Department of Information and Communication Technology

Faculty of Technology

University of Ruhuna

Database Management Systems Practicum

ICT 1222

Assignment 02 – Mini Project Group

Group 03

Submitted to: Mr.P.H.P. Nuwan Laksiri

Submitted by: TG/2022/1393 – B.S.M.N.Hansini

TG/2022/1372 – K.S.Kaushalya

TG/2022/1378 – N.A.M.N.Arachchi

TG/2022/1400 – J.Dhanushiya

Contents

Brief introduction about the problem/group project

Regarding the database creating task , we discuss the issues and potential enhancements should perform in this system.. The goal of this project is to evaluate the database's present efficiency and effectiveness. The database is an essential tool for handling student data, academic records, and administrative procedures.

• **Easy Data Entry**: It reduces the need for manual data entry and helps keep information accurate.

• **Prevents Duplicates**: It helps avoid duplicate records, so data remains clean and organized.

• **Fast Data Access**: Information can be quickly retrieved when needed, saving time..

• **Good Security**: TECMIS has security measures, like access control, to keep data safe.

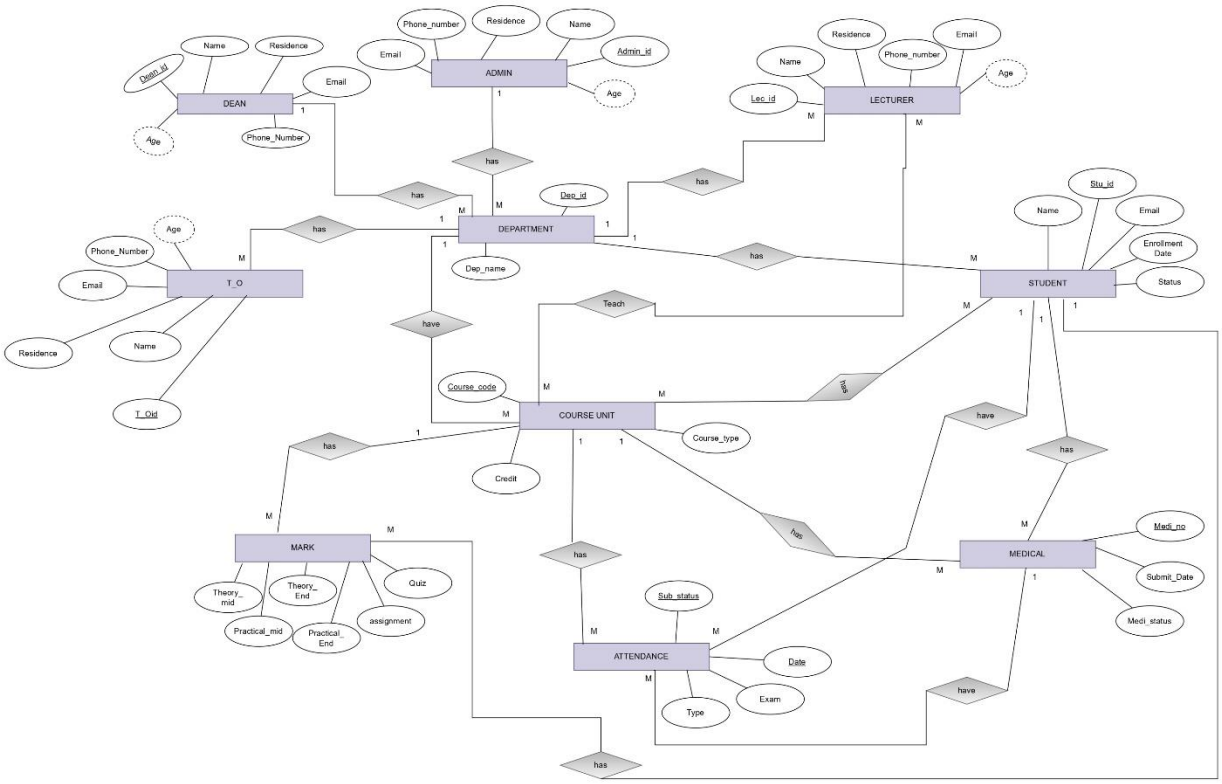
Brief introduction to the solution

To address the needs of the system, we propose a **Student Management System (SMS)** that will streamline academic and administrative tasks. This system will be a comprehensive platform designed to manage student data, academic records, attendance, and grades while providing secure access to faculty, students, and administrators.

- **User Management**: Secure, role-based logins for administrators, faculty, and students, ensuring data access is properly managed.
- **Course Management**: Tools for managing course schedules and easily adding, editing, or removing courses.
- **Enrollment and Attendance Tracking**: Real-time tracking of student enrollments and attendance.
- **Grade Management**: An efficient grading system for entering and reporting grades, with analytics to identify trends and improve academic performance.

- **Integration with Existing Systems**: Compatibility with other university systems, such as the Student Information System (SIS) and Learning Management System (LMS).

Proposed ER/EER diagram



Proposed Relational mapping diagram

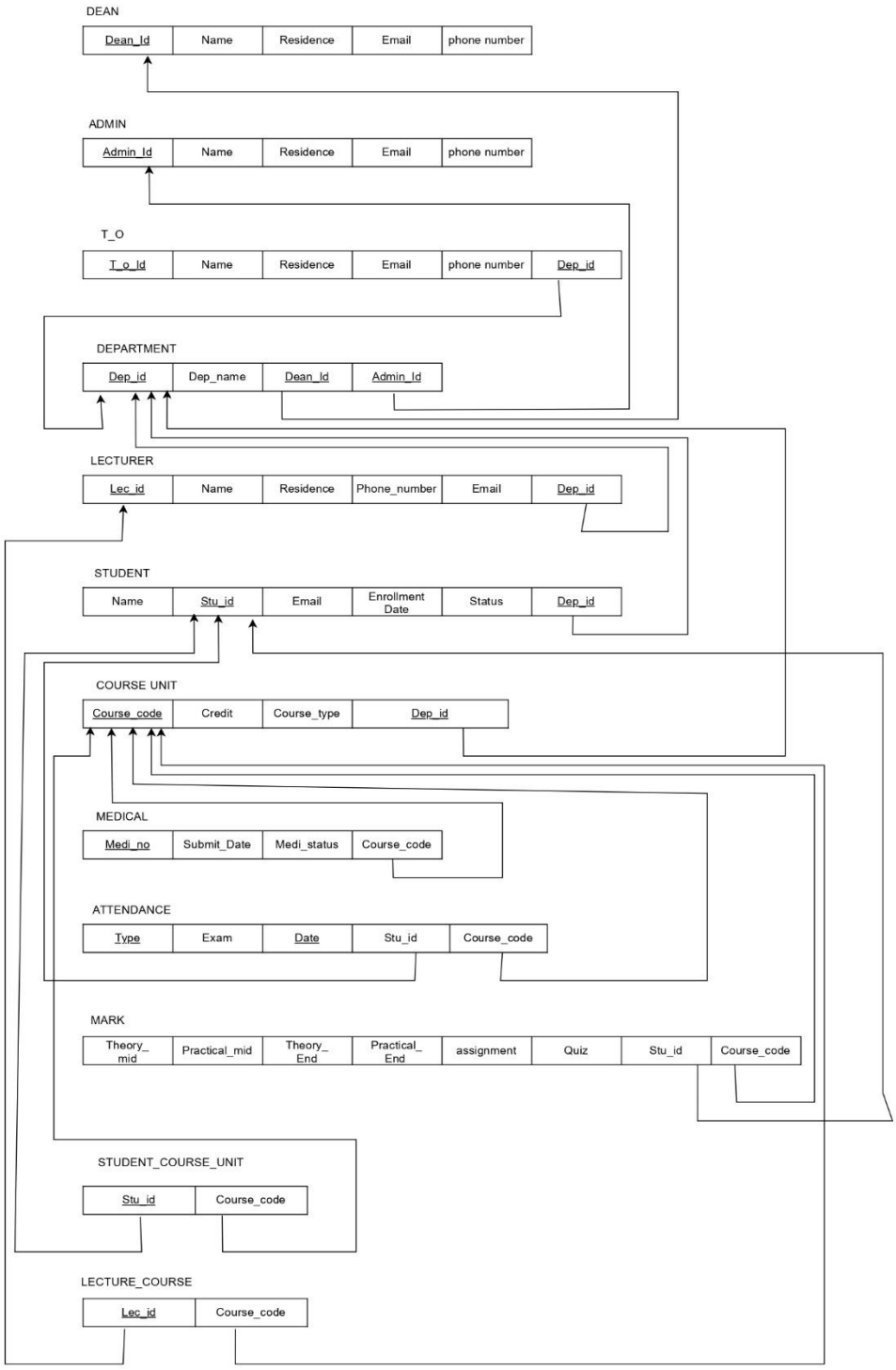


Table structures of solution

Admin Table

```
mysql> desc admin;
+-----------+-------------+------+-----+---------+-------+
| Field     | Type        | Null | Key | Default | Extra |
+-----------+-------------+------+-----+---------+-------+
| Admin_id  | char(10)    | NO   | PRI | NULL    |       |
| Name      | varchar(30) | YES  |     | NULL    |       |
| Email     | varchar(50) | YES  |     | NULL    |       |
| Phone_no  | varchar(11) | YES  |     | NULL    |       |
| Residence | varchar(20) | YES  |     | NULL    |       |
+-----------+-------------+------+-----+---------+-------+
5 rows in set (0.07 sec)
```

Dean table

```
mysql> desc dean;
+-----------+-------------+------+-----+---------+-------+
| Field     | Type        | Null | Key | Default | Extra |
+-----------+-------------+------+-----+---------+-------+
| Dean_id   | char(10)    | NO   | PRI | NULL    |       |
| Name      | varchar(30) | YES  |     | NULL    |       |
| Email     | varchar(50) | YES  |     | NULL    |       |
| Residence | varchar(20) | YES  |     | NULL    |       |
| Phone_no  | varchar(11) | YES  |     | NULL    |       |
+-----------+-------------+------+-----+---------+-------+
5 rows in set (0.00 sec)
```

Technical officer table

```
mysql> desc t_o;
+-----------+-------------+------+-----+---------+-------+
| Field     | Type        | Null | Key | Default | Extra |
+-----------+-------------+------+-----+---------+-------+
| Tec_id    | char(10)    | NO   | PRI | NULL    |       |
| Dep_id    | char(6)     | NO   | PRI | NULL    |       |
| Name      | varchar(30) | YES  |     | NULL    |       |
| Phone_no  | varchar(11) | YES  |     | NULL    |       |
| Residence | varchar(20) | YES  |     | NULL    |       |
+-----------+-------------+------+-----+---------+-------+
5 rows in set (0.00 sec)
```

Department table

```
mysql> desc department;
+----------+-------------+------+-----+---------+-------+
| Field    | Type        | Null | Key | Default | Extra |
+----------+-------------+------+-----+---------+-------+
| Dep_id   | char(6)     | NO   | PRI | NULL    |       |
| Dep_name | varchar(50) | YES  |     | NULL    |       |
| Admin_id | char(10)    | YES  |     | NULL    |       |
| Dean_id  | char(10)    | YES  |     | NULL    |       |
+----------+-------------+------+-----+---------+-------+
4 rows in set (0.01 sec)
```

Course_unit Table

```
mysql> desc course_unit;
+-------------+-------------+------+-----+---------+-------+
| Field       | Type        | Null | Key | Default | Extra |
+-------------+-------------+------+-----+---------+-------+
| Course_code | char(10)    | NO   | PRI | NULL    |       |
| Course_name | varchar(40) | YES  |     | NULL    |       |
| Dep_id      | char(6)     | YES  | MUL | NULL    |       |
| Type        | varchar(20) | YES  |     | NULL    |       |
| Credit      | int         | YES  |     | NULL    |       |
+-------------+-------------+------+-----+---------+-------+
5 rows in set (0.11 sec)
```

Lecturer Table

```
mysql> desc lecturer;
+-----------+-------------+------+-----+---------+-------+
| Field     | Type        | Null | Key | Default | Extra |
+-----------+-------------+------+-----+---------+-------+
| Lec_id    | char(10)    | NO   | PRI | NULL    |       |
| Name      | varchar(30) | YES  |     | NULL    |       |
| Email     | varchar(50) | YES  |     | NULL    |       |
| Dep_id    | char(6)     | YES  | MUL | NULL    |       |
| Phone_no  | varchar(11) | YES  |     | NULL    |       |
| Residence | varchar(20) | YES  |     | NULL    |       |
+-----------+-------------+------+-----+---------+-------+
6 rows in set (0.00 sec)
```

Students table

```
mysql> desc student;
+-----------------+-------------+------+-----+---------+-------+
| Field           | Type        | Null | Key | Default | Extra |
+-----------------+-------------+------+-----+---------+-------+
| Stu_id          | char(10)    | NO   | PRI | NULL    |       |
| Name            | varchar(30) | YES  |     | NULL    |       |
| Email           | varchar(50) | YES  |     | NULL    |       |
| Dep_id          | char(6)     | YES  | MUL | NULL    |       |
| Enrollment_date | date        | YES  |     | NULL    |       |
| Status          | varchar(20) | YES  |     | NULL    |       |
+-----------------+-------------+------+-----+---------+-------+
6 rows in set (0.04 sec)
```

Medical table

```
mysql> desc medical;
+-------------+-------------+------+-----+---------+-------+
| Field       | Type        | Null | Key | Default | Extra |
+-------------+-------------+------+-----+---------+-------+
| Medi_no     | char(6)     | NO   | PRI | NULL    |       |
| Course_code | char(10)    | YES  | MUL | NULL    |       |
| Stu_id      | char(10)    | YES  | MUL | NULL    |       |
| Submit_date | date        | YES  |     | NULL    |       |
| M_status    | varchar(20) | YES  |     | NULL    |       |
+-------------+-------------+------+-----+---------+-------+
5 rows in set (0.52 sec)
```

Lecturer_course table

```
mysql> desc lecturer_course;
+-------------+----------+------+-----+---------+-------+
| Field       | Type     | Null | Key | Default | Extra |
+-------------+----------+------+-----+---------+-------+
| Lec_id      | char(10) | NO   | PRI | NULL    |       |
| Course_code | char(10) | NO   | PRI | NULL    |       |
+-------------+----------+------+-----+---------+-------+
2 rows in set (0.04 sec)
```

Student_course table

```
mysql>
mysql> desc student_course;
+-------------+----------+------+-----+---------+-------+
| Field       | Type     | Null | Key | Default | Extra |
+-------------+----------+------+-----+---------+-------+
| Stu_id      | char(10) | NO   | PRI | NULL    |       |
| Course_code | char(10) | NO   | PRI | NULL    |       |
+-------------+----------+------+-----+---------+-------+
2 rows in set (0.10 sec)
```

Attendance table

```
mysql> DESC Attendance;
+-------------+-------------+------+-----+---------+-------+
| Field       | Type        | Null | Key | Default | Extra |
+-------------+-------------+------+-----+---------+-------+
| Stu_id      | char(10)    | NO   | PRI | NULL    |       |
| Course_code | char(10)    | NO   | PRI | NULL    |       |
| Date        | date        | NO   | PRI | NULL    |       |
| Type        | varchar(20) | YES  |     | NULL    |       |
| Sub_status  | varchar(12) | NO   | PRI | NULL    |       |
+-------------+-------------+------+-----+---------+-------+
5 rows in set (0.00 sec)
```

Table views

```
mysql> SHOW FULL TABLES WHERE TABLE_TYPE='VIEW';
+------------------------+------------+
| Tables_in_techmis      | Table_type |
+------------------------+------------+
| attendance_count       | VIEW       |
| attendance_percentage  | VIEW       |
| ca_marks               | VIEW       |
| ca_marks_percentage    | VIEW       |
| cgpa                   | VIEW       |
| display_marks          | VIEW       |
| eligibility            | VIEW       |
| eligibility_ca         | VIEW       |
| eligible_with_medical  | VIEW       |
| final_eligibility      | VIEW       |
| final_marks            | VIEW       |
| final_quiz_marks       | VIEW       |
| grade_points           | VIEW       |
| mid_marks              | VIEW       |
| not_eligible_only      | VIEW       |
| sgpa                   | VIEW       |
+------------------------+------------+
16 rows in set (0.04 sec)
```

Tools and technologies that you have used

Draw.io:

- Used to draw ER diagram, relational schema.

MYSQL, Notepad and Microsoft Word:

- Used to create database and maintain.
- Collect some sample Data.

GitHub and GitHub Desktop:

- Version Control

Security measures that we have taken to protect our Database.

- Admin - With All privileges with Grant Option for all the tables in the database.
- Dean - With All privileges without Grant for all the tables in the database.
- Lecturer – All privileges without Grant and user creation for all the tables in the database.
- Technical Officer - Read, write and update permissions for attendance related tables/views.
- Student - Read permission for final attendance and final marks/Grades tables/views.
- We set password to the users to access to the database as a security option.

Brief description about DB Accounts/Users and the reasons for creating such Accounts/Users.

Our Learning Management System contain below user accounts:

- Admin
  - Admin can access full tables in our database.
  - Maintain database.
- Dean
  - Only accessibility to the database to review.

- Lecturer

    - Access to all tables with user creation

- Technical Officer
    - Access to Read, write and update the tables and permissions for changes in attendance related tables/views.
- Student
    - Access for attendance, marks and grade tables for only view.

Code snippets to support your work.

```
==========================================================================
ATTENDANCE WITH PERCENTAGE AND ELIGIBILITY BY GIVING REGISTRATION NUMBER
==========================================================================

DELIMITER //

CREATE PROCEDURE GetStudentAttendance(IN
reg_no CHAR(10))
BEGIN
   SELECT
      Student.Stu_id,
      Student.Name,
      Student.Email,
      Attendance_percentage.Course_code,
      Attendance_percentage.Attendance_Percentage,
      CASE
         WHEN
Attendance_percentage.Attendance_Percentage >
80.0000 THEN 'Eligible'
         ELSE 'Not Eligible'
      END AS Attendance_Eligibility
   FROM
      Student
   JOIN
      Attendance_percentage ON Student.Stu_id =
Attendance_percentage.Stu_id
   WHERE
      Student.Stu_id = reg_no;
END //

DELIMITER ;

call GetStudentAttendance('stu_22_01');
```

```
mysql> call GetStudentAttendance('stu_22_01');
+-----------+-------------+-----------------+-------------+----------------------+------------------------+
| Stu_id    | Name        | Email           | Course_code | Attendance_Percentage | Attendance_Eligibility |
+-----------+-------------+-----------------+-------------+----------------------+------------------------+
| Stu_22_01 | Ishan Silva | ishan@gmail.com | ENG1222     |             100.0000 | Eligible               |
| Stu_22_01 | Ishan Silva | ishan@gmail.com | ICT1212     |              66.6667 | Not Eligible           |
| Stu_22_01 | Ishan Silva | ishan@gmail.com | ICT1222     |             100.0000 | Eligible               |
```

==========

BY GIVING REGISTRATION NUMBER AND COURSE CODE

==========

```
DELIMITER //

CREATE PROCEDURE StudentAttendance(IN reg_no
CHAR(10), IN course_code CHAR(10))
BEGIN
   SELECT
      Student.Stu_id,
      Student.Name,
      Student.Email,
      Attendance_percentage.Course_code,
      Attendance_percentage.Attendance_Percentage,
      CASE
         WHEN
Attendance_percentage.Attendance_Percentage          >
80.0000 THEN 'Eligible'
         ELSE 'Not Eligible'
      END AS Attendance_Eligibility
   FROM
      Student
   JOIN
      Attendance_percentage    ON    Student.Stu_id    =
Attendance_percentage.Stu_id
   WHERE
      Student.Stu_id = reg_no
      AND     Attendance_percentage.Course_code     =
course_code;
END //

DELIMITER ;
```

CALL StudentAttendance('stu_22_01', 'ENG1222');

```
mysql> CALL StudentAttendance('stu_22_01', 'ENG1222');
+-----------+-------------+-----------------+-------------+----------------------+-----------------------+
| Stu_id    | Name        | Email           | Course_code | Attendance_Percentage | Attendance_Eligibility |
+-----------+-------------+-----------------+-------------+----------------------+-----------------------+
| Stu_22_01 | Ishan Silva | ishan@gmail.com | ENG1222     |             100.0000 | Eligible              |
+-----------+-------------+-----------------+-------------+----------------------+-----------------------+
1 row in set (0.00 sec)
```

==========

GET ELIGIBILITY FOR THEORY BY GIVING STUDENT NO AND COURSE CODE

==========

```
DELIMITER //

CREATE PROCEDURE GetTheoryAttendance(IN reg_no CHAR(10), IN course_code CHAR(10))
BEGIN
  SELECT
    Student.Stu_id,
    Student.Name,
    Attendance_count.Course_code,
    Attendance_count.sub_status,
    Attendance_count.Present_Count,
    (Attendance_count.Present_Count / 15) * 100 AS Attendance_Percentage,
    CASE
      WHEN (Attendance_count.Present_Count / 15) * 100 > 80.0000 THEN 'Eligible'
      ELSE 'Not Eligible'
    END AS Attendance_Eligibility
  FROM
    Student
  JOIN
    Attendance_count ON Student.Stu_id = Attendance_count.Stu_id
  WHERE
    Student.Stu_id = reg_no
    AND Attendance_count.Course_code = course_code
    AND Attendance_count.sub_status = 'Theory';
END //
```

```
DELIMITER ;

CALL GetTheoryAttendance('stu_22_01', 'ICT1212');
```

```
mysql> CALL GetTheoryAttendance('stu_22_01', 'ICT1212');
+-----------+------------+-------------+------------+---------------+----------------------+-----------------------+
| Stu_id    | Name       | Course_code | sub_status | Present_Count | Attendance_Percentage | Attendance_Eligibility |
+-----------+------------+-------------+------------+---------------+----------------------+-----------------------+
| Stu_22_01 | Ishan Silva | ICT1212     | theory     |            10 |              66.6667 | Not Eligible          |
+-----------+------------+-------------+------------+---------------+----------------------+-----------------------+
1 row in set (0.06 sec)
```

==========

GET PRACTICAL ELIGIBILITY BY GIVING STUDENT NO AND COURSE CODE

==========

```
DELIMITER //

CREATE PROCEDURE GetPracticalAttendance(IN
reg_no CHAR(10), IN course_code CHAR(10))
BEGIN
  SELECT
    Student.Stu_id,
    Student.Name,
    Attendance_count.Course_code,
    Attendance_count.sub_status,
    Attendance_count.Present_Count,
    (Attendance_count.Present_Count / 15) * 100 AS
Attendance_Percentage,
    CASE
      WHEN (Attendance_count.Present_Count / 15)
* 100 > 80.0000 THEN 'Eligible'
      ELSE 'Not Eligible'
    END AS Attendance_Eligibility
  FROM
    Student
  JOIN
    Attendance_count ON Student.Stu_id =
Attendance_count.Stu_id
  WHERE
    Student.Stu_id = reg_no
    AND Attendance_count.Course_code =
course_code
```

```
    AND Attendance_count.sub_status = 'practical';
END //

DELIMITER ;

CALL GetPracticalAttendance('stu_22_01', 'ICT1222');
```

```
mysql> CALL GetPracticalAttendance('stu_22_01', 'ICT1222');
+-----------+-------------+-------------+------------+---------------+----------------------+-----------------------+
| Stu_id    | Name        | Course_code | sub_status | Present_Count | Attendance_Percentage | Attendance_Eligibility |
+-----------+-------------+-------------+------------+---------------+----------------------+-----------------------+
| Stu_22_01 | Ishan Silva | ICT1222     | practical  |            15 |              100.0000 | Eligible              |
+-----------+-------------+-------------+------------+---------------+----------------------+-----------------------+
1 row in set (0.00 sec)
```

==========

CREATE PROCEDURE FOR CA MARKS BY GIVING COURSE CODE FOR WHOLE BATCH

==========

```
DELIMITER //

CREATE PROCEDURE GetCAMarksSummary(IN
course_code CHAR(10))
BEGIN
  SELECT
    CA_marks.stu_id,
    CA_marks.course_code,
    CA_marks.total_CA_marks,
    CA_marks_percentage.CA_marks_percentage

  FROM
    CA_marks
  JOIN
    CA_marks_percentage ON CA_marks.stu_id =
CA_marks_percentage.stu_id
    AND CA_marks.course_code =
CA_marks_percentage.course_code
  WHERE
    CA_marks.course_code = course_code
  GROUP BY
    CA_marks.stu_id, CA_marks.course_code;
END //

DELIMITER ;
```

```
mysql> CALL GetCAMarksSummary('ICT1222');
+----------+-------------+----------------+---------------------+
| stu_id   | course_code | total_CA_marks | CA_marks_percentage |
+----------+-------------+----------------+---------------------+
| Stu_18_07 | ICT1222    |              0 |                   0 |
| Stu_19_02 | ICT1222    |             34 |                  68 |
| Stu_19_06 | ICT1222    |             28 |                  56 |
```

==========

CREATE PROCEDURE FOR CA MARKS FOR INDVIDUAL BY GIVING STUDENT NO AND COURSE CODE

==========

```
DELIMITER //

CREATE PROCEDURE GetindividualCA (IN
Course_code CHAR(10),IN Stu_id CHAR(10))
BEGIN
        SELECT
                CA_marks.stu_id,
        CA_marks.course_code,
        CA_marks.total_CA_marks,
        CA_marks_percentage.CA_marks_percentage
        FROM
                CA_marks,CA_marks_percentage
        WHERE
                CA_marks.stu_id                 =
CA_marks_percentage.stu_id
        AND         CA_marks.course_code        =
CA_marks_percentage.course_code
                AND     (CA_marks.course_code   =
course_code
                AND CA_marks.stu_id = stu_id)
        GROUP BY
                CA_marks.stu_id,
CA_marks.course_code;
END //

DELIMITER ;

CALL GetindividualCA('ICT1212','stu_22_01');
```

```
mysql> CALL GetindividualCA('ICT1212','stu_22_01');
+-----------+-------------+----------------+---------------------+
| stu_id    | course_code | total_CA_marks | CA_marks_percentage |
+-----------+-------------+----------------+---------------------+
| Stu_22_01 | ICT1212     |             40 |                  80 |
+-----------+-------------+----------------+---------------------+
1 row in set (0.01 sec)
```

==========

CREATE PROCEDURE FOR CA MARKS BY GIVING REGISTRATION NO FOR WHOLE BATCH

==========

```
DELIMITER //

CREATE PROCEDURE GetCAsummary(IN Stu_id
CHAR(10))
BEGIN
        SELECT
                CA_marks.stu_id,
        CA_marks.course_code,
        CA_marks.total_CA_marks,
        CA_marks_percentage.CA_marks_percentage
        FROM
                CA_marks,CA_marks_percentage
        WHERE
                CA_marks.stu_id                 =
CA_marks_percentage.stu_id
        AND         CA_marks.course_code        =
CA_marks_percentage.course_code
                AND ( CA_marks.stu_id = stu_id)
        GROUP BY
                CA_marks.stu_id,
CA_marks.course_code;
END //

DELIMITER ;

CALL GetCAsummary('stu_22_01');
```

```
mysql> CALL GetCAsummary('stu_22_01');
+-----------+-------------+----------------+---------------------+
| stu_id    | course_code | total_CA_marks | CA_marks_percentage |
+-----------+-------------+----------------+---------------------+
| Stu_22_01 | ENG1222     |             37 |                  74 |
| Stu_22_01 | ICT1212     |             40 |                  80 |
| Stu_22_01 | ICT1222     |             42 |                  84 |
```

==========
CREATE PROCEDURE FOR FINAL MARKS FOR INDIVIDUAL BY GIVING REGISTRATON NO
==========

```
DELIMITER //

CREATE PROCEDURE Getfinalsummary(IN Stu_id
CHAR(10))
BEGIN
        SELECT
                final_marks.stu_id,
        final_marks.course_code,
        final_marks.final_mark
        FROM
                final_marks
        WHERE
                final_marks.stu_id=stu_id
        GROUP BY
                final_marks.stu_id,
final_marks.course_code;
END //

DELIMITER ;

CALL Getfinalsummary('stu_22_03');
```

```
mysql> CALL Getfinalsummary('stu_22_03');
+-----------+-------------+------------+
| stu_id    | course_code | final_mark |
+-----------+-------------+------------+
| Stu_22_03 | ENG1222     |         68 |
| Stu_22_03 | ICT1212     |          0 |
```

==========
CREATE PROCEDURE FOR FINAL MARKS FOR WHOLE BATCH
==========

```
DELIMITER //

CREATE PROCEDURE Getfinalmarkswhole(IN
course_code CHAR(10))
BEGIN
   SELECT
      final_marks.stu_id,
      final_marks.course_code,
      final_marks.final_mark
```

```
   FROM
      final_marks

   WHERE
      final_marks.course_code = course_code
   GROUP BY
      final_marks.stu_id, final_marks.course_code;
END //

DELIMITER ;

CALL  Getfinalmarkswhole('ICT1222');
```

```
mysql> CALL  Getfinalmarkswhole('ICT1222');
+-----------+-------------+------------+
| stu_id    | course_code | final_mark |
+-----------+-------------+------------+
| Stu_18_07 | ICT1222     |          0 |
| Stu_19_02 | ICT1222     |         72 |
```

==========

CHEACK ELIGIBILITY FOR CA +ATTENDANCE FOR INDIVIDUAL GIVING STU_ID & COURSE CODE

==========

```
DELIMITER //

CREATE PROCEDURE GetFinalEligibility(IN reg_no
CHAR(10), IN course_code CHAR(10))
BEGIN
   SELECT
      FINAL_eligibility.stu_id,
      FINAL_eligibility.course_code,
      FINAL_eligibility.CA_Eligibility,
      FINAL_eligibility.Eligibility_Status,
      FINAL_eligibility.Final_Exam_eligibility
   FROM
      FINAL_eligibility
   WHERE
      FINAL_eligibility.stu_id = reg_no
      AND         FINAL_eligibility.course_code         =
course_code;
END //

DELIMITER ;
```

CALL GetFINALEligibility('STU_22_04', 'ICT1212');

```
mysql> CALL GetFINALEligibility('STU_22_04', 'ICT1212');
+-----------+-------------+----------------+-------------------+---------------------------+
| Stu_id    | Course_code | CA_Eligibility | Eligibility_Status | Final_Exam_Eligibility    |
+-----------+-------------+----------------+-------------------+---------------------------+
| Stu_22_04 | ICT1212     | Not eligible   | eligible          | Not Eligible for Final Exam |
+-----------+-------------+----------------+-------------------+---------------------------+
1 row in set (0.03 sec)
```

==========

CREATE PROCEDURES GET FINAL ELIGIBILITY FOR INDIVIDUALS BY GIVING STUDENT NUMBER

==========

```
DELIMITER //

CREATE PROCEDURE GetFinalEligibilityALL(IN
reg_no CHAR(10))
BEGIN
  SELECT
    FINAL_eligibility.stu_id,
    FINAL_eligibility.course_code,
    FINAL_eligibility.CA_Eligibility,
    FINAL_eligibility.Eligibility_Status,
    FINAL_eligibility.Final_Exam_eligibility
  FROM
    FINAL_eligibility
  WHERE
    FINAL_eligibility.stu_id = reg_no;

END //

DELIMITER ;
```

CALL GetFINALEligibilityALL('STU_22_04');

```
mysql> CALL GetFINALEligibilityALL('STU_22_04');
+-----------+-------------+----------------+-------------------+---------------------------+
| Stu_id    | Course_code | CA_Eligibility | Eligibility_Status | Final_Exam_Eligibility    |
+-----------+-------------+----------------+-------------------+---------------------------+
| stu_22_04 | ENG1222     | Not eligible   | eligible          | Not Eligible for Final Exam |
| Stu_22_04 | ICT1212     | Not eligible   | eligible          | Not Eligible for Final Exam |
| Stu_22_04 | ICT1222     | Eligible       | eligible          | Eligible for Final Exam     |
```

==========
CREATE PROCEDURE FOR DISPLAY ALL MARKS FOR INDIVIDUAL
==========

```
DELIMITER //

CREATE PROCEDURE getallmarksindividual(IN
reg_no CHAR(10))
BEGIN
        SELECT
                display_marks.stu_id,
                display_marks.course_code,
                display_marks.quiz_assignment_mark,
                display_marks.mid_mark,
                display_marks.total_ca_marks,
                display_marks.final_mark
        FROM display_marks
        WHERE display_marks.stu_id=reg_no;
END //

DELIMITER ;

CALL getallmarksindividual('stu_22_01');
```

```
mysql> CALL getallmarksindividual('stu_22_01');
+-----------+-------------+---------------------+----------+----------------+------------+
| stu_id    | course_code | quiz_assignment_mark | mid_mark | total_ca_marks | final_mark |
+-----------+-------------+---------------------+----------+----------------+------------+
| Stu_22_01 | ENG1222     |                  13 |       24 |             37 |         94 |
| Stu_22_01 | ICT1212     |                  16 |       24 |             40 |          0 |
```

==========
GET FINAL GRADE BY GIVING COURSE CODE
==========

```
DELIMITER //

CREATE PROCEDURE GetStudentGrades(IN
course_code CHAR(10))
BEGIN
   SELECT
      Final_marks.Stu_id,
      Final_marks.course_code,
      Final_marks.FINAL_MARK,
      Student.Status,
```

```
      IF(Student.Status = 'repeat', 'C',
         IF(Final_marks.FINAL_MARK >= 90, 'A+',
         IF(Final_marks.FINAL_MARK >= 84, 'A',
         IF(Final_marks.FINAL_MARK >= 75, 'A-',
         IF(Final_marks.FINAL_MARK >= 70, 'B+',
         IF(Final_marks.FINAL_MARK >= 65, 'B',
         IF(Final_marks.FINAL_MARK >= 60, 'B-',
         IF(Final_marks.FINAL_MARK >= 55, 'C+',
         IF(Final_marks.FINAL_MARK >= 50, 'C',
         IF(Final_marks.FINAL_MARK >= 45, 'C-',
         IF(Final_marks.FINAL_MARK >= 40, 'D+',
         IF(Final_marks.FINAL_MARK   >=   35,   'D',
'F'))))))))))) AS Grade
   FROM
      Final_marks,student

   WHERE
         Final_marks.Stu_id = Student.Stu_id AND
      ( Final_marks.course_code = course_code);
END //

DELIMITER ;

CALL GetStudentGrades('ICT1212');
```

```
mysql> CALL GetStudentGrades('ICT1212');
+-----------+-------------+-----------+-----------+-------+
| Stu_id    | course_code | FINAL_MARK | Status    | Grade |
+-----------+-------------+-----------+-----------+-------+
| Stu_18_07 | ICT1212     |          0 | Suspend   | F     |
| Stu_19_02 | ICT1212     |         68 | Repeat    | C     |
| Stu_19_06 | ICT1212     |         68 | Repeat    | C     |
```

==========

GET FINAL GRADE BY GIVING COURSE CODE

==========

```
DELIMITER //

CREATE                                  PROCEDURE
GetStudentGradesINDIVIDUAL(IN                reg_no
CHAR(10),IN course_code CHAR(10))
BEGIN
   SELECT
      Final_marks.Stu_id,
```

```
    Final_marks.course_code,
    Final_marks.FINAL_MARK,
    Student.Status,
    IF(Student.Status = 'repeat', 'C',
      IF(Final_marks.FINAL_MARK >= 90, 'A+',
      IF(Final_marks.FINAL_MARK >= 84, 'A',
      IF(Final_marks.FINAL_MARK >= 75, 'A-',
      IF(Final_marks.FINAL_MARK >= 70, 'B+',
      IF(Final_marks.FINAL_MARK >= 65, 'B',
      IF(Final_marks.FINAL_MARK >= 60, 'B-',
      IF(Final_marks.FINAL_MARK >= 55, 'C+',
      IF(Final_marks.FINAL_MARK >= 50, 'C',
      IF(Final_marks.FINAL_MARK >= 45, 'C-',
      IF(Final_marks.FINAL_MARK >= 40, 'D+',
      IF(Final_marks.FINAL_MARK >= 35, 'D',
'F'))))))))))) AS Grade
  FROM
    Final_marks,student

  WHERE
        Final_marks.Stu_id = Student.Stu_id AND
    ( Final_marks.Stu_id = reg_no AND
Final_marks.course_code = course_code);
END //

DELIMITER ;

CALL
GetStudentGradesINDIVIDUAL('stu_22_02','ICT1212'
);
```

```
mysql> CALL GetStudentGradesINDIVIDUAL('stu_22_02','ICT1212');
+-----------+-------------+------------+------------+-------+
| Stu_id    | course_code | FINAL_MARK | Status     | Grade |
+-----------+-------------+------------+------------+-------+
| Stu_22_02 | ICT1212     |         50 | Non_repeat | C     |
+-----------+-------------+------------+------------+-------+
1 row in set (0.00 sec)
```

==========
GET GRADES FOR SUBJECT WITH MC
==========

```
DELIMITER //

CREATE PROCEDURE GetAllStudentGradesWITHMC()
BEGIN
   SELECT
      Final_marks.Stu_id,
      Final_marks.course_code,
      Final_marks.FINAL_MARK,
      Student.Status,
      IF(Medical.Reason = 'EXAM', 'MC',
         IF(Student.Status = 'repeat', 'C',
         IF(Final_marks.FINAL_MARK >= 90, 'A+',
         IF(Final_marks.FINAL_MARK >= 84, 'A',
         IF(Final_marks.FINAL_MARK >= 75, 'A-',
         IF(Final_marks.FINAL_MARK >= 70, 'B+',
         IF(Final_marks.FINAL_MARK >= 65, 'B',
         IF(Final_marks.FINAL_MARK >= 60, 'B-',
         IF(Final_marks.FINAL_MARK >= 55, 'C+',
         IF(Final_marks.FINAL_MARK >= 50, 'C',
         IF(Final_marks.FINAL_MARK >= 45, 'C-',
         IF(Final_marks.FINAL_MARK >= 40, 'D+',
         IF(Final_marks.FINAL_MARK >= 35, 'D',
'F')))))))))))) AS Grade
   FROM
      Final_marks
   JOIN
      Student ON Final_marks.Stu_id = Student.Stu_id
   LEFT JOIN
      Medical ON Final_marks.Stu_id = Medical.Stu_id
AND Final_marks.course_code = Medical.Course_code
   WHERE
      Medical.Reason IS NULL OR Medical.Reason =
'EXAM';

END //

DELIMITER ;
```

CALL GetAllStudentGradesWITHMC();

```
mysql> CALL GetAllStudentGradesWITHMC();
+-----------+-------------+-----------+------------+-------+
| Stu_id    | course_code | FINAL_MARK | Status     | Grade |
+-----------+-------------+-----------+------------+-------+
| Stu_18_07 | ENG1222     |          0 | Suspend    | F     |
| Stu_18_07 | ICT1212     |          0 | Suspend    | F     |
```

==========

GET GRADES OF A STUDENT FOR ALL THE SUBJECT BY REG_NO(INDIVIDUAL)

==========

```
DELIMITER //

CREATE PROCEDURE GetStudentGrade(IN
REG_NO VARCHAR (10))
BEGIN
   SELECT
      Final_marks.Stu_id,
      Final_marks.course_code,
      Final_marks.FINAL_MARK,
      Student.Status,
      IF(Medical.Reason = 'EXAM', 'MC',
         IF(Student.Status = 'repeat', 'C',
         IF(Final_marks.FINAL_MARK >= 90, 'A+',
         IF(Final_marks.FINAL_MARK >= 84, 'A',
         IF(Final_marks.FINAL_MARK >= 75, 'A-',
         IF(Final_marks.FINAL_MARK >= 70, 'B+',
         IF(Final_marks.FINAL_MARK >= 65, 'B',
         IF(Final_marks.FINAL_MARK >= 60, 'B-',
         IF(Final_marks.FINAL_MARK >= 55, 'C+',
         IF(Final_marks.FINAL_MARK >= 50, 'C',
         IF(Final_marks.FINAL_MARK >= 45, 'C-',
         IF(Final_marks.FINAL_MARK >= 40, 'D+',
         IF(Final_marks.FINAL_MARK >= 35, 'D',
'F')))))))))))) AS Grade
   FROM
      Final_marks
   JOIN
      Student ON Final_marks.Stu_id = Student.Stu_id
   LEFT JOIN
```

```
    Medical ON Final_marks.Stu_id = Medical.Stu_id
AND Final_marks.course_code = Medical.Course_code
  WHERE
    Final_marks.Stu_id = REG_NO
    AND (Medical.Reason IS NULL OR
Medical.Reason = 'EXAM');
END //

DELIMITER ;

CALL GETSTUDENTGRADE('STU_22_01');
```

```
mysql> CALL GETSTUDENTGRADE('STU_22_01');
+-----------+-------------+------------+------------+-------+
| Stu_id    | course_code | FINAL_MARK | Status     | Grade |
+-----------+-------------+------------+------------+-------+
| Stu_22_01 | ENG1222     |         94 | Non_repeat | A+    |
| Stu_22_01 | ICT1222     |         74 | Non_repeat | B+    |
```

==========

CREATE PROCEDURE FOR GET SGPA AND CGPA FOR WHOLE BATCH

==========

```
DELIMITER //

CREATE PROCEDURE SGPA_CGPA_FOR_ALL()
BEGIN
        SELECT

        SGPA.stu_id,SGPA.SGPA,CGPA.CGPA
        FROM SGPA,CGPA
        WHERE SGPA.stu_id=CGPA.stu_id;
END //

DELIMITER ;

CALL SGPA_CGPA_FOR_ALL();
```

```
mysql> CALL SGPA_CGPA_FOR_ALL();
+-----------+---------+---------+
| stu_id    | SGPA    | CGPA    |
+-----------+---------+---------+
| Stu_18_07 | 0.00000 | 0.00000 |
| Stu_19_02 | 2.00000 | 2.00000 |
```

==========

CREATE PROCEDURE FOR GET SGPA AND CGPA FOR INDIVIDUAL

==========

```
DELIMITER //

CREATE PROCEDURE SGPA_CGPA_FOR_INDIVIDUAL(IN stu_id CHAR(10))
BEGIN
	SELECT

	SGPA.stu_id,SGPA.SGPA,CGPA.CGPA
	FROM SGPA,CGPA
	WHERE SGPA.stu_id=CGPA.stu_id AND SGPA.stu_id=stu_id;
END //

DELIMITER ;

CALL SGPA_CGPA_FOR_INDIVIDUAL('stu_22_01');
```

```
mysql> CALL SGPA_CGPA_FOR_INDIVIDUAL('stu_22_01');
+-----------+---------+---------+
| stu_id    | SGPA    | CGPA    |
+-----------+---------+---------+
| Stu_22_01 | 3.14211 | 3.04118 |
+-----------+---------+---------+
1 row in set (0.01 sec)
```

==========
CREATE VIEW FOR GET ATTENDANCE COUNT
==========

```
CREATE VIEW Attendance_count AS
SELECT
   attendance.Stu_id,
   attendance.Course_code,
   attendance.sub_status,
   COUNT(IF(attendance.type = 'present', 1, NULL)) AS Present_Count
FROM
   attendance
JOIN
   student ON attendance.Stu_id = student.Stu_id
WHERE
   student.status = 'non_repeat'
GROUP BY
   attendance.Stu_id,
   attendance.Course_code,
   attendance.sub_status;
```

```
mysql> select * from Attendance_count ;
+-----------+-------------+------------+---------------+
| Stu_id    | Course_code | sub_status | Present_Count |
+-----------+-------------+------------+---------------+
| stu_22_01 | ENG1222     | theory     |            15 |
| Stu_22_01 | ICT1212     | theory     |            10 |
| Stu_22_01 | ICT1222     | practical  |            15 |
```

==========
CREATE VIEW FOR GET ATTENDANCE PERCENTAGE
==========

```
CREATE VIEW Attendance_percentage AS
SELECT
   Stu_id,
   Course_code,
   sub_status,
   Present_Count,
   (Present_Count / 15) * 100 AS Attendance_Percentage
FROM
   Attendance_count;
```

```
mysql> select * from Attendance_percentage ;
+-----------+-------------+------------+---------------+----------------------+
| Stu_id    | Course_code | sub_status | Present_Count | Attendance_Percentage |
+-----------+-------------+------------+---------------+----------------------+
| stu_22_01 | ENG1222     | theory     |            15 |             100.0000 |
| Stu_22_01 | ICT1212     | theory     |            10 |              66.6667 |
```

==========

CREATE VIEW FOR GET ATTENDANCE ELIGIBILITY

==========

```
CREATE VIEW Eligibility AS
SELECT
   Stu_id,
   Course_code,
   sub_status,
   Attendance_Percentage,
   IF(Attendance_Percentage > 80.0000, 'eligible', 'not eligible') AS Eligibility_Status
FROM
   Attendance_percentage;
```

```
mysql> select * from Eligibility;
+-----------+-------------+------------+----------------------+-------------------+
| Stu_id    | Course_code | sub_status | Attendance_Percentage | Eligibility_Status |
+-----------+-------------+------------+----------------------+-------------------+
| stu_22_01 | ENG1222     | theory     |              100.0000 | eligible          |
| Stu_22_01 | ICT1212     | theory     |               66.6667 | not eligible      |
```

==========

CREATE VIEW FOR GET TOTAL CA MARKS

==========

```
CREATE VIEW CA_marks AS
SELECT
stu_id,
course_code,
Quiz,
Assignment,
theory_mid,
practical_mid,
        SUM(quiz+assignment+theory_mid+practical_mid)AS total_CA_marks
FROM marks
GROUP BY stu_id,course_code;
```

```
mysql> select * from CA_marks ;
+-----------+-------------+------+------------+------------+--------------+---------------+
| stu_id    | course_code | Quiz | Assignment | theory_mid | practical_mid | total_CA_marks |
+-----------+-------------+------+------------+------------+--------------+---------------+
| Stu_18_07 | ENG1222     |    0 |          0 |          0 |            0 |             0 |
| Stu_18_07 | ICT1212     |    0 |          0 |          0 |            0 |             0 |
```

==========

CREATE VIEW FOR GET CA MARKS PERCENTAGE

==========

```
CREATE VIEW CA_marks_percentage AS
SELECT
stu_id,
course_code,
Quiz,
Assignment,
theory_mid,
```

```
practical_mid,
total_CA_marks,
        (total_ca_marks*2) AS CA_marks_percentage
FROM CA_marks
GROUP BY stu_id,course_code;
```

```
mysql> select * from CA_marks_percentage ;
+-----------+-------------+------+------------+------------+--------------+---------------+--------------------+
| stu_id    | course_code | Quiz | Assignment | theory_mid | practical_mid | total_CA_marks | CA_marks_percentage |
+-----------+-------------+------+------------+------------+--------------+---------------+--------------------+
| Stu_18_07 | ENG1222     |    0 |          0 |          0 |             0 |             0 |                  0 |
| Stu_18_07 | ICT1212     |    0 |          0 |          0 |             0 |             0 |                  0 |
```

==========

CREATE VIEW FOR GET CA ELIGIBILITY

==========

```
CREATE VIEW Eligibility_CA AS
SELECT
stu_id,
course_code,
total_CA_marks,
CA_marks_percentage,
        IF(CA_marks_percentage>50,'Eligible','Not eligible')AS CA_Eligibility
FROM   CA_marks_percentage
GROUP BY stu_id,course_code;
```

```
mysql> select * from Eligibility_CA ;
+-----------+-------------+---------------+--------------------+-----------------+
| stu_id    | course_code | total_CA_marks | CA_marks_percentage | CA_Eligibility |
+-----------+-------------+---------------+--------------------+-----------------+
| Stu_18_07 | ENG1222     |             0 |                  0 | Not eligible    |
| Stu_18_07 | ICT1212     |             0 |                  0 | Not eligible    |
```

==========

CREATE VIEW FOR GET ELIGIBILITY FOR FINAL EXAM

==========

```
CREATE VIEW Final_Eligibility AS
SELECT
   e.Stu_id,
   e.Course_code,
   e.Eligibility_Status,
   ca.CA_Eligibility,
   CASE
      WHEN e.Eligibility_Status = 'eligible' AND ca.CA_Eligibility = 'Eligible' THEN 'Eligible for Final Exam'
      ELSE 'Not Eligible for Final Exam'
   END AS Final_Exam_Eligibility
FROM
   Eligibility e
JOIN
   Eligibility_CA ca ON e.Stu_id = ca.stu_id AND e.Course_code = ca.course_code;
```

==========

CREATE VIEW FOR GET FINAL MARKS

==========

```
CREATE VIEW Final_marks AS
   SELECT
   Stu_id,course_code,((SUM(theory_end+practical_end))*2) AS FINAL_MARK
   FROM marks
GROUP BY stu_id,course_code;
```

```
mysql> select * from Final_marks ;
+-----------+-------------+------------+
| Stu_id    | course_code | FINAL_MARK |
+-----------+-------------+------------+
| Stu_18_07 | ENG1222     |          0 |
| Stu_18_07 | ICT1212     |          0 |
```

==========

CREATE VIEW FOR GET TOTAL MID MARKS

==========

```
CREATE VIEW mid_marks AS
   SELECT
   stu_id,course_code,(SUM(theory_mid+practical_mid))AS mid_mark
   FROM marks
   GROUP BY stu_id,course_code;
```

```
mysql> select * from mid_marks ;
+-----------+-------------+----------+
| stu_id    | course_code | mid_mark |
+-----------+-------------+----------+
| Stu_18_07 | ENG1222     |        0 |
| Stu_18_07 | ICT1212     |        0 |
```

==========

CREATE VIEW FOR GET TOTAL QUIZ + ASSIGNMENT MARKS

==========

```
CREATE VIEW final_quiz_marks AS
   SELECT
   stu_id,course_code,(SUM(quiz+assignment))AS quiz_assignment_mark
   FROM marks
   GROUP BY stu_id,course_code;
```

```
mysql> select * from final_quiz_marks ;
+-----------+-------------+---------------------+
| stu_id    | course_code | quiz_assignment_mark |
+-----------+-------------+---------------------+
| Stu_18_07 | ENG1222     |                   0 |
| Stu_18_07 | ICT1212     |                   0 |
```

==========
CREATE VIEW FOR DISPLAY ALL MARKS
==========

```
CREATE VIEW display_marks AS
SELECT
   final_marks.stu_id,
   final_marks.course_code,
   final_quiz_marks.quiz_assignment_mark,
   mid_marks.mid_mark,
   ca_marks.total_ca_marks,
   final_marks.final_mark
FROM
   final_marks, final_quiz_marks, mid_marks, ca_marks
WHERE
   final_marks.stu_id = final_quiz_marks.stu_id
   AND final_marks.course_code = final_quiz_marks.course_code
   AND final_marks.stu_id = mid_marks.stu_id
   AND final_marks.course_code = mid_marks.course_code
   AND final_marks.stu_id = ca_marks.stu_id
   AND final_marks.course_code = ca_marks.course_code
GROUP BY
   final_marks.stu_id, final_marks.course_code;
```

```
mysql> select * from display_marks ;
+-----------+-------------+----------------------+----------+----------------+------------+
| stu_id    | course_code | quiz_assignment_mark | mid_mark | total_ca_marks | final_mark |
+-----------+-------------+----------------------+----------+----------------+------------+
| Stu_18_07 | ENG1222     |                    0 |        0 |              0 |          0 |
| Stu_18_07 | ICT1212     |                    0 |        0 |              0 |          0 |
```

==========
CREATE VIEW FOR GET GRADE POINT FOR GPA CALCULATION
==========

```
CREATE VIEW Grade_Points AS
SELECT
   FINAL_MARKS.Stu_id,
   FINAL_MARKS.course_code,
   FINAL_MARK,
   STUDENT.Status,
   IF(Status = 'repeat', 2.0,
      IF(FINAL_MARK >= 90, 4.0,
      IF(FINAL_MARK >= 84, 4.0,
      IF(FINAL_MARK >= 75, 3.7,
      IF(FINAL_MARK >= 70, 3.3,
      IF(FINAL_MARK >= 65, 3.0,
      IF(FINAL_MARK >= 60, 2.7,
      IF(FINAL_MARK >= 55, 2.3,
      IF(FINAL_MARK >= 50, 2.0,
      IF(FINAL_MARK >= 45, 1.7,
```

```
      IF(FINAL_MARK >= 40, 1.3,
      IF(FINAL_MARK >= 35, 1.0, 0.0))))))))))) AS Grade_Point
FROM
   Final_marks,student
WHERE
    Final_marks.Stu_id = Student.Stu_id;
```

```
mysql> select * from Grade_Points ;
+----------+-------------+-----------+-----------+-------------+
| Stu_id   | course_code | FINAL_MARK | Status    | Grade_Point |
+----------+-------------+-----------+-----------+-------------+
| Stu_18_07 | ENG1222    |          0 | Suspend   |         0.0 |
| Stu_18_07 | ICT1212    |          0 | Suspend   |         0.0 |
```

==========

CREATE VIEW FOR CALCULATE SGPA

==========

```
CREATE VIEW SGPA AS
SELECT
   grade_points.stu_id,
   (SUM(grade_points.grade_point * course_unit.credit) / SUM(course_unit.credit)) AS SGPA
FROM
   grade_points,student_course,course_unit
where
   grade_points.course_code = student_course.course_code
AND
   student_course.course_code = course_unit.course_code
GROUP BY
   grade_points.stu_id;
```

```
mysql> select * from SGPA;
+----------+---------+
| stu_id   | SGPA    |
+----------+---------+
| Stu_18_07 | 0.00000 |
| Stu_19_02 | 2.00000 |
```

==========

CREATE VIEW FOR CALCULATE CGPA

==========

```
CREATE VIEW CGPA AS
SELECT
   grade_points.stu_id,
   (SUM(grade_points.grade_point * course_unit.credit) / SUM(course_unit.credit)) AS CGPA
FROM
   grade_points,student_course,course_unit
where
   grade_points.course_code = student_course.course_code
AND
   student_course.course_code = course_unit.course_code AND course_unit.course_code!='ENG1222'
GROUP BY
```

grade_points.stu_id;

```
mysql> select * from CGPA;
+-----------+---------+
| stu_id    | CGPA    |
+-----------+---------+
| Stu_18_07 | 0.00000 |
| Stu_19_02 | 2.00000 |
```

==========

Problems that we faced during the development of the solution.

- How to create the user accounts in ER separate entities or as a disjoint.
- When entering data for the attendance table for the fifteen weeks.
- Faced difficulty in making foreign key references between medical and attendance table.
- Faced difficulty in retrieving data under the condition of eligibility with medical.

Solutions/how we have overcome the above identified problems.

- We create separate entities for the users in our final ER.
- We assume fourteen weeks for lectures and another week for exam.
- We use student id and course code as foreign key.
- We unable to fulfill the condition.

New database technologies/trends that we have used to develop the backend.

- MySQL
- Notepad
- Microsoft Word
- GitHub

If you are going to host your backend, where are you going to host it and reasons for the selection.

Cloud environment,

- Can access by everyone anywhere at any time
- Backup and recovery
- Cost efficiency
- security

If you are going to host your backend in a cloud environment what are the things/changes that you have to do in your backend.

- We want to buy a database server
- Establish backup processes.
- Modify the database schema

Individual contribution to the backend development

TG Number	Contribution
TG/2022/1394	• Collect data • Create tables insert data • Create views and procedures • Draw ER diagram • Create final report
TG/2022/1372	• Collect data • Create tables insert data • Create views and procedures • Draw final ER ,RM

TG/2022/1378	• Collect data • Create tables insert data • Create views and procedures • Create SRS Report • Create final report
TG/2022/1400	• Collect data • Create tables insert data • Create views and procedures • Create RM • Create final report

References

Lecture Notes

W3Schools.com

You tube videos