNT(*) FROM User
    -> WHERE Uname = 'alice' AND Pwd = SHA2('password', 256);
+----------+
| COUNT(*) |
+----------+
|        1 |
+----------+
1 row in set (0.00 sec)

mysql> SELECT COUNT(*) FROM User
    -> WHERE Uname = 'alice' AND Pwd = SHA2('password123', 256);
+----------+
| COUNT(*) |
+----------+
|        0 |
+----------+
1 row in set (0.00 sec)

mysql>
```

```
-- Query 5) Password reset to update the stored password against username
-- Verify security question answer before updating password
SELECT COUNT(*) FROM Student_Phone JOIN User USING(S_ID) WHERE Phone_no =
'123-456-7890' AND Uname = 'alice'; -- here phone number is entered by user.
-- The query above returns 0 if security q has failed
-- Update password for a user
UPDATE User SET Pwd = SHA2('newpassword', 256) WHERE Uname = 'alice';
```

```
mysql> SELECT COUNT(*) FROM Student_Phone JOIN User USING(S_ID) WHERE Phone_no = '123-456-7890' AND Uname = 'alice'; -- here phone number is en
tered by user.
+----------+
| COUNT(*) |
+----------+
|        1 |
+----------+
1 row in set (0.00 sec)

mysql> UPDATE User SET Pwd = SHA2('newpassword', 256) WHERE Uname = 'alice';
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql>
```

```
-- Query 6) User should be able to manage the profile
-- Update phone number of a student
UPDATE Student_Phone SET Phone_no = '111-222-3344' WHERE S_ID = 3;
-- Update name of a student
UPDATE Student SET Name = 'Jane Doe' WHERE S_ID = 1001;
```

```
mysql> UPDATE Student_Phone SET Phone_no = '111-222-3344' WHERE S_ID = 3;
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> UPDATE Student SET Name = 'Jane Doe' WHERE S_ID = 1001;
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql>
```

```
-- Query 7) Query to search for students based on simple filters like attendance, student name
-- Search for students with attendance status of 'present'
SELECT * FROM Student WHERE S_ID IN ( SELECT S_ID FROM Attendance_Record WHERE Status = 'present' );
-- Search for students with name containing ‘Emily’
SELECT * FROM Student WHERE Name LIKE '%Emily%';
```

```
mysql> UPDATE Student_Phone SET Phone_no = '111-222-3344' WHERE S_ID = 3;
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> UPDATE Student SET Name = 'Jane Doe' WHERE S_ID = 1001;
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql>
```

```
-- Query 8) Query to search for students based on filters such as
-- average percentage of classes attended, requests for leave permission, absent without permission
-- Search for students with attendance rate less than 80%
SELECT * FROM Student WHERE S_ID IN ( SELECT S_ID FROM Attendance_Record GROUP BY S_ID HAVING AVG(CASE WHEN Status = 'present' THEN 1 ELSE 0 END) < 0.8 );
-- Search for students who have requested for leave permission
SELECT * FROM Student WHERE S_ID IN ( SELECT S_ID FROM Attendance_Record WHERE Permission_taken = TRUE );
-- Search for students who have been absent without permission
```

```
SELECT * FROM Student WHERE S_ID IN ( SELECT S_ID FROM Attendance_Record
WHERE Permission_taken = FALSE AND Status = 'absent' );
```

```
mysql> SELECT * FROM Student WHERE S_ID IN ( SELECT S_ID FROM Attendance_Record WHERE Status = 'present' );
+------+-------------+------------+
| S_ID | Name        | DOB        |
+------+-------------+------------+
|    1 | Ava Johnson | 2002-06-03 |
|    2 | Ethan Lee   | 2003-08-12 |
+------+-------------+------------+
2 rows in set (0.13 sec)

mysql> SELECT * FROM Student WHERE Name LIKE '%Emily%';
+------+-------------+------------+
| S_ID | Name        | DOB        |
+------+-------------+------------+
|    5 | Emily Brown | 2001-01-15 |
+------+-------------+------------+
1 row in set (0.00 sec)

mysql>
```

```
-- Query 9) Teachers can mark the attendance for students every day
-- Format:
--     INSERT INTO Attendance_Record (S_ID, C_ID, Section_ID, Date, Status,
Permission_taken)
--     VALUES (student_id, course_id, section_id, current_date, 'present', false);
INSERT INTO Attendance_Record (S_ID, C_ID, Section_ID, Date, Status,
Permission_taken) VALUES (1, 1, 1, current_date, 'present', NULL);
INSERT INTO Attendance_Record (S_ID, C_ID, Section_ID, Date, Status,
Permission_taken) VALUES (2, 3, 2, current_date, 'present', NULL);
INSERT INTO Attendance_Record (S_ID, C_ID, Section_ID, Date, Status,
Permission_taken) VALUES (3, 2, 3, current_date, 'absent', TRUE);
-- Replace student_id, course_id, and section_id with the corresponding IDs of the
-- student, course, and section for which attendance is being marked.
```

```
mysql> INSERT INTO Attendance_Record (S_ID, C_ID, Section_ID, Date, Status, Permission_taken) VALUES (1, 1, 1, current_date, 'present', NULL);
Query OK, 1 row affected (0.00 sec)

mysql> INSERT INTO Attendance_Record (S_ID, C_ID, Section_ID, Date, Status, Permission_taken) VALUES (2, 3, 2, current_date, 'present', NULL);
Query OK, 1 row affected (0.00 sec)

mysql> INSERT INTO Attendance_Record (S_ID, C_ID, Section_ID, Date, Status, Permission_taken) VALUES (3, 2, 3, current_date, 'absent', TRUE);
Query OK, 1 row affected (0.00 sec)

mysql> select * from attendance_record;
+------+------+------+------------+------------+---------+------------------+
| A_ID | S_ID | C_ID | Section_ID | Date       | Status  | Permission_taken |
+------+------+------+------------+------------+---------+------------------+
|    1 |    1 |    1 |          1 | 2023-04-11 | present |             NULL |
|    2 |    2 |    3 |          2 | 2023-04-11 | present |             NULL |
|    3 |    3 |    2 |          3 | 2023-04-11 | absent  |                1 |
+------+------+------+------------+------------+---------+------------------+
3 rows in set (0.00 sec)

mysql>
```

```
-- Query 10) Teachers can view attendance report for the entire class
-- Format:
-- SELECT s.Name, a.Date, a.Status
-- FROM Student s JOIN Takes t ON s.S_ID = t.S_ID JOIN Attendance_Record a ON t.C_ID
= a.C_ID AND t.Section_ID = a.Section_ID
```

```
-- WHERE t.C_ID = course_id AND t.Section_ID = section_id;
-- Replace course_id and section_id with the corresponding IDs
SELECT s.Name, a.Date, a.Status
FROM Student s JOIN Takes t ON s.S_ID = t.S_ID JOIN Attendance_Record a ON t.C_ID =
a.C_ID AND t.Section_ID = a.Section_ID
WHERE t.C_ID = 1 AND t.Section_ID = 1;
SELECT s.Name, a.Date, a.Status
FROM Student s JOIN Takes t ON s.S_ID = t.S_ID JOIN Attendance_Record a ON t.C_ID =
a.C_ID AND t.Section_ID = a.Section_ID
WHERE t.C_ID = 3 AND t.Section_ID = 1;
```

```
mysql> SELECT s.Name, a.Date, a.Status
    -> FROM Student s JOIN Takes t ON s.S_ID = t.S_ID JOIN Attendance_Record a ON t.C_ID = a.C_ID AND t.Section_ID = a.Section_ID
    -> WHERE t.C_ID = 1 AND t.Section_ID = 1;
+-------------+------------+---------+
| Name        | Date       | Status  |
+-------------+------------+---------+
| Ava Johnson | 2023-04-11 | present |
+-------------+------------+---------+
1 row in set (0.00 sec)

mysql>
mysql> SELECT s.Name, a.Date, a.Status
    -> FROM Student s JOIN Takes t ON s.S_ID = t.S_ID JOIN Attendance_Record a ON t.C_ID = a.C_ID AND t.Section_ID = a.Section_ID
    -> WHERE t.C_ID = 3 AND t.Section_ID = 1;
Empty set (0.00 sec)

mysql>
```

```
-- Query 11) Attendance patterns for the class can be viewed by the teacher
-- Format:
-- SELECT a.Date, COUNT(CASE WHEN a.Status = 'present' THEN 1 END) AS Present,
-- COUNT(CASE WHEN a.Status = 'absent' THEN 1 END) AS Absent
-- FROM Takes t JOIN Attendance_Record a ON t.C_ID = a.C_ID AND t.Section_ID =
a.Section_ID
-- WHERE t.C_ID = course_id AND t.Section_ID = section_id GROUP BY a.Date;
-- Replace course_id and section_id with the corresponding IDs
SELECT a.Date, COUNT(CASE WHEN a.Status = 'present' THEN 1 END) AS Present,
COUNT(CASE WHEN a.Status = 'absent' THEN 1 END) AS Absent
FROM Takes t JOIN Attendance_Record a ON t.C_ID = a.C_ID AND t.Section_ID =
a.Section_ID
WHERE t.C_ID = 1 AND t.Section_ID = 1 GROUP BY a.Date;
```

```
mysql> SELECT a.Date, COUNT(CASE WHEN a.Status = 'present' THEN 1 END) AS Present,
    -> COUNT(CASE WHEN a.Status = 'absent' THEN 1 END) AS Absent
    -> FROM Takes t JOIN Attendance_Record a ON t.C_ID = a.C_ID AND t.Section_ID = a.Section_ID
    -> WHERE t.C_ID = 1 AND t.Section_ID = 1 GROUP BY a.Date;
+------------+---------+--------+
| Date       | Present | Absent |
+------------+---------+--------+
| 2023-04-11 |       1 |      0 |
+------------+---------+--------+
1 row in set (0.00 sec)

mysql>
```

```
-- Query 12) Students should be able to view their weekly attendance report
```

```
-- Format:
-- SELECT a.Date, a.Status FROM Attendance_Record a JOIN Takes t ON a.C_ID = t.C_ID AND a.Section_ID = t.Section_ID
-- WHERE t.S_ID = student_id AND a.Date BETWEEN current_date - INTERVAL 7 DAY AND current_date;
-- Replace student_id with its corresponding id
SELECT a.Date, a.Status FROM Attendance_Record a JOIN Takes t ON a.C_ID = t.C_ID AND a.Section_ID = t.Section_ID
WHERE t.S_ID = 1 AND a.Date BETWEEN current_date - INTERVAL 7 DAY AND current_date;
```

```
mysql> SELECT a.Date, a.Status FROM Attendance_Record a JOIN Takes t ON a.C_ID = t.C_ID AND a.Section_ID = t.Section_ID
    -> WHERE t.S_ID = 1 AND a.Date BETWEEN current_date - INTERVAL 7 DAY AND current_date;
+------------+---------+
| Date       | Status  |
+------------+---------+
| 2023-04-11 | present |
+------------+---------+
1 row in set (0.00 sec)

mysql>
```

```
-- Query 13) Admin can generate analysis reports on key metrics such as attendance rates, attendance patterns for students
SELECT Student.S_ID, Student.Name, COUNT(Attendance_Record.Status) AS Total_Classes,
SUM(CASE WHEN Attendance_Record.Status = 'present' THEN 1 ELSE 0 END) AS Present,
SUM(CASE WHEN Attendance_Record.Status = 'absent' THEN 1 ELSE 0 END) AS Absent,
(SUM(CASE WHEN Attendance_Record.Status = 'present' THEN 1 ELSE 0 END) / COUNT(Attendance_Record.Status)) * 100 AS Attendance_Rate
FROM Student
LEFT JOIN Attendance_Record ON Student.S_ID = Attendance_Record.S_ID
GROUP BY Student.S_ID, Student.Name;
```

```
mysql> SELECT Student.S_ID, Student.Name, COUNT(Attendance_Record.Status) AS Total_Classes,
    -> SUM(CASE WHEN Attendance_Record.Status = 'present' THEN 1 ELSE 0 END) AS Present,
    -> SUM(CASE WHEN Attendance_Record.Status = 'absent' THEN 1 ELSE 0 END) AS Absent,
    -> (SUM(CASE WHEN Attendance_Record.Status = 'present' THEN 1 ELSE 0 END) / COUNT(Attendance_Record.Status)) * 100 AS Attendance_Rate
    -> FROM Student
    -> LEFT JOIN Attendance_Record ON Student.S_ID = Attendance_Record.S_ID
    -> GROUP BY Student.S_ID, Student.Name;
+------+------------------+---------------+---------+--------+-----------------+
| S_ID | Name             | Total_Classes | Present | Absent | Attendance_Rate |
+------+------------------+---------------+---------+--------+-----------------+
|    1 | Ava Johnson      |             1 |       1 |      0 |        100.0000 |
|    2 | Ethan Lee        |             1 |       1 |      0 |        100.0000 |
|    3 | Olivia Chen      |             1 |       0 |      1 |          0.0000 |
|    4 | Liam Kim         |             0 |       0 |      0 |            NULL |
|    5 | Emily Brown      |             0 |       0 |      0 |            NULL |
|    6 | Noah Patel       |             0 |       0 |      0 |            NULL |
|    7 | Chloe Davis      |             0 |       0 |      0 |            NULL |
|    8 | William Hernandez |             0 |       0 |      0 |            NULL |
|    9 | Sophia Singh     |             0 |       0 |      0 |            NULL |
|   10 | Michael Jones    |             0 |       0 |      0 |            NULL |
| 1001 | Jane Doe         |             0 |       0 |      0 |            NULL |
+------+------------------+---------------+---------+--------+-----------------+
11 rows in set, 8 warnings (0.00 sec)

mysql>
```

# Link to GDrive Videos:

Apul Ranjan (2021A7PS2415P):
https://drive.google.com/drive/folders/1OWRA_CiA-VAjiZsCGLt0hQQ9pjB_8L7A?usp=sharing

Peeyush Vatsa(2021A7PS0007P)
https://drive.google.com/drive/folders/1OY8EQ9a02jh4mo46gQWce4f7SktS7x_0?usp=sharing