Topic: **STUDENT MANAGEMENT SYSTEM**

SUBMITTED TO:

DR ASIF SHOHAIL

SUBMITTED BY:

- AQDAS SHABBIR ___ BITF21M022
- MUHAMMAD JAFAR ___ BITF21M032
- MUHAMMAD HASEEB ___ BITF21M023

This page is intentionally left blank

Introduction

This student management system is a comprehensive software solution that is used to enhance the administrative process

At its core, the system manages faculty-related information, including faculty identification (Faculty_ID) and associated details such as faculty names (F_name). The hierarchical structure of faculties and departments is maintained through the connection between the Faculty and Department tables, where each department is linked to a unique identifier (dept_ID) and references the faculty it belongs to through the use of foreign keys (Facult_ID).

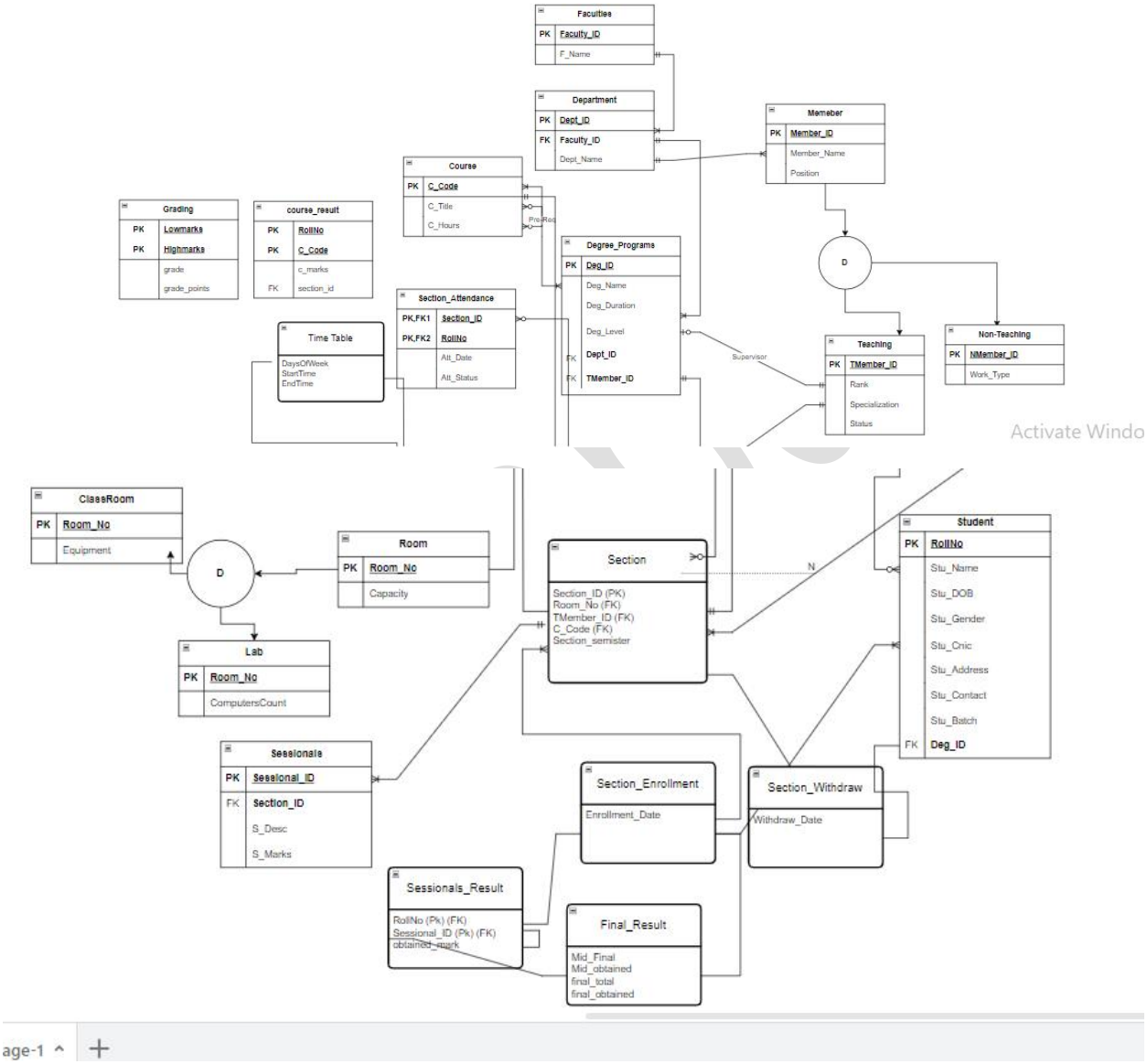
The system also caters to the diverse roles within the academic environment through its Member, TEACHING, and NON_TEACHING entities. Members are individuals associated with academic positions, and TEACHING and NON_TEACHING serve as subtypes representing academic and non-academic roles, respectively.

For academic programs and student management, the system incorporates the DEGREE_PROGRAMS and STUDENT tables. DEGREE_PROGRAMS captures details of academic programs, including degree identification (degree_ID) and program-specific information, with the added feature of linking a faculty member as a supervisor (TMem_ID). Meanwhile, STUDENT records store essential student information such as roll numbers (rollNo), names, dates of birth, and contact details, all organized by department.

The course-related aspects are handled through the COURSES, PRE_COURSE, and DEGREE_COURSES tables. COURSES contain course details, including course codes (C_Code), titles, and hours. The association between courses is maintained in the PRE_COURSE table, ensuring prerequisites are identified. The DEGREE_COURSES table manages the many-to-many relationship between academic degrees and courses.

To facilitate classroom and schedule management, the system includes the ROOM, CLASSROOM, LAB, SECTION, SECTION_ENROLLMENT, SECTION_WITHDRAW, SECTION_ATTENDANCE, TIME_TABLE, SESSIONAL, SESSIONAL_RESULT, FINAL_RESULT, and GRADING tables. These tables collectively manage room information, class schedules, student enrollment, attendance tracking, sessional assessments, and final results.

ERD OF THE STUDENT MANAGEMENT SYSTEM



TRANSFORMATION INTO RELATIONS

FACULIES (Faculty_ID,F_name) *:. Here **Faculty_ID = PK***

Department (dept_ID,dept_name, **faculty_ID)** :. *Here **dept_ID = PK** and **Facult_ID = FK***

SUBER-TYPE:

Member(Member_ID,member_name,position,**dept_ID**) *:. Here **Member_ID = PK and dep_ID = FK***

SUBTYPE:

TEACHING(TMember_ID,Specialization,Status) *:. Here **TMember_ID = PK***

NON_TEACHING (NMember_ID, Work_Type) *:.Here **NMember_ID = PK***

DEGREE_PROGRAMS(degree_ID,degree_name,duration,level,**dept_ID,TMem_ID)**

*Here:. **Degree_ID = Pk and dept_ID = FK and TMem_ID = FK (Supervisor)***

STUDENT(rollNO,stu_name,stu_DOB,stu_Gender,stu_CNIC,stu_Address,Batch,**dept_ID**,contact)

*Here:. **rollNo = PK and dept_ID = fk***

COURSES(C_Code,C_title,C_hours) *:. Here **C_Code = PK***

PRE_COURSE(P_Code,C_Code) *:. Here **P_code,C_Code = PK and C_Code = FK***

DEGREE_COURSES(degree_ID,C_code)

*:. Here **degree_ID, C_code = PK and degree_ID = Fk and C_code = FK (due to many to many relation)***

SUPERTYPE:

ROOM(Room_No,capacity,Type) *:. Here **Room_No = pk***

SUBTYPE:

CLASSROOM(Room_NO, Equiment)

LAB(Room_NO,Computer_count)

SECTION(section_ID,section_semister,**TMem_ID,C_Code,Room_No)**

:. Here ***section_ID = Pk and TMemp_ID = FK and C_Code = Fk and Room_NO = FK***

SECTION_ENROLLMENT(section_ID,rollNo,Enrollment_date**)**

:. *Here* ***section_ID,rollNo = PK and sectionID = Fk and rollNo = FK***

SECTION_WITHDRAW(section_ID,rollNo,withdraw_date**)**

:. *Here* ***section_ID,rollNo = PK and section_ID = Fk and rollNo = FK***

SECTION_ATTENDANCE(section_ID,RollNO,Attendance_Date,Att_Status)

:. Here ***sectionID,rollNo,Attenddance_Date = PK and section_ID = Fk and rollNo = FK***

TIME_TABLE(section_ID,Room_ID,DayOfWeek,StartTime,EndTime)

:. Here ***section_ID ,room_ID = PK and section_ID = FK and Room_ID = FK***

SESSIONAL(sessional_ID,section_ID,sessional_Type,marks)

:. Here ***sessional_ID = PK and section_ID = FK***

SESSIONAL_RESULT(sessional_ID,rollNo,obt_marks)

:. *Here* ***sessional_ID,rollNo = PK and sessional_ID = FK and rollNo = FK***

FINAL_RESULT(sessional_ID,rollNo,mid_total,mid_obt,final_total,final_obt)

:. Here ***sessional_ID,rollNo = PK and also FK referring to sessional_result and rollNo = Fk referring to section_enrollment***

GRADING(lowmarks,**himarks**,grade,points)

:. Here **lomarks,himarks = PK**

COURSE(C_code,section_id,c_marks,rollNo) Here:. C_code,rollNo = PK and section_id and rollNo = fk and c_code = fk

TOP DOWN APPROACH

Now we evaluate each table using normalization

• First Normal Form:

For the first normal form (1NF) to be hold in a relation, there exists multi-value attribute contact in a student.

STUDENT(studentID,stu_name,DOB,Gender,CNIC,Address,Batch,**deptID**) Here:. **studentID = PK and deptID = fk**

STUDENT_CONTACT(studentID,CONTACT) Here:. **StudentID,contact = PK (due to multi value attribute)**

Note: So, all the tables remain the same, no splitting required

• Second Normal Form

For the second normal form (2NF) to be hold in a relation, there doesn't exist any **partial-dependency** ina relation which means that

Note: Such relation doesn't exist in any of the relation. To such relation to be hold the primary key must be composite.

• Third Normal Form

For the third Normal form (3NF) to be hold in a relation, there doesn't exist any transitive dependency in a relationship which means Non-Prime Attribute → Non-Prime Attribute

Note: Such relation doesn't exist in any of the relation.

Relational schema

Faculty

Faculty_id	Faculty_name

Department

Dept_id	Dept_name	Faculty_id

Member (Super Type)

Member_id	Member_name	position	Dept_id

Teaching(sub type of member)

T_member_id	specialization	Status

Non Teaching(sub type of member)

N_meber_id	Work_type

Degree Programs

Degree_id	Degree_name	duration	Level	Dept_id	T_member_id

Students

rollNO	stu_name	stu_DOB	stu_Gender	stu_CNIC	stu_Address	Batch	dept_ID	,contact

Courses

C_code	C_title	C_hours

Pre_Courses

P_code	C_code

Degree courses

Degree_id	C_code

Room (super type)

Room_no	Capacity	type

Class room(subtype of Room)

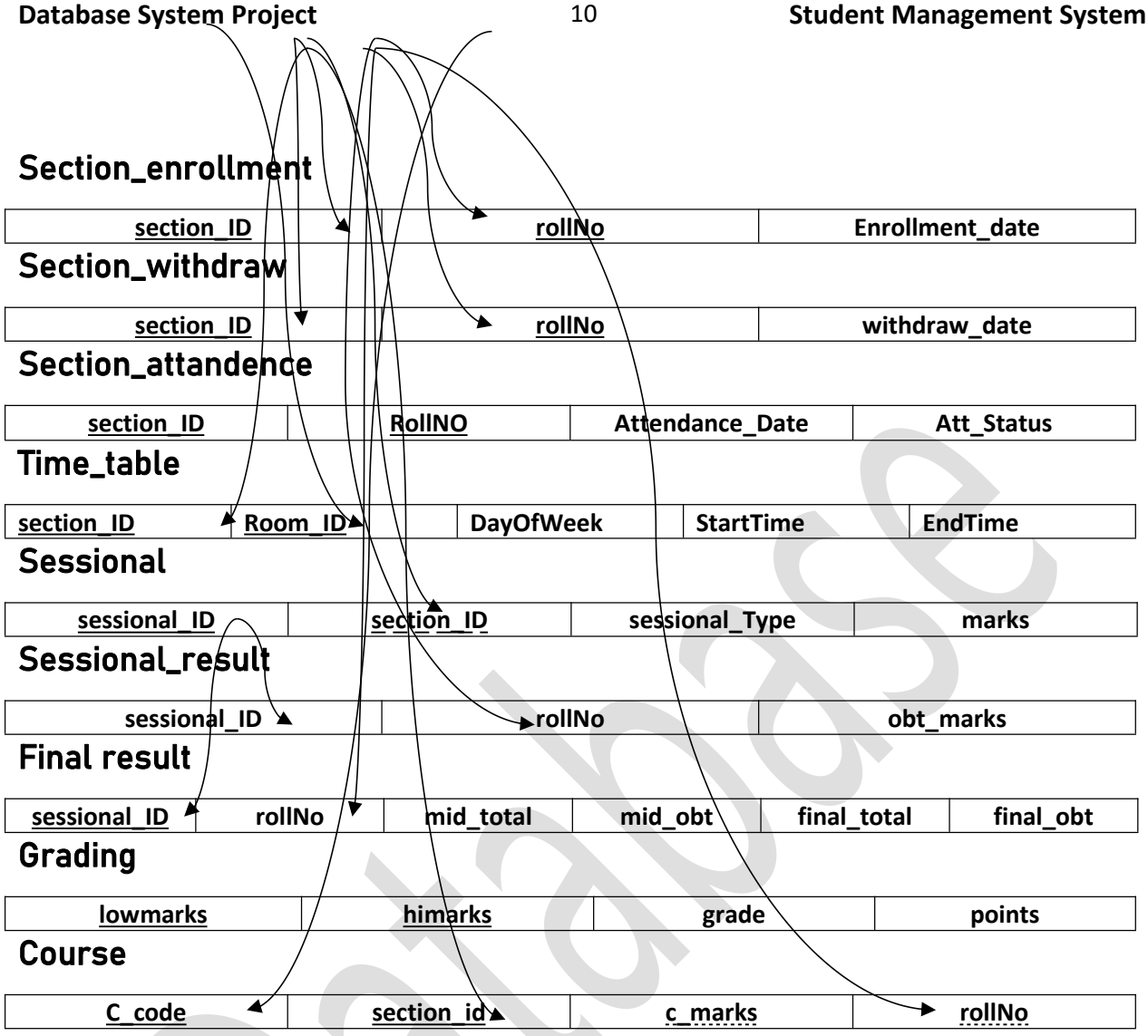
Room_no	Equiment

Labs (subtype of Room)

Room_no	Computer_count

Section

section_ID	section_semister	TMem_ID	C_Code	Room_No


Section_enrollment
section_ID
rollNo
Enrollment_date
Section_withdraw
section_ID
rollNo
withdraw_date
Section_attandence
section_ID
RollNO
Attendance_Date
Att_Status
Time_table
section_ID
Room_ID
DayOfWeek
StartTime
EndTime
Sessional
sessional_ID
section_ID
sessional_Type
marks
Sessional_result
sessional_ID
rollNo
obt_marks
Final result
sessional_ID
rollNo
mid_total
mid_obt
final_total
final_obt
Grading
lowmarks
himarks
grade
points
Course
C_code
section_id
c_marks
rollNo

BOTTOM UP APPORACH

FACULTIES(Faculty_ID**,**F_name, dept_ID**,**dept_name)

Dept_id = pk

So in this the relation is in 1NF and 2NF but not in 3NF because faculty_id → f_name

FACULTIES(**Faculty_ID**,F_name)

Department(**dept_id**,dept_name,faculty_id)

Department(dept_id**,**member_name, degree_ID**,**degree_name,duration,level)

Here:.**degree_id = pk**

So it is in 1NF and 2NF but not in 3NF because dept_Id→dept_name

So,

Department(**dept_id**,dept_name,faculty_id)

Degree_program(**degree_id**,degree_name,duration,level,dept_id)

Department(**dept_id**,dept_name,faculty_id, Member_id,member_name,position)

Here :. Member_id = pk

So it is in 1NF and 2NF but not in 3NF because dept_Id→dept_name

So,

Department(dept_id,dept_name,faculty_name)

Member(**member_id**,member_name,position,dept_id)

Degree_program(**degree_id**,degree_name,duration,level,dept_id,rollNo,stu_name,stu_batch,gender,cnic)

Here:. rollNo = pk

So it is in 1NF and 2NF but not in 3NF because degree_Id→degree_name

So,

Degree_program(**degree_id**,degree_name,duration,level,dept_id)

Student(**rollNo**,stu_name,stu_batch,gender,cnic,degree_id)

Degree_program(**degree_id**,degree_name,duration,level,dept_id,C_Code,C_title,C_hours)

Since there can be multiple course offered in a degree programs so

Degree_program(**degree_id**,degree_name,duration,level,dept_id)

Course(**degree_id,C_Code**,C_title,C_hours)

In Course C_code →c_title,c_hours

Course(**degree_id,C_Code)**

Course(C_Code,C_title,C_hours)

FACULTIES(Faculty_ID, F_name)

DEPARTMENT(Dept_ID, Dept_name, Faculty_ID)

MEMBER(Member_ID, Member_name, Position, Dept_ID)

MEMBER(Member_ID, Member_name, Position, Dept_ID)

TEACHING(TMember_ID, Specialization, Status)

NON_TEACHING(NMember_ID, Work_Type)

DEGREE_PROGRAMS(Degree_ID, Degree_name, Duration, Level, Dept_ID, TMem_ID)

STUDENT(RollNo, Stu_name, Stu_DOB, Stu_Gender, Stu_CNIC, Stu_Address, Batch, Dept_ID, Contact)

PRE_COURSE(P_Code, C_Code

Relationship and Connectivity

Entity – Relationship Diagram of system

The following Entity-Relationship Diagram (ERD) represents the relationships of different entities in this database system with respect to their attributes and primary as well as foreign keys. It also representsthe 1:1, 1:M and M:N relationships within this system.

Connectivity Table

Entity	Relationship	Connectivity	Entity
FACULTIES	HAS	1:M	DEPARTMENTS
DEPARTMENTS	HAS	1:M	MEMBERS
DEPARTMENTS	OFFERS	1:M	DEGREE_PROGRAMS
TEACHING	SUPERVISE	1:1	DEGREE_PROGRAMS
STUDENTS	ENROLLED	M:1	DEGREE_PROGRAMS
DEGREE_PROGRAMS	HAS	M:M	COURSE
COURSES	HAS	1:M	SECTIONS
STUDENTS	ENROLLED	1:M	SECTIONS
SECTIONS	HAS	1:M	SECTION_ATTENDANCE
SECTION	HAS	1:M	SESSIONALS
SESSIONAL	HAS	1:M	SESSIONAL_RESULT
ROOM	HAS	1:M	SECTION

1. **FACULTIES - Department:**

- **Relationship:** One-to-Many
- **Connectivity:** Each Faculty can be associated with multiple Departments, but each Department is associated with only one Faculty.

2. **Member (Super-Type) - TEACHING (Sub-Type) / NON_TEACHING (Sub-Type):**
    - **Relationship:** Disjoint Subtypes
    - **Connectivity:** Member is the super-type, and TEACHING and NON_TEACHING are the subtypes. Each Member is associated with either TEACHING or NON_TEACHING.
3. **DEGREE_PROGRAMS - Member (Supervisor):**
    - **Relationship:** Many-to-One
    - **Connectivity:** Each DEGREE_PROGRAM can have one supervisor (TMem_ID), and a Member (supervisor) can be associated with multiple DEGREE_PROGRAMS.
4. **STUDENT - Department:**
    - **Relationship:** Many-to-One
    - **Connectivity:** Each STUDENT is associated with one Department, and a Department can have multiple STUDENTS.
5. **COURSES - PRE_COURSE:**
    - **Relationship:** One-to-Many
    - **Connectivity:** Each COURSE can have multiple prerequisites (PRE_COURSE), but each PRE_COURSE corresponds to one COURSE.
6. **DEGREE_COURSES - DEGREE_PROGRAMS / COURSES:**
    - **Relationship:** Many-to-Many
    - **Connectivity:** There is a many-to-many relationship between DEGREE_COURSES and DEGREE_PROGRAMS/COURSES, indicating that a DEGREE_PROGRAM can have multiple COURSES, and a COURSE can be part of multiple DEGREE_PROGRAMS.
7. **ROOM (Super-Type) - CLASSROOM (Sub-Type) / LAB (Sub-Type):**
    - **Relationship:** Disjoint Subtypes
    - **Connectivity:** ROOM is the super-type, and CLASSROOM and LAB are the subtypes. Each ROOM is either a CLASSROOM or a LAB.

8. **SECTION - SECTION_ENROLLMENT / SECTION_WITHDRAW / SECTION_ATTENDANCE:**
    - **Relationship:** One-to-Many (for each of the three tables)
    - **Connectivity:** Each SECTION can have multiple records in SECTION_ENROLLMENT, SECTION_WITHDRAW, and SECTION_ATTENDANCE, but each record in these tables corresponds to one SECTION.
9. **TIME_TABLE - SECTION / ROOM:**
    - **Relationship:** Many-to-One (for both SECTION and ROOM)
    - **Connectivity:** Each record in TIME_TABLE is associated with one SECTION and one ROOM.
10. **SESSIONAL - SECTION:**
    - **Relationship:** Many-to-One
    - **Connectivity:** Each SESSIONAL record is associated with one SECTION.
11. **SESSIONAL_RESULT - SESSIONAL / STUDENT:**
    - **Relationship:** Many-to-One for both SESSIONAL and STUDENT
    - **Connectivity:** Each SESSIONAL_RESULT record is associated with one SESSIONAL and one STUDENT.
12. **FINAL_RESULT - SESSIONAL_RESULT / STUDENT:**
    - **Relationship:** Many-to-One for both SESSIONAL_RESULT and STUDENT
    - **Connectivity:** Each FINAL_RESULT record is associated with one SESSIONAL_RESULT and one STUDENT.
13. **GRADING - COURSE:**
    - **Relationship:** One-to-Many
    - **Connectivity:** Each COURSE can have multiple GRADING records, but each GRADING record corresponds to one COURSE.
14. **COURSE - SECTION / STUDENT:**
    - **Relationship:** Many-to-One (for both SECTION and STUDENT)

- **Connectivity:** Each COURSE is associated with one SECTION and one STUDENT.

Description of the relations

Faculty

Attributes	Data Type	Size	Constraint
Faculty_ID	Char	3	Primary Key
F_Name	Varchar2	20	not null

Department

Attributes	Data Type	Size	Constraint
Dept_ID	Number	3	Primary Key
Dept_name	Varchar2	20	not null
Faculty_ID	Char	3	Foreign Key

Members

Attributes	Data Type	Size	Constraint
Member_ID	Char	4	Primary key
Member_name	Varchar2	20	not null
Position	Varchar2	12	not null
Dept_ID	Number	3	Foreign key

- **Teaching**

Attributes	Data Type	Size	Constraint
TMem_ID	Char	4	Primary Key
Specialization	Varchar2	15	
Status	Varachar2	8	Not null

- **Non-Teaching**

Attributes	Data Type	Size	Constraint
NMem_ID	Char	4	Primary Key
Work_type	Varchar2	20	Not null

- **Degree_Program**

Attributes	Data Type	Size	Constraint
Degree_ID	Number	3	Primary key
Degree_name	Varchar2	20	Not null
Duration	Number	1	Not null
Degree_Level	Varchar2	20	Not null
Dept_ID	Number	3	Foreign key
TMem_ID	Char	4	Foreign key

- **Student**

Attributes	Data Type	Size	Constraint
RollNo	Char	10	Primary Key
Stu_name	Varchar2	40	Not null
Stu_DOB	Date		Not null
Stu_Gender	Char	1	Can be ‘F’ or ‘M’
Stu_CNIC	Char	14	Unique

Street	Number	3	Not null
State	Varchar2	25	Not null
City	Varchar2	25	Not null
Zip	Number	3	null
Degree_ID	NUMBER	3	FOREIGN KEY
Stu_Batch	Char	3	Not null, start with F or S

- **Student_Contact**

Attributes	Data Type	Size	Constraint
Student_ID	Char	10	Primary key, foreign key
Contact	Number	11	Primary key

- **Courses**

Attributes	Data Type	Size	Constraint
C_code	Char	6	Primary key
C_Title	Varchar2	30	Not null
C_hours	Number	2,1	Can be 0.5,1,2,3

- **Pre_requisites**

Attributes	Data Type	Size	Constraint
P_code	Char	6	Primary Key
C_code	char	6	Primary key,foreign key

- **Degree_Courses**

Attributes	Data Type	Size	Constraint
Degree_ID	number	3	Primary key,foreign key
C_code	char	6	Primary key , foreign key

- **Room**

Attributes	Data Type	Size	Constraint
Room_No	number	2	Primary key
Room_Capacity	Number	3	Not null (greater than 0)
Room_Type	Varchar2	10	Not null (can be lab or classroom)

- **LAB**

Attributes	Data Type	Size	Constraint
Room_No	Number	2	Primary key
Computer_Count	Number	2	Not null (greater then 0)

- **CLASSROOM**

Attributes	Data Type	Size	Constraint
Room_No	Number	2	Primary key
Equiment	Varchar2	20	null

- **Section**

Attributes	Data Type	Size	Constraint
Section_ID	Char	4	Primary key
Room_no	Number	2	Foeign key
Semester	Number	1	Not null
C_code	char	6	Foreign key
TMember_id	Char	4	Foreign key

- **Section_enrollment**

Attributes	Data Type	Size	Constraint
Section_ID	Char	4	Primary key, foreign key
rollNo	Char	10	Primary key, foreign key
Enrollment_date	date		Default sysdate

- **Section_withdraw**

Attributes	Data Type	Size	Constraint
Section_ID	Char	4	Primary key, foreign key
rollNo	Char	10	Primary key, foreign key
withdraw_date	date		null

- **Section_Attendance**

Attributes	Data Type	Size	Constraint
Section_ID	char	4	Primary key,

			Foreign key
rollNo	char	10	Foreign key, primary key
Attendance_date	Date		Default sysdate
status	char	1	Only A or P

- **Time_Table**

Attributes	Data Type	Size	Constraint
Section_ID	Char	4	Primary key
Room_ID	Number	2	Primary key
DaysOfWeek	Varchar2	12	Primary key
StartTime	Date{time stamp}		
EndTime	Date{time stamp}		

- **Sessional**

Attributes	Data Type	Size	Constraint
Sessional_ID	Number	2	Primary key
Section_ID	Char	4	Foreign key
S_Desc	Varchar2	15	Not null
S_marks	Number	2	Greater then or equal to 0

- **Sessional_result**

Attributes	Data Type	Size	Constraint
rollNo	Char	10	Primary key,foreign key
Sessionals_ID	Number	2	Primary

			key,foreign key
Obt_marks	Number	3,2	Greater then or equal to 0

- **Final_result**

Attributes	Data Type	Size	Constraint
Sessional_ID	Number	2	Primary key,foreign key
rollNo	Char	10	Primary key,foreign key
Mid_total	Number	2,1	Greater then or equal to 0
Mid_obtained	Number	2,1	Greater then 0
Final_total	Number	3,2	Greater then 0
Final_obtained	Number	3,2	Greater then or equal to 0

- **Grading**

Attributes	Data Type	Size	Constraint
Lowmarks	Number	3	Primary Key
Himarks	Number	3	Primary Key
Points	Number	2,1	not null
Grade	Char	1	not null

- **Course_result**

Attribute	Data Type	Size	Constraint
C_Code	char	6	Primary key,Foreign key
Section_ID	Char	4	Foreign key
C_marks	Number	4,1	Default 0

Rollno	Char	11	Primary key,Foreign key

TABLES CREATION USING SQL

● Faculties Table

```
CREATE TABLE FACULTIES(
FACULTY_ID CHAR(3) CONSTRAINT FACULTIES_PK PRIMARY KEY,
F_NAME VARCHAR2(20) NOT NULL
);
```

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
FACULTIES	FACULTY_ID	CHAR	3	-	-	1	-	-	-
	F_NAME	VARCHAR2	20	-	-	-	-	-	-

DEPARTMENT TABLE

```
CREATE TABLE DEPARTMENT(
DEPT_ID NUMBER(3) CONSTRAINT DEPARTMENT_PK PRIMARY KEY,
DEPT_NAME VARCHAR2(20) NOT NULL,
FACULTY_ID CHAR(3) CONSTRAINT DEPARTMENT_FK REFERENCES
FACULTIES(FACULTY_ID)
)
```

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
DEPARTMENT	DEPT_ID	NUMBER	-	3	0	1	-	-	-
	DEPT_NAME	VARCHAR2	20	-	-	-	-	-	-
	FACULTY_ID	CHAR	3	-	-	-	✓	-	-

MEMBERS TABLE

```
CREATE TABLE MEMBERS(
MEMBER_ID CHAR(4) CONSTRAINT MEMBERS_PK PRIMARY KEY,
MEMBER_NAME VARCHAR2(20) NOT NULL,
ROLE_TYPE VARCHAR2(12) CHECK (LOWER(ROLE_TYPE) IN ('teaching','non-teaching')),
DEPT_ID NUMBER(3) CONSTRAINT MEMBERS_FK REFERENCES DEPARTMENT(DEPT_ID)
)
```

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
MEMBERS	MEMBER_ID	CHAR	4	-	-	1	-	-	-
	MEMBER_NAME	VARCHAR2	20	-	-	-	-	-	-
	ROLE_TYPE	VARCHAR2	12	-	-	-	✓	-	-
	DEPT_ID	NUMBER	-	3	0	-	✓	-	-

TEACHING (SUBTYPE) TABLE

```
CREATE TABLE TEACHING(
TMEM_ID CHAR(4) PRIMARY KEY,
SPECIALIZATION VARCHAR2(15),
STATUS VARCHAR(8) CHECK (LOWER(STATUS) IN ('regular','visiting')),
CONSTRAINT TEACHING_FK FOREIGN KEY(TMEM_ID) REFERENCES MEMBERS(MEMBER_ID))
```

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
TEACHING	TMEM_ID	CHAR	4	-	-	1	-	-	-
	SPECIALIZATION	VARCHAR2	15	-	-	-	✓	-	-
	STATUS	VARCHAR2	8	-	-	-	✓	-	-

NON_TEACHING (SUBTYPE) TABLE

```
CREATE TABLE NON_TEACHING(
NMEM_ID CHAR(4) PRIMARY KEY,
WORK_TYPE VARCHAR2(20),
CONSTRAINT NON_TEACHING_FK FOREIGN KEY(NMEM_ID) REFERENCES MEMBERS(MEMBER_ID)
)
```

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
NON_TEACHING	NMEM_ID	CHAR	4	-	-	1	-	-	-
	WORK_TYPE	VARCHAR2	20	-	-	-	✓	-	-

DEGREE_PROGRA TABLE

```
CREATE TABLE DEGREE_PROGRAM(
DEGREE_ID NUMBER(3) CONSTRAINT DEGREE_PROGRAN_PK PRIMARY KEY,
DEGREE_NAME VARCHAR2(20) NOT NULL,
DURATION NUMBER(1) NOT NULL,
LEVEL2 VARCHAR2(20) CHECK (LOWER(LEVEL2) IN ('graduate','undergraduate')),
DEPT_ID NUMBER(3) REFERENCES DEPARTMENT(DEPT_ID),
TMEM_ID CHAR(4) REFERENCES TEACHING(TMEM_ID),
```

```
CHECK (DURATION > 0)
)
```

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
DEGREE_PROGRAM	DEGREE_ID	NUMBER	-	3	0	1	-	-	-
	DEGREE_NAME	VARCHAR2	20	-	-	-	-	-	-
	DURATION	NUMBER	-	1	0	-	-	-	-
	LEVEL2	VARCHAR2	20	-	-	-	✓	-	-
	DEPT_ID	NUMBER	-	3	0	-	✓	-	-
	TMEM_ID	CHAR	4	-	-	-	✓	-	-

STUDENT TABLE

```
CREATE TABLE STUDENT(
ROLLNO CHAR(10) CONSTRAINT STUDENT_PK PRIMARY KEY,
STUDENT_NAME VARCHAR2(40) NOT NULL,
STU_DOB DATE,
STU_GENDER CHAR(1) CHECK (LOWER(STU_GENDER) IN ('f','m')),
STU_CNIC CHAR(14) UNIQUE,
STREET NUMBER(3),
STATE VARCHAR2(25) NOT NULL,
CITY VARCHAR2(25) NOT NULL,
ZIP NUMBER(3),
STU_BATCH CHAR(3) CHECK (STU_BATCH LIKE 'f%' OR STU_BATCH LIKE 's%'
OR STU_BATCH LIKE 'F%' OR STU_BATCH LIKE 'S%'),
DEGREE_ID NUMBER(3) REFERENCES DEGREE_PROGRAM(DEGREE_ID))
```

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
STUDENT	ROLLNO	CHAR	10	-	-	1	-	-	-
	STUDENT_NAME	VARCHAR2	40	-	-	-	-	-	-
	STU_DOB	DATE	7	-	-	-	✓	-	-
	STU_GENDER	CHAR	1	-	-	-	✓	-	-
	STU_CNIC	CHAR	14	-	-	-	✓	-	-
	STREET	NUMBER	-	3	0	-	✓	-	-
	STATE	VARCHAR2	25	-	-	-	-	-	-
	CITY	VARCHAR2	25	-	-	-	-	-	-
	ZIP	NUMBER	-	3	0	-	✓	-	-
	STU_BATCH	CHAR	3	-	-	-	✓	-	-
	DEGREE_ID	NUMBER	-	3	0	-	✓	-	-

STUDENT_CONTACT TABLE

```
CREATE TABLE STUDENT_CONTACT(
ROLLNO CHAR(10) REFERENCES STUDENT(ROLLNO),
CONTACT CHAR(11),
PRIMARY KEY(ROLLNO,CONTACT)
)
```

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
STUDENT_CONTACT	ROLLNO	CHAR	10	-	-	1	-	-	-
	CONTACT	CHAR	11	-	-	2	-	-	-

COURSES TABLE

```
CREATE TABLE COURSE(
C_CODE CHAR(6) PRIMARY KEY,
C_TITLE VARCHAR2(30) NOT NULL,
C_HOURS NUMBER(2,1) CHECK (C_HOURS IN (0.5,1,2,3))
);
```

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
COURSE	C_CODE	CHAR	6	-	-	1	-	-	-
	C_TITLE	VARCHAR2	30	-	-	-	-	-	-
	C_HOURS	NUMBER	-	2	1	-	✓	-	-

PRE_REQ

```
CREATE TABLE PRE_REQ(
P_CODE CHAR(6),
C_CODE CHAR(6) REFERENCES COURSE(C_CODE),
PRIMARY KEY(P_CODE,C_CODE)
)
```

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
PRE_REQ	P_CODE	CHAR	6	-	-	1	-	-	-
	C_CODE	CHAR	6	-	-	2	-	-	-

DEGREE_COURSES

```
CREATE TABLE DEGREE_COURSES(
DEGREE_ID NUMBER(3) REFERENCES DEGREE_PROGRAM(DEGREE_ID),
C_CODE CHAR(6) REFERENCES COURSE(C_CODE),
PRIMARY KEY(DEGREE_ID,C_CODE)
)
```

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
DEGREE_COURSES	DEGREE_ID	NUMBER	-	3	0	1	-	-	-
	C_CODE	CHAR	6	-	-	2	-	-	-

ROOM TABLE

```
CREATE TABLE ROOM(
ROOM_NO NUMBER(2) PRIMARY KEY,
ROOM_CAPACITY NUMBER(3) NOT NULL,
ROOM_TYPE VARCHAR2(10) NOT NULL,
CHECK (ROOM_CAPACITY > 0),
CHECK (LOWER(ROOM_TYPE) IN ('lab','classroom'))
)
```

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
ROOM	ROOM_NO	NUMBER	-	2	0	1	-	-	-
	ROOM_CAPACITY	NUMBER	-	3	0	-	-	-	-
	ROOM_TYPE	VARCHAR2	10	-	-	-	-	-	-

CLASSROOM (SUBTYPE) TABLE

```
CREATE TABLE CLASSROOM(
ROOM_NO NUMBER(2) REFERENCES ROOM(ROOM_NO),
EQUIMENT VARCHAR2(20) NOT NULL,
PRIMARY KEY (ROOM_NO)
)
```

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
CLASSROOM	ROOM_NO	NUMBER	-	2	0	1	-	-	-
	EQUIMENT	VARCHAR2	20	-	-	-	-	-	-

LAB (SUBTYPE) TABLE

```
CREATE TABLE LAB(
ROOM_NO NUMBER(2) REFERENCES ROOM(ROOM_NO),
COMPUTER_COUNT NUMBER(3) CHECK (COMPUTER_COUNT >= 0),
PRIMARY KEY (ROOM_NO)
)
```

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
LAB	ROOM_NO	NUMBER	-	2	0	1	-	-	-
	COMPUTER_COUNT	NUMBER	-	3	0	-	✓	-	-

SECTION TABLE

```
CREATE TABLE SECTION(
SECTION_ID CHAR(4) PRIMARY KEY,
```

```
ROOM_NO NUMBER(2) REFERENCES ROOM(ROOM_NO),
SEMISTER NUMBER(1) CHECK (SEMISTER>0),
C_CODE CHAR(6) REFERENCES COURSE(C_CODE),
TMEM_ID CHAR(4) REFERENCES TEACHING(TMEM_ID)
)
```

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
SECTION	SECTION_ID	CHAR	4	-	-	1	-	-	-
	ROOM_NO	NUMBER	-	2	0	-	✓	-	-
	SEMISTER	NUMBER	-	1	0	-	✓	-	-
	C_CODE	CHAR	6	-	-	-	✓	-	-
	TMEM_ID	CHAR	4	-	-	-	✓	-	-

SECTION_ENROLLMENT TABLE

```
CREATE TABLE SECTION_ENROLLMENT(
ROLLNO CHAR(10) REFERENCES STUDENT(ROLLNO),
SECTION_ID CHAR(4) REFERENCES SECTION(SECTION_ID),
ENROLLMENT_DATE DATE DEFAULT SYSDATE,
PRIMARY KEY(ROLLNO,SECTION_ID)
)
```

NOTE: ***FIRST CHECK IS ROLLNO IS ENROLLED***

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
SECTION_ENROLLMENT	ROLLNO	CHAR	10	-	-	1	-	-	-
	SECTION_ID	CHAR	4	-	-	2	-	-	-
	ENROLLMENT_DATE	DATE	7	-	-	-	✓	SYSDATE	-

- **SECTION_WITHDRAW TABLE**

```
CREATE TABLE SECTION_WITHDRAW(
ROLLNO CHAR(10) REFERENCES STUDENT(ROLLNO),
SECTION_ID CHAR(4) REFERENCES SECTION(SECTION_ID),
WITHDRAW_DATE DATE DEFAULT SYSDATE,
PRIMARY KEY(ROLLNO,SECTION_ID));
```

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
SECTION_WITHDRAW	ROLLNO	CHAR	10	-	-	1	-	-	-
	SECTION_ID	CHAR	4	-	-	2	-	-	-
	WITHDRAW_DATE	DATE	7	-	-	-	✓	SYSDATE	-

- **SECTION_ATTENDANE TABLE**

```
CREATE TABLE SECTION_ATTENDANCE(
SECTION_ID CHAR(4),
ROLLNO CHAR(10),
ATT_DATE DATE,
ATT_STATUS CHAR(1) CHECK (ATT_STATUS IN ('P','A','p','a')),
PRIMARY KEY (SECTION_ID,ROLLNO,ATT_DATE),
FOREIGN KEY (SECTION_ID,ROLLNO) REFERENCES
SECTION_ENROLLMENT(SECTION_ID,ROLLNO)
);
```

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
SECTION_ATTENDANCE	SECTION_ID	CHAR	4	-	-	1	-	-	-
	ROLLNO	CHAR	10	-	-	2	-	-	-
	ATT_DATE	DATE	7	-	-	3	-	-	-
	ATT_STATUS	CHAR	1	-	-	-	✓	-	-

- **TIME_TABLE TABLE**

```
CREATE TABLE TIME_TABLE(
SECTION_ID CHAR(4) REFERENCES SECTION(SECTION_ID),
ROOM_NO NUMBER(2) REFERENCES ROOM(ROOM_NO),
DAYS_OF_WEEK VARCHAR2(12),
START_TIME TIMESTAMP NOT NULL,
END_TIME TIMESTAMP NOT NULL,
PRIMARY KEY(SECTION_ID,ROOM_NO,DAYS_OF_WEEK)
)
```

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
TIME_TABLE	SECTION_ID	CHAR	4	-	-	1	-	-	-
	ROOM_NO	NUMBER	-	2	0	2	-	-	-
	DAYS_OF_WEEK	VARCHAR2	12	-	-	3	-	-	-
	START_TIME	TIMESTAMP(6)	11	-	6	-	-	-	-
	END_TIME	TIMESTAMP(6)	11	-	6	-	-	-	-

- **SESSIONAL TABLE**

```
CREATE TABLE SESSIONAL(
SESSIONAL_ID NUMBER(2) PRIMARY KEY,
SECTION_ID CHAR(4) REFERENCES SECTION(SECTION_ID),
S_DESC VARCHAR2(15) NOT NULL,
S_MARKS NUMBER(2) CHECK (S_MARKS >= 0),
UNIQUE(S_DESC,SECTION_ID)
```

```
);
```

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
SESSIONAL	SESSIONAL_ID	NUMBER	-	2	0	1	-	-	-
	SECTION_ID	CHAR	4	-	-	-	✓	-	-
	S_DESC	VARCHAR2	15	-	-	-	-	-	-
	S_MARKS	NUMBER	-	2	0	-	✓	-	-

- ***SESSIONAL_RESULT TABLE***

```
CREATE TABLE SESSIONAL_RESULT(
ROLLNO CHAR(10) REFERENCES STUDENT(ROLLNO),
SESSIONAL_ID NUMBER(2) REFERENCES SESSIONAL(SESSIONAL_ID),
OBT_MARKS NUMBER(3,1) CHECK (OBT_MARKS >= 0),
PRIMARY KEY(ROLLNO,SESSIONAL_ID)
)
```

TRIGGER NEEDED

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
SESSIONAL_RESULT	ROLLNO	CHAR	10	-	-	1	-	-	-
	SESSIONAL_ID	NUMBER	-	2	0	2	-	-	-
	OBT_MARKS	NUMBER	-	3	1	-	✓	-	-

- **FINAL_RESULT**

```
CREATE TABLE FINAL_RESULT(
SECTION_ID CHAR(4),
ROLLNO CHAR(10),
MID_TOTAL NUMBER(3,1) CHECK (MID_TOTAL >=0),
MID_OBT NUMBER(3,1) CHECK (MID_OBT >=0),
```

```
FINAL_TOTAL NUMBER(3,1) CHECK (FINAL_TOTAL >=0),
FINAL_OBT NUMBER(3,1) CHECK (FINAL_OBT >=0),
PRIMARY KEY(SECTION_ID,ROLLNO),
FOREIGN KEY(SECTION_ID,ROLLNO) REFERENCES
SECTION_ENROLLMENT(SECTION_ID,ROLLNO))
```

TRIGGER NEEDED

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
FINAL_RESULT	SECTION_ID	CHAR	4	-	-	1	-	-	-
	ROLLNO	CHAR	10	-	-	2	-	-	-
	MID_TOTAL	NUMBER	-	3	1	-	✓	-	-
	MID_OBT	NUMBER	-	3	1	-	✓	-	-
	FINAL_TOTAL	NUMBER	-	3	1	-	✓	-	-
	FINAL_OBT	NUMBER	-	3	1	-	✓	-	-

- **GRADING**

```
CREATE TABLE GRADING(
LOWMARKS NUMBER(3),
HIMARKS NUMBER(3),
GRADE_POINTS NUMBER(2,1) NOT NULL,
GRADE CHAR(2) NOT NULL,
PRIMARY KEY(LOWMARKS,HIMARKS)
)
```

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
GRADING	LOWMARKS	NUMBER	-	3	0	1	-	-	-
	HIMARKS	NUMBER	-	3	0	2	-	-	-
	GRADE_POINTS	NUMBER	-	2	1	-	-	-	-
	GRADE	CHAR	2	-	-	-	-	-	-

- **COURSE_RESULT**

```
CREATE TABLE COURSE_RESULT(
C_CODE CHAR(6) REFERENCES COURSE(C_CODE),
SECTION_ID CHAR(4),
C_MARKS NUMBER(4,1) DEFAULT 0,
ROLLNO CHAR(10),
PRIMARY KEY(ROLLNO,C_CODE),
FOREIGN KEY(ROLLNO,SECTION_ID) REFERENCES
SECTION_ENROLLMENT(ROLLNO,SECTION_ID)
);
```

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
COURSE_RESULT	C_CODE	CHAR	6	-	-	2	-	-	-
	SECTION_ID	CHAR	4	-	-	-	✓	-	-
	C_MARKS	NUMBER	-	4	1	-	✓	0	-
	ROLLNO	CHAR	10	-	-	1	-	-	-

Dummy Data Entry:

- **FACULTIES TABLE**

```
INSERT INTO FACULTIES VALUES('F01','FCIT');
```

FACULTY_ID	F_NAME
F01	FCIT

- **DEPARTMENT TABLE**

```
INSERT INTO DEPARTMENT VALUES(1,'IT','F01');
INSERT INTO DEPARTMENT VALUES(2,'CS','F01');
INSERT INTO DEPARTMENT VALUES(3,'SE','F01');
```

DEPT_ID	DEPT_NAME	FACULTY_ID
1	IT	F01
2	CS	F01
3	SE	F01

- **MEMBER TABLE**

```
INSERT INTO MEMBERS VALUES('M001','SYED WAQAR','TEACHING',1);
INSERT INTO MEMBERS VALUES('M002','ASIF SOHAIL','TEACHING',1);
INSERT INTO MEMBERS VALUES('M003','KASHIF MURTAZA','TEACHING',1);
INSERT INTO MEMBERS VALUES('M004','ANZAR','TEACHING',1);
INSERT INTO MEMBERS VALUES('M005','MAM ZARA','TEACHING',1);
```

```
INSERT INTO MEMBERS VALUES('M006','KHAWAR BUTT','TEACHING',1);
INSERT INTO MEMBERS VALUES('M007','ALTAF HUSSAIN','NON-TEACHING',1);
INSERT INTO MEMBERS VALUES('M008','HAQ NAWAZ','NON-TEACHING',1);
```

MEMBER_ID	MEMBER_NAME	ROLE_TYPE	DEPT_ID
M001	SYED WAQAR	TEACHING	1
M002	ASIF SOHAIL	TEACHING	1
M003	KASHIF MURTAZA	TEACHING	1
M004	ANZAR	TEACHING	1
M005	MAM ZARA	TEACHING	1
M006	KHAWAR BUTT	TEACHING	1
M007	ALTAF HUSSAIN	NON-TEACHING	1
M008	HAQ NAWAZ	NON-TEACHING	1

- **TEACHING TABLE**

```
INSERT INTO TEACHING VALUES('M002','DATABASE EXPERT','REGULAR')
INSERT INTO TEACHING VALUES('M001','IT EXPERT','REGULAR')
INSERT INTO TEACHING VALUES('M003','LINEAR EXPERT','REGULAR')
INSERT INTO TEACHING VALUES('M004','SOFTWARE EXPERT','REGULAR')
INSERT INTO TEACHING VALUES('M005','MODELING EXPERT','VISITING')
INSERT INTO TEACHING VALUES('M006','ISLAMIC','VISITING')
```

TMEM_ID	SPECIALIZATION	STATUS
M002	DATABASE EXPERT	REGULAR
M001	IT EXPERT	REGULAR
M005	MODELING EXPERT	VISITING
M006	ISLAMIC	VISITING
M003	LINEAR EXPERT	REGULAR
M004	SOFTWARE EXPERT	REGULAR

- **NON_TEACHING TABLE**

```
INSERT INTO NON_TEACHING VALUES('M007','GUARD')
INSERT INTO NON_TEACHING VALUES('M008','SWEEPER')
```

NMEM_ID	WORK_TYPE
M007	GUARD
M008	SWEEPER

- **DEGREE_PROGRAM TABLE**

```
INSERT INTO DEGREE_PROGRAM VALUES(1,'BSIT',4,'GRADUATE',1,'M003');
INSERT INTO DEGREE_PROGRAM VALUES(2,'BSCS',4,'GRADUATE',2,'M001');
INSERT INTO DEGREE_PROGRAM VALUES(3,'BSSE',4,'GRADUATE',3,'M002');
```

DEGREE_ID	DEGREE_NAME	DURATION	LEVEL	DEPT_ID	TMEM_ID
1	BSIT	4	GRADUATE	1	M003
2	BSCS	4	GRADUATE	2	M001
3	BSSE	4	GRADUATE	3	M002

- **STUDENT TABLE**

```
INSERT INTO STUDENT VALUES('BITF21M022','AQDAS SHABBIR',TO_DATE('22-MARCH-2002','DD-MON-YYYY'),'M','3520209750644','F21',1,7,'LAHORE','LAHORE',200)

INSERT INTO STUDENT VALUES('BITF21M023','MUHAMMAD HASEEB',TO_DATE('10-FEB-2002','DD-MON-YYYY'),'M','3520209750642','F21',1,8,'NAROWAL','NAROWAL',201)

INSERT INTO STUDENT VALUES('BITF21M032','SYED JAFFAR ALI NAQVI',TO_DATE('10-JAN-2001','DD-MON-YYYY'),'M','3520209750623','F21',1,9,'GUJRANWALA','GUJRANWALA',202)

INSERT INTO STUDENT VALUES('BITF21M020','HUZAIFA RAUF',TO_DATE('10-JAN-2000','DD-MON-YYYY'),'M','3520209750433','F21',1,10,,'RAHEEM YAR KHAN','RAHEEM YAR KHAN',203)

INSERT INTO STUDENT VALUES('BITF21M010','UMER',TO_DATE('1-APR-2002','DD-MON-YYYY'),'M','3520232750433','F21',1,11,'SAHIWAL' ,'SAHIWAL',204)

INSERT INTO STUDENT VALUES('BITF21M009','WARDA',TO_DATE('1-JUN-2001','DD-MON-YYYY'),'F','3520232752333','SAHIWAL','F21',12,'SAHIWAL' ,'SAHIWAL',205)

INSERT INTO STUDENT VALUES('BITF21M003','AYESHA',TO_DATE('1-MAY-2001','DD-MON-YYYY'),'F','3520234352333','SAHIWAL','F21',1,13,'SAHIWAL' ,'SAHIWAL',205)
```

ROLLNO	STUDENT_NAME	STU_DOB	STU_GENDER	STU_CNIC	STREET	STATE	CITY	ZIP	STU_BATCH	DEGREE_ID
BITF21M022	AQDAS SHABBIR	03/22/2002	M	3520209750644	7	LAHORE	LAHORE	200	F21	1
BITF21M023	MUHAMMAD HASEEB	02/10/2002	M	3520209750642	8	NAROWAL	NAROWAL	201	F21	1
BITF21M032	SYED JAFFAR ALI NAQVI	01/10/2001	M	3520209750623	9	GUJRANWALA	GUJRANWALA	202	F21	1
BITF21M020	HUZAIFA RAUF	01/10/2000	M	3520209750433	10	RAHEEM YAR KHAN	RAHEEM YAR KHAN	203	F21	1
BITF21M010	UMER	04/01/2002	M	3520232750433	11	SAHIWAL	SAHIWAL	204	F21	1
BITF21M009	WARDA	06/01/2001	F	3520232752333	12	SAHIWAL	SAHIWAL	205	F21	1
BITF21M003	AYESHA	05/01/2001	F	3520234352333	13	SAHIWAL	SAHIWAL	206	F21	1

- **STUDENT_CONTACT**

```
INSERT INTO STUDENT_CONTACT VALUES('BITF21M022','03008827297')
INSERT INTO STUDENT_CONTAC
T VALUES('BITF21M022','03244053164')
INSERT INTO STUDENT_CONTACT VALUES('BITF21M023','03081414005')
INSERT INTO STUDENT_CONTACT VALUES('BITF21M020','03041414005')
INSERT INTO STUDENT_CONTACT VALUES('BITF21M032','03431414005')
INSERT INTO STUDENT_CONTACT VALUES('BITF21M009','03331414005')
INSERT INTO STUDENT_CONTACT VALUES('BITF21M003','03231414005')
```

ROLLNO	CONTACT
BITF21M003	03231414005
BITF21M009	03331414005
BITF21M020	03041414005
BITF21M022	03008827297
BITF21M022	03244053164
BITF21M023	03081414005
BITF21M032	03431414005

- **COURSE TABLE**

```
INSERT INTO COURSE VALUES('CC-213','DATA STRUCTURE AND
ALGORITHM',3);
INSERT INTO COURSE VALUES('CC-215','DATABASE SYSTEMS',3);
INSERT INTO COURSE VALUES('CC-212','SOFTWARE ENGINEERING',3);
```

```
INSERT INTO COURSE VALUES('SI-21X','MODELING AND SIMULATION',3);
INSERT INTO COURSE VALUES('MS-252','LINEAR ALGEBRA',3);
INSERT INTO COURSE VALUES('HQ-282','QURAN TRANSLATION',0.5);
INSERT INTO COURSE VALUES('CC-112','PROGRAMMING FUNDAMENTALS',3);
INSERT INTO COURSE VALUES('CC-211','OBJECT ORIENTED PROGRAMMIN',3);
```

C_CODE	C_TITLE	C_HOURS
HQ-282	QURAN TRANSLATION	.5
CC-112	PROGRAMMING FUNDAMENTALS	3
CC-211	OBJECT ORIENTED PROGRAMMIN	3
CC-213	DATA STRUCTURE AND ALGORITHM	3
CC-215	DATABASE SYSTEMS	3
CC-212	SOFTWARE ENGINEERING	3
SI-21X	MODELING AND SIMULATION	3
MS-252	LINEAR ALGEBRA	3

- **PRE-REQ TABLE**

```
INSERT INTO PRE_REQ VALUES('CC-112','CC-211');
INSERT INTO PRE_REQ VALUES('CC-211','CC-213');
```

P_CODE	C_CODE
CC-112	CC-211
CC-211	CC-213

- **DEGREE_COURSES**

```
INSERT INTO DEGREE_COURSES VALUES(1,'HQ-282');
```

```
INSERT INTO DEGREE_COURSES VALUES(1,'CC-112');
INSERT INTO DEGREE_COURSES VALUES(1,'CC-211');
INSERT INTO DEGREE_COURSES VALUES(1,'CC-213');
INSERT INTO DEGREE_COURSES VALUES(1,'CC-215');
INSERT INTO DEGREE_COURSES VALUES(1,'CC-212');
INSERT INTO DEGREE_COURSES VALUES(1,'SI-21X');
INSERT INTO DEGREE_COURSES VALUES(1,'MS-252');
INSERT INTO DEGREE_COURSES VALUES(2,'MS-252');
```

DEGREE_ID	C_CODE
1	CC-112
1	CC-211
1	CC-212
1	CC-213
1	CC-215
1	HQ-282
1	MS-252
1	SI-21X
2	MS-252

- **ROOM TABLE**

```
INSERT INTO ROOM VALUES(1,50,'CLASSROOM');
INSERT INTO ROOM VALUES(2,50,'CLASSROOM');
INSERT INTO ROOM VALUES(3,50,'CLASSROOM');
```

```
INSERT INTO ROOM VALUES(4,50,'CLASSROOM');
INSERT INTO ROOM VALUES(5,50,'CLASSROOM');
INSERT INTO ROOM VALUES(6,50,'CLASSROOM');
INSERT INTO ROOM VALUES(7,50,'CLASSROOM');
INSERT INTO ROOM VALUES(8,50,'LAB');
INSERT INTO ROOM VALUES(9,50,'LAB');
INSERT INTO ROOM VALUES(10,50,'LAB');
```

ROOM_NO	ROOM_CAPACITY	ROOM_TYPE
1	50	CLASSROOM
2	50	CLASSROOM
3	50	CLASSROOM
4	50	CLASSROOM
5	50	CLASSROOM
6	50	CLASSROOM
7	50	CLASSROOM
8	50	LAB
9	50	LAB
10	50	LAB

- **LAB TABLE**

```
INSERT INTO LAB VALUES(8,50);
INSERT INTO LAB VALUES(9,60);
INSERT INTO LAB VALUES(10,66);
```

ROOM_NO	COMPUTER_COUNT
8	50
9	60
10	66

- **CLASSROOM TABLE**

```
INSERT INTO CLASSROOM VALUES(1,'LED');
INSERT INTO CLASSROOM VALUES(2,'LED');
INSERT INTO CLASSROOM VALUES(3,'LED');
INSERT INTO CLASSROOM VALUES(4,'LED');
INSERT INTO CLASSROOM VALUES(5,'PROJECTOR');
INSERT INTO CLASSROOM VALUES(6,'PROJECTOR');
INSERT INTO CLASSROOM VALUES(7,'PROJECTOR');
```

ROOM_NO	EQUIMENT
1	LED
2	LED
3	LED
4	LED
5	PROJECTOR
6	PROJECTOR
7	PROJECTOR

- **SECTION TABLE**

```
INSERT INTO SECTION VALUES('S001',1,1,'CC-213','M001');
INSERT INTO SECTION VALUES('S002',1,1,'CC-215','M002');
```

```
INSERT INTO SECTION VALUES('S003',1,1,'CC-212','M004');
INSERT INTO SECTION VALUES('S004',1,1,'MS-252','M003');
INSERT INTO SECTION VALUES('S005',1,1,'SI-21X','M005');
INSERT INTO SECTION VALUES('S006',1,1,'HQ-282','M006');
```

SECTION_ID	ROOM_NO	SEMISTER	C_CODE	TMEM_ID
S001	1	1	CC-213	M001
S002	1	1	CC-215	M002
S003	1	1	CC-212	M004
S004	1	1	MS-252	M003
S005	1	1	SI-21X	M005
S006	1	1	HQ-282	M006

- **SECTION_ENROLLMENT TABLE**

```
INSERT INTO SECTION_ENROLLMENT VALUES('BITF21M022','S001',SYSDATE);
INSERT INTO SECTION_ENROLLMENT VALUES('BITF21M022','S002',SYSDATE);
INSERT INTO SECTION_ENROLLMENT VALUES('BITF21M022','S003',SYSDATE);
INSERT INTO SECTION_ENROLLMENT VALUES('BITF21M022','S004',SYSDATE);
INSERT INTO SECTION_ENROLLMENT VALUES('BITF21M022','S005',SYSDATE);
INSERT INTO SECTION_ENROLLMENT VALUES('BITF21M022','S006',SYSDATE);
```

```
INSERT INTO SECTION_ENROLLMENT VALUES('BITF21M023','S001',SYSDATE);
INSERT INTO SECTION_ENROLLMENT VALUES('BITF21M023','S002',SYSDATE);
INSERT INTO SECTION_ENROLLMENT VALUES('BITF21M023','S003',SYSDATE);
INSERT INTO SECTION_ENROLLMENT VALUES('BITF21M023','S004',SYSDATE);
```

```
INSERT INTO SECTION_ENROLLMENT VALUES('BITF21M023','S005',SYSDATE);
INSERT INTO SECTION_ENROLLMENT VALUES('BITF21M023','S006',SYSDATE);


INSERT INTO SECTION_ENROLLMENT VALUES('BITF21M010','S001',SYSDATE);
INSERT INTO SECTION_ENROLLMENT VALUES('BITF21M010','S002',SYSDATE);
INSERT INTO SECTION_ENROLLMENT VALUES('BITF21M010','S003',SYSDATE);
INSERT INTO SECTION_ENROLLMENT VALUES('BITF21M010','S004',SYSDATE);
INSERT INTO SECTION_ENROLLMENT VALUES('BITF21M010','S005',SYSDATE);
INSERT INTO SECTION_ENROLLMENT VALUES('BITF21M010','S006',SYSDATE);
```

```
INSERT INTO SECTION_ENROLLMENT VALUES('BITF21M020','S001',SYSDATE);
INSERT INTO SECTION_ENROLLMENT VALUES('BITF21M020','S002',SYSDATE);
INSERT INTO SECTION_ENROLLMENT VALUES('BITF21M020','S003',SYSDATE);
INSERT INTO SECTION_ENROLLMENT VALUES('BITF21M020','S004',SYSDATE);
INSERT INTO SECTION_ENROLLMENT VALUES('BITF21M020','S005',SYSDATE);
INSERT INTO SECTION_ENROLLMENT VALUES('BITF21M020','S006',SYSDATE);


INSERT INTO SECTION_ENROLLMENT VALUES('BITF21M032','S001',SYSDATE);
INSERT INTO SECTION_ENROLLMENT VALUES('BITF21M032','S002',SYSDATE);
INSERT INTO SECTION_ENROLLMENT VALUES('BITF21M032','S003',SYSDATE);
INSERT INTO SECTION_ENROLLMENT VALUES('BITF21M032','S004',SYSDATE);
INSERT INTO SECTION_ENROLLMENT VALUES('BITF21M032','S005',SYSDATE);
INSERT INTO SECTION_ENROLLMENT VALUES('BITF21M032','S006',SYSDATE);
```

```
INSERT INTO SECTION_ENROLLMENT VALUES('BITF21M009','S001',SYSDATE);
INSERT INTO SECTION_ENROLLMENT VALUES('BITF21M009','S002',SYSDATE);
INSERT INTO SECTION_ENROLLMENT VALUES('BITF21M009','S003',SYSDATE);
INSERT INTO SECTION_ENROLLMENT VALUES('BITF21M009','S004',SYSDATE);
INSERT INTO SECTION_ENROLLMENT VALUES('BITF21M009','S005',SYSDATE);
INSERT INTO SECTION_ENROLLMENT VALUES('BITF21M009','S006',SYSDATE);

INSERT INTO SECTION_ENROLLMENT VALUES('BITF21M003','S001',SYSDATE);
INSERT INTO SECTION_ENROLLMENT VALUES('BITF21M003','S002',SYSDATE);
INSERT INTO SECTION_ENROLLMENT VALUES('BITF21M003','S003',SYSDATE);
INSERT INTO SECTION_ENROLLMENT VALUES('BITF21M003','S004',SYSDATE);
INSERT INTO SECTION_ENROLLMENT VALUES('BITF21M003','S005',SYSDATE);
INSERT INTO SECTION_ENROLLMENT
VALUES('BITF21M003','S006',SYS0906DATE);
```

ROLLNO	SECTION_ID	ENROLLMENT_DATE
BITF21M022	S001	12/16/2023
BITF21M022	S002	12/16/2023
BITF21M022	S003	12/16/2023
BITF21M022	S004	12/16/2023
BITF21M022	S005	12/16/2023
BITF21M022	S006	12/16/2023
BITF21M023	S001	12/16/2023
BITF21M023	S002	12/16/2023
BITF21M023	S003	12/16/2023
BITF21M023	S004	12/16/2023
BITF21M023	S005	12/16/2023
BITF21M023	S006	12/16/2023
BITF21M010	S001	12/16/2023
BITF21M010	S002	12/16/2023
BITF21M010	S003	12/16/2023
BITF21M010	S004	12/16/2023
BITF21M010	S005	12/16/2023
BITF21M010	S006	12/16/2023
BITF21M020	S001	12/16/2023
BITF21M020	S002	12/16/2023
BITF21M020	S003	12/16/2023

- **SECTION_ATTENDANCE TABLE**

```
INSERT INTO SECTION_ATTENDANCE
VALUES('S001','BITF21M022',TO_DATE('19-DEC-2023','DD-MON-YYYY'),'P');

INSERT INTO SECTION_ATTENDANCE
VALUES('S002','BITF21M022',TO_DATE('19-DEC-2023','DD-MON-YYYY'),'P');
```

```
INSERT INTO SECTION_ATTENDANCE
VALUES('S003','BITF21M022',TO_DATE('19-DEC-2023','DD-MON-YYYY'),'P');

INSERT INTO SECTION_ATTENDANCE
VALUES('S004','BITF21M022',TO_DATE('19-DEC-2023','DD-MON-YYYY'),'A');

INSERT INTO SECTION_ATTENDANCE
VALUES('S005','BITF21M022',TO_DATE('19-DEC-2023','DD-MON-YYYY'),'P');

INSERT INTO SECTION_ATTENDANCE
VALUES('S006','BITF21M022',TO_DATE('19-DEC-2023','DD-MON-YYYY'),'P');



INSERT INTO SECTION_ATTENDANCE
VALUES('S001','BITF21M023',TO_DATE('19-DEC-2023','DD-MON-YYYY'),'P');

INSERT INTO SECTION_ATTENDANCE
VALUES('S002','BITF21M023',TO_DATE('19-DEC-2023','DD-MON-YYYY'),'P');

INSERT INTO SECTION_ATTENDANCE
VALUES('S003','BITF21M023',TO_DATE('19-DEC-2023','DD-MON-YYYY'),'P');

INSERT INTO SECTION_ATTENDANCE
VALUES('S004','BITF21M023',TO_DATE('19-DEC-2023','DD-MON-YYYY'),'A');

INSERT INTO SECTION_ATTENDANCE
VALUES('S005','BITF21M023',TO_DATE('19-DEC-2023','DD-MON-YYYY'),'P');

INSERT INTO SECTION_ATTENDANCE
VALUES('S006','BITF21M023',TO_DATE('19-DEC-2023','DD-MON-YYYY'),'A');


INSERT INTO SECTION_ATTENDANCE
VALUES('S001','BITF21M010',TO_DATE('19-DEC-2023','DD-MON-YYYY'),'P');
```

```
INSERT INTO SECTION_ATTENDANCE
VALUES('S002','BITF21M010',TO_DATE('19-DEC-2023','DD-MON-YYYY'),'P');

INSERT INTO SECTION_ATTENDANCE
VALUES('S003','BITF21M010',TO_DATE('19-DEC-2023','DD-MON-YYYY'),'P');

INSERT INTO SECTION_ATTENDANCE
VALUES('S004','BITF21M010',TO_DATE('19-DEC-2023','DD-MON-YYYY'),'P');

INSERT INTO SECTION_ATTENDANCE
VALUES('S005','BITF21M010',TO_DATE('19-DEC-2023','DD-MON-YYYY'),'P');

INSERT INTO SECTION_ATTENDANCE
VALUES('S006','BITF21M010',TO_DATE('19-DEC-2023','DD-MON-YYYY'),'P');



INSERT INTO SECTION_ATTENDANCE
VALUES('S001','BITF21M020',TO_DATE('19-DEC-2023','DD-MON-YYYY'),'P');

INSERT INTO SECTION_ATTENDANCE
VALUES('S002','BITF21M020',TO_DATE('19-DEC-2023','DD-MON-YYYY'),'P');

INSERT INTO SECTION_ATTENDANCE
VALUES('S003','BITF21M020',TO_DATE('19-DEC-2023','DD-MON-YYYY'),'P');

INSERT INTO SECTION_ATTENDANCE
VALUES('S004','BITF21M020',TO_DATE('19-DEC-2023','DD-MON-YYYY'),'A');

INSERT INTO SECTION_ATTENDANCE
VALUES('S005','BITF21M020',TO_DATE('19-DEC-2023','DD-MON-YYYY'),'P');

INSERT INTO SECTION_ATTENDANCE
VALUES('S006','BITF21M020',TO_DATE('19-DEC-2023','DD-MON-YYYY'),'A');
```

```
INSERT INTO SECTION_ATTENDANCE
VALUES('S001','BITF21M009',TO_DATE('19-DEC-2023','DD-MON-YYYY'),'P');

INSERT INTO SECTION_ATTENDANCE
VALUES('S002','BITF21M009',TO_DATE('19-DEC-2023','DD-MON-YYYY'),'P');

INSERT INTO SECTION_ATTENDANCE
VALUES('S003','BITF21M009',TO_DATE('19-DEC-2023','DD-MON-YYYY'),'P');

INSERT INTO SECTION_ATTENDANCE
VALUES('S004','BITF21M009',TO_DATE('19-DEC-2023','DD-MON-YYYY'),'A');

INSERT INTO SECTION_ATTENDANCE
VALUES('S005','BITF21M009',TO_DATE('19-DEC-2023','DD-MON-YYYY'),'P');

INSERT INTO SECTION_ATTENDANCE
VALUES('S006','BITF21M009',TO_DATE('19-DEC-2023','DD-MON-YYYY'),'A');


INSERT INTO SECTION_ATTENDANCE
VALUES('S001','BITF21M003',TO_DATE('19-DEC-2023','DD-MON-YYYY'),'P');

INSERT INTO SECTION_ATTENDANCE
VALUES('S002','BITF21M003',TO_DATE('19-DEC-2023','DD-MON-YYYY'),'P');

INSERT INTO SECTION_ATTENDANCE
VALUES('S003','BITF21M003',TO_DATE('19-DEC-2023','DD-MON-YYYY'),'P');

INSERT INTO SECTION_ATTENDANCE
VALUES('S004','BITF21M003',TO_DATE('19-DEC-2023','DD-MON-YYYY'),'A');

INSERT INTO SECTION_ATTENDANCE
VALUES('S005','BITF21M003',TO_DATE('19-DEC-2023','DD-MON-YYYY'),'P');

INSERT INTO SECTION_ATTENDANCE
VALUES('S006','BITF21M003',TO_DATE('19-DEC-2023','DD-MON-YYYY'),'A');
```

```
INSERT INTO SECTION_ATTENDANCE
VALUES('S001','BITF21M032',TO_DATE('19-DEC-2023','DD-MON-YYYY'),'P');

INSERT INTO SECTION_ATTENDANCE
VALUES('S002','BITF21M032',TO_DATE('19-DEC-2023','DD-MON-YYYY'),'P');

INSERT INTO SECTION_ATTENDANCE
VALUES('S003','BITF21M032',TO_DATE('19-DEC-2023','DD-MON-YYYY'),'P');

INSERT INTO SECTION_ATTENDANCE
VALUES('S004','BITF21M032',TO_DATE('19-DEC-2023','DD-MON-YYYY'),'A');

INSERT INTO SECTION_ATTENDANCE
VALUES('S005','BITF21M032',TO_DATE('19-DEC-2023','DD-MON-YYYY'),'P');

INSERT INTO SECTION_ATTENDANCE
VALUES('S006','BITF21M032',TO_DATE('19-DEC-2023','DD-MON-YYYY'),'A');
```

SECTION_ID	ROLLNO	ATT_DATE	ATT_STATUS
S001	BITF21M022	12/19/2023	P
S002	BITF21M022	12/19/2023	P
S003	BITF21M022	12/19/2023	P
S004	BITF21M022	12/19/2023	A
S005	BITF21M022	12/19/2023	P
S006	BITF21M022	12/19/2023	P
S001	BITF21M023	12/19/2023	P
S002	BITF21M023	12/19/2023	P
S003	BITF21M023	12/19/2023	P
S004	BITF21M023	12/19/2023	A
S005	BITF21M023	12/19/2023	P
S006	BITF21M023	12/19/2023	A
S001	BITF21M010	12/19/2023	P
S002	BITF21M010	12/19/2023	P
S003	BITF21M010	12/19/2023	P
S004	BITF21M010	12/19/2023	P
S005	BITF21M010	12/19/2023	P
S006	BITF21M010	12/19/2023	P

- **SESSIONAL TABLE**

```
INSERT INTO SESSIONAL VALUES(1,'S001','QUIZ',10)
INSERT INTO SESSIONAL VALUES(2,'S001','ASSIGNMNET',10)
INSERT INTO SESSIONAL VALUES(3,'S001','PROJECT',5)

INSERT INTO SESSIONAL VALUES(4,'S002','QUIZ',10);
INSERT INTO SESSIONAL VALUES(5,'S002','ASSIGNMNET',8);
INSERT INTO SESSIONAL VALUES(6,'S002','PROJECT',5);
```

```
INSERT INTO SESSIONAL VALUES(7,'S002','PRESENTATION',2);


INSERT INTO SESSIONAL VALUES(8,'S003','QUIZ',10);
INSERT INTO SESSIONAL VALUES(9,'S003','ASSIGNMNET',10);
INSERT INTO SESSIONAL VALUES(10,'S003','PROJECT',5);


INSERT INTO SESSIONAL VALUES(11,'S004','QUIZ',15);
INSERT INTO SESSIONAL VALUES(12,'S004','ASSIGNMNET',10);


INSERT INTO SESSIONAL VALUES(13,'S005','QUIZ',15);
INSERT INTO SESSIONAL VALUES(14,'S005','ASSIGNMNET',10);
INSERT INTO SESSIONAL VALUES(15,'S006','ASSIGNMNET',25);
```

SESSIONAL_ID	SECTION_ID	S_DESC	S_MARKS
4	S002	QUIZ	10
5	S002	ASSIGNMNET	8
6	S002	PROJECT	5
7	S002	PRESENTATION	2
8	S003	QUIZ	10
9	S003	ASSIGNMNET	10
10	S003	PROJECT	5
11	S004	QUIZ	15
12	S004	ASSIGNMNET	10
13	S005	QUIZ	15
14	S005	ASSIGNMNET	10
15	S006	ASSIGNMNET	25
1	S001	QUIZ	10
2	S001	ASSIGNMNET	10
3	S001	PROJECT	5

- **SESSIONAL_RESULT TABLE**

```
INSERT INTO SESSIONAL_RESULT VALUES('BITF21M022',1,9);
INSERT INTO SESSIONAL_RESULT VALUES('BITF21M022',2,9);
INSERT INTO SESSIONAL_RESULT VALUES('BITF21M022',3,4);
INSERT INTO SESSIONAL_RESULT VALUES('BITF21M023',1,10);
INSERT INTO SESSIONAL_RESULT VALUES('BITF21M023',2,5);
INSERT INTO SESSIONAL_RESULT VALUES('BITF21M023',3,4);
INSERT INTO SESSIONAL_RESULT VALUES('BITF21M020',1,9);
INSERT INTO SESSIONAL_RESULT VALUES('BITF21M020',2,4);
INSERT INTO SESSIONAL_RESULT VALUES('BITF21M020',3,4);
```

```
INSERT INTO SESSIONAL_RESULT VALUES('BITF21M010',1,6);
INSERT INTO SESSIONAL_RESULT VALUES('BITF21M010',2,9);
INSERT INTO SESSIONAL_RESULT VALUES('BITF21M010',3,4);
INSERT INTO SESSIONAL_RESULT VALUES('BITF21M032',1,9);
INSERT INTO SESSIONAL_RESULT VALUES('BITF21M032',2,4);
INSERT INTO SESSIONAL_RESULT VALUES('BITF21M032',3,4);
INSERT INTO SESSIONAL_RESULT VALUES('BITF21M009',1,9);
INSERT INTO SESSIONAL_RESULT VALUES('BITF21M009',2,5);
INSERT INTO SESSIONAL_RESULT VALUES('BITF21M009',3,4);
INSERT INTO SESSIONAL_RESULT VALUES('BITF21M003',1,9);
INSERT INTO SESSIONAL_RESULT VALUES('BITF21M003',2,9);
INSERT INTO SESSIONAL_RESULT VALUES('BITF21M003',3,5);
```

ROLLNO	SESSIONAL_ID	OBT_MARKS
BITF21M022	4	9
BITF21M022	5	3
BITF21M022	6	4
BITF21M022	7	2
BITF21M023	4	10
BITF21M023	5	5
BITF21M023	6	4
BITF21M020	4	9
BITF21M020	5	4
BITF21M020	6	4
BITF21M010	4	6
BITF21M010	6	4
BITF21M032	4	9
BITF21M032	5	4

- **FINAL_RESULT TABLE**

```
INSERT INTO FINAL_RESULT VALUES('S001','BITF21M022',35,3,40,38);
INSERT INTO FINAL_RESULT VALUES('S002','BITF21M022',35,33,40,35);
INSERT INTO FINAL_RESULT VALUES('S003','BITF21M022',35,30,40,35);
INSERT INTO FINAL_RESULT VALUES('S004','BITF21M022',35,34,40,35);
INSERT INTO FINAL_RESULT VALUES('S005','BITF21M022',35,30,40,39);
INSERT INTO FINAL_RESULT VALUES('S006','BITF21M022',35,30,40,39);
INSERT INTO FINAL_RESULT VALUES('S001','BITF21M023',35,32,40,37);
INSERT INTO FINAL_RESULT VALUES('S002','BITF21M023',35,34,40,36);
INSERT INTO FINAL_RESULT VALUES('S003','BITF21M023',35,32,40,34);
INSERT INTO FINAL_RESULT VALUES('S004','BITF21M023',35,33,40,32);
INSERT INTO FINAL_RESULT VALUES('S005','BITF21M023',35,33,40,33);
INSERT INTO FINAL_RESULT VALUES('S006','BITF21M023',35,32,40,33);

INSERT INTO FINAL_RESULT VALUES('S001','BITF21M010',35,32,40,37);
INSERT INTO FINAL_RESULT VALUES('S002','BITF21M010',35,34,40,36);
INSERT INTO FINAL_RESULT VALUES('S003','BITF21M010',35,32,40,34);
INSERT INTO FINAL_RESULT VALUES('S004','BITF21M010',35,33,40,32);
INSERT INTO FINAL_RESULT VALUES('S005','BITF21M010',35,33,40,33);
INSERT INTO FINAL_RESULT VALUES('S006','BITF21M010',35,32,40,33);

INSERT INTO FINAL_RESULT VALUES('S001','BITF21M032',35,32,40,37);
INSERT INTO FINAL_RESULT VALUES('S002','BITF21M032',35,34,40,36);
```

```
INSERT INTO FINAL_RESULT VALUES('S003','BITF21M032',35,32,40,34);
INSERT INTO FINAL_RESULT VALUES('S004','BITF21M032',35,33,40,32);
INSERT INTO FINAL_RESULT VALUES('S005','BITF21M032',35,33,40,33);
INSERT INTO FINAL_RESULT VALUES('S006','BITF21M032',35,32,40,33);

INSERT INTO FINAL_RESULT VALUES('S001','BITF21M009',35,33,40,39);
INSERT INTO FINAL_RESULT VALUES('S002','BITF21M009',35,32,40,32);
INSERT INTO FINAL_RESULT VALUES('S003','BITF21M009',35,29,40,34);
INSERT INTO FINAL_RESULT VALUES('S004','BITF21M009',35,28,40,34);
INSERT INTO FINAL_RESULT VALUES('S005','BITF21M009',35,33,40,35);
INSERT INTO FINAL_RESULT VALUES('S006','BITF21M009',35,31,40,34);

INSERT INTO FINAL_RESULT VALUES('S001','BITF21M003',35,33,40,39);
INSERT INTO FINAL_RESULT VALUES('S002','BITF21M003',35,32,40,32);
INSERT INTO FINAL_RESULT VALUES('S003','BITF21M003',35,29,40,34);
INSERT INTO FINAL_RESULT VALUES('S004','BITF21M003',35,28,40,34);
INSERT INTO FINAL_RESULT VALUES('S005','BITF21M003',35,33,40,35);
INSERT INTO FINAL_RESULT VALUES('S006','BITF21M003',35,31,40,34);

INSERT INTO FINAL_RESULT VALUES('S001','BITF21M020',35,33,40,39);
INSERT INTO FINAL_RESULT VALUES('S002','BITF21M020',35,32,40,32);
INSERT INTO FINAL_RESULT VALUES('S003','BITF21M020',35,29,40,34);
INSERT INTO FINAL_RESULT VALUES('S004','BITF21M020',35,28,40,34);
```

```
INSERT INTO FINAL_RESULT VALUES('S005','BITF21M020',35,33,40,35);
INSERT INTO FINAL_RESULT VALUES('S006','BITF21M020',35,31,40,34);
```

TRIGGERS:

- **TRIGGER TO CHECK TEACHING STAFF**

```
CREATE OR REPLACE TRIGGER TEACHING_MEMBER
BEFORE INSERT ON TEACHING
FOR EACH ROW
DECLARE
MEMBER_ROLE MEMBERS.ROLE_TYPE%TYPE;
BEGIN
SELECT ROLE_TYPE INTO MEMBER_ROLE FROM MEMBERS WHERE MEMBER_ID = :NEW.TMEM_ID;
IF UPPER(MEMBER_ROLE) <> 'TEACHING' THEN
RAISE_APPLICATION_ERROR(-20001,'NON_TEACHING STAFF CANNOT BE INSERTED');
END IF;
END;
```

- **TRIGGER TO CHECK FOR NON TEACHING MEMBER**

```
CREATE OR REPLACE TRIGGER NON_TEACHING_MEMBER
BEFORE INSERT ON NON_TEACHING
FOR EACH ROW
DECLARE
MEMBER_ROLE MEMBERS.ROLE_TYPE%TYPE;
BEGIN
SELECT ROLE_TYPE INTO MEMBER_ROLE FROM MEMBERS WHERE
MEMBER_ID = :NEW.NMEM_ID;
IF UPPER(MEMBER_ROLE) <> 'NON_TEACHING' THEN
RAISE_APPLICATION_ERROR(-20002,'TEACHING STAFF CANNOT BE
INSERTED');
END IF;
END;
```

- **TRIGGER TO CHECK CLASS ROOM**

```
CREATE OR REPLACE TRIGGER CLASSROOM_CHECK
BEFORE INSERT ON CLASSROOM
FOR EACH ROW
DECLARE
ROOM_TYPE_TEMP ROOM.ROOM_TYPE%TYPE;
BEGIN
```

```
SELECT ROOM_TYPE INTO ROOM_TYPE_TEMP FROM ROOM WHERE
ROOM_NO = :NEW.ROOM_NO;

IF UPPER(ROOM_TYPE_TEMP ) <> 'CLASSROOM' THEN

RAISE_APPLICATION_ERROR(-20003,' ROOM CANNOT BE INSERTED');

END IF;

END;
```

- **TRIGGER TO CHECK FOR LAB**

```
CREATE OR REPLACE TRIGGER LAB_CHECK

BEFORE INSERT ON LAB

FOR EACH ROW

DECLARE

ROOM_TYPE_TEMP ROOM.ROOM_TYPE%TYPE;

BEGIN

SELECT ROOM_TYPE INTO ROOM_TYPE_TEMP FROM ROOM WHERE
ROOM_NO = :NEW.ROOM_NO;

IF UPPER(ROOM_TYPE_TEMP ) <> 'LAB' THEN

RAISE_APPLICATION_ERROR(-20004,' ROOM CANNOT BE INSERTED');

END IF;

END;
```

- **TRIGGER TO CHECK THE MAXIMUM LIMIT OF THE SECTION**

```
CREATE OR REPLACE TRIGGER SECTION_LIMIT
BEFORE INSERT ON SECTION_ENROLLMENT
FOR EACH ROW
DECLARE
 ENROLLED_STUDENT NUMBER(2) := 0;
BEGIN
 SELECT COUNT(*) INTO ENROLLED_STUDENT FROM
SECTION_ENROLLMENT WHERE SECTION_ID = :NEW.SECTION_ID;
 IF ENROLLED_STUDENT = 50 THEN
  RAISE_APPLICATION_ERROR(-20999,'SECTION IS FULL');
END IF;
END;
```

- **TRIGGER TO CHECK TO WITHDRAW ONLY THOSE STUDENTS WHO ARE ENROLLED IN THAT SECTION. IF ENROLLED THEN REMOVE THAT STUDENT FROM SECTION_ENROLLMENT**

```
CREATE OR REPLACE TRIGGER SECTION_WITHDRAW_CHECK
BEFORE INSERT ON SECTION_WITHDRAW
FOR EACH ROW
DECLARE
   STUDENT_RECORD SECTION_ENROLLMENT%ROWTYPE;
BEGIN

   SELECT * INTO STUDENT_RECORD
   FROM SECTION_ENROLLMENT
   WHERE SECTION_ID = :NEW.SECTION_ID AND ROLLNO = :NEW.ROLLNO;



   IF SQL%NOTFOUND THEN
      RAISE_APPLICATION_ERROR(-20998, 'STUDENT IS NOT ENROLLED IN
SECTION');
   END IF;



   DELETE FROM SECTION_ENROLLMENT
   WHERE SECTION_ID = :NEW.SECTION_ID AND ROLLNO = :NEW.ROLLNO;
END;
```

- **TRIGGER TO CHECK IF USER NOT ENTERED MARKS GREATER THEN SESSIONALS MARKS**

```
CREATE OR REPLACE TRIGGER SESSIONAL_MARKS_CHECK
BEFORE INSERT OR UPDATE ON SESSIONAL_RESULT
FOR EACH ROW
DECLARE
TOTAL_MARKS SESSIONAL.S_MARKS%TYPE;
BEGIN
SELECT S_MARKS INTO TOTAL_MARKS FROM SESSIONAL WHERE SESSIONAL_ID = :NEW.SESSIONAL_ID;
IF :NEW.OBT_MARKS > TOTAL_MARKS THEN
RAISE_APPLICATION_ERROR(-20996,'OBTAINED MARSK IS HIGH THEN TOTAL MARKS');
END IF;
END;
```

- **TRIGGER TO CHECK THE OBT MARKS IS LESS THEN FINAL MARKS**

```
CREATE OR REPLACE TRIGGER FINAL_MARKS_CHECK
BEFORE INSERT OR UPDATE ON FINAL_RESULT
FOR EACH ROW
DECLARE
SESSIONAL_MARKS SESSIONAL_RESULT.OBT_MARKS%TYPE;
TOTAL_MARKS NUMBER(4,1) := 0;
BEGIN
```

```
IF :NEW.MID_OBT > :NEW.MID_TOTAL
OR :NEW.FINAL_OBT > :NEW.FINAL_TOTAL THEN

RAISE_APPLICATION_ERROR(-20995,'OBTAINED MARKS IS HIGH THEN TOTAL
MARKS');

END IF;


SELECT SUM(OBT_MARKS) INTO SESSIONAL_MARKS FROM SESSIONAL_RESULT
JOIN SESSIONAL ON SESSIONAL_RESULT.SESSIONAL_ID =
SESSIONAL.SESSIONAL_ID WHERE ROLLNO = :NEW.ROLLNO AND SECTION_ID
= :NEW.SECTION_ID;

TOTAL_MARKS := SESSIONAL_MARKS + :NEW.MID_OBT + :NEW.FINAL_OBT;


IF TOTAL_MARKS > 100 THEN

RAISE_APPLICATION_ERROR(-20993,'TOTAL MARKS CANNOT EXCEED 100');

END IF;

END;
```

- **TRIGGER TO INSERT MARKS IN COURSE_RESULT**

```
CREATE OR REPLACE TRIGGER COURSE_RESULT_INSERTION

AFTER INSERT OR UPDATE ON FINAL_RESULT

FOR EACH ROW

DECLARE

SESSIONAL_MARKS SESSIONAL_RESULT.OBT_MARKS%TYPE;

TOTAL_MARKS NUMBER(4,1) := 0;

COURSE_CODE COURSE.C_CODE%TYPE;

BEGIN
```

```
SELECT SUM(OBT_MARKS) INTO SESSIONAL_MARKS FROM SESSIONAL_RESULT
JOIN SESSIONAL ON SESSIONAL_RESULT.SESSIONAL_ID =
SESSIONAL.SESSIONAL_ID WHERE ROLLNO = :NEW.ROLLNO AND SECTION_ID
= :NEW.SECTION_ID;

TOTAL_MARKS := SESSIONAL_MARKS + :NEW.MID_OBT + :NEW.FINAL_OBT;


SELECT COURSE.C_CODE INTO COURSE_CODE FROM SECTION JOIN COURSE ON
COURSE.C_CODE = SECTION.C_CODE WHERE SECTION_ID = :NEW.SECTION_ID;


IF INSERTING THEN

INSERT INTO COURSE_RESULT
VALUES(COURSE_CODE,:NEW.SECTION_ID,TOTAL_MARKS,:NEW.ROLLNO);

ELSIF UPDATING THEN

UPDATE COURSE_RESULT

SET C_MARKS = TOTAL_MARKS WHERE ROLLNO = :NEW.ROLLNO AND
SECTION_ID = :NEW.SECTION_ID;

END IF;

END;
```

- TRIGGER TO CHECK THE PRE-REQ BEFORE INSERTION

```
CREATE OR REPLACE TRIGGER CHECK_PRE_REQ_CLEAR
BEFORE INSERT ON SECTION_ENROLLMENT
FOR EACH ROW
DECLARE
COURSES_RECORD PRE_REQ%ROWTYPE;
CURSOR PRE_COURSE IS SELECT * FROM PRE_REQ WHERE C_CODE = (SELECT
    C_CODE FROM SECTION WHERE SECTION_ID = :NEW.SECTION_ID);
IS_CLEAR NUMBER(2) := 0;

BEGIN
OPEN PRE_COURSE;
LOOP
FETCH PRE_COURSE INTO COURSES_RECORD;
EXIT WHEN PRE_COURSE%NOTFOUND;


SELECT COUNT(*) INTO IS_CLEAR FROM COURSE_RESULT WHERE ROLLNO
    = :NEW.ROLLNO AND C_CODE = COURSES_RECORD.P_CODE;
IF IS_CLEAR = 0 THEN
RAISE_APPLICATION_ERROR(-20112,'STUDENT NOT CLEARED THE PRE-REQ');
END IF;
IS_CLEAR := 0;
END LOOP;
CLOSE PRE_COURSE;
END;
```

SELECT STATEMENT

- **ATTENDANCE OF STUDENT**

```
SELECT
S.ROLLNO,STUDENT.STUDENT_NAME,S.SECTION_ID,S.ATT_DATE,S.ATT_STA
TUS,SE.C_CODE,C_TITLE FROM SECTION_ATTENDANCE S JOIN SECTION SE
ON S.SECTION_ID = SE.SECTION_ID
JOIN COURSE C ON SE.C_CODE = C.C_CODE  JOIN STUDENT ON
STUDENT.ROLLNO = S.ROLLNO;
```

ROLLNO	STUDENT_NAME	SECTION_ID	ATT_DATE	ATT_STATUS	C_CODE	C_TITLE
BITF21M003	AYESHA	S006	12/19/2023	A	HQ-282	QURAN TRANSLATION
BITF21M009	WARDA	S006	12/19/2023	A	HQ-282	QURAN TRANSLATION
BITF21M010	UMER	S006	12/19/2023	P	HQ-282	QURAN TRANSLATION
BITF21M020	HUZAIFA RAUF	S006	12/19/2023	A	HQ-282	QURAN TRANSLATION
BITF21M032	SYED JAFFAR ALI NAQVI	S006	12/19/2023	A	HQ-282	QURAN TRANSLATION
BITF21M023	MUHAMMAD HASEEB	S006	12/19/2023	A	HQ-282	QURAN TRANSLATION
BITF21M022	AQDAS SHABBIR	S006	12/19/2023	P	HQ-282	QURAN TRANSLATION
BITF21M003	AYESHA	S001	12/19/2023	P	CC-213	DATA STRUCTURE AND ALGORITHM
BITF21M009	WARDA	S001	12/19/2023	P	CC-213	DATA STRUCTURE AND ALGORITHM
BITF21M010	UMER	S001	12/19/2023	P	CC-213	DATA STRUCTURE AND ALGORITHM
BITF21M020	HUZAIFA RAUF	S001	12/19/2023	P	CC-213	DATA STRUCTURE AND ALGORITHM
BITF21M032	SYED JAFFAR ALI NAQVI	S001	12/19/2023	P	CC-213	DATA STRUCTURE AND ALGORITHM
BITF21M023	MUHAMMAD HASEEB	S001	12/19/2023	P	CC-213	DATA STRUCTURE AND ALGORITHM
BITF21M022	AQDAS SHABBIR	S001	12/19/2023	P	CC-213	DATA STRUCTURE AND ALGORITHM
BITF21M003	AYESHA	S002	12/19/2023	P	CC-215	DATABASE SYSTEMS
BITF21M009	WARDA	S002	12/19/2023	P	CC-215	DATABASE SYSTEMS
BITF21M010	UMER	S002	12/19/2023	P	CC-215	DATABASE SYSTEMS

- **STUDENTS ENROLLED IN COURSES**

```
SELECT
S.ROLLNO,S.STUDENT_NAME,SS.ENROLLMENT_DATE,SECTION.SECTION_ID,
SECTION.C_CODE,C_TITLE FROM STUDENT S JOIN SECTION_ENROLLMENT
SS ON S.ROLLNO = SS.ROLLNO JOIN SECTION ON SS.SECTION_ID =
SECTION.SECTION_ID JOIN COURSE ON COURSE.C_CODE =
SECTION.C_CODE ORDER BY S.ROLLNO;
```

ROLLNO	STUDENT_NAME	ENROLLMENT_DATE	SECTION_ID	C_CODE	C_TITLE
BITF21M003	AYESHA	12/16/2023	S006	HQ-282	QURAN TRANSLATION
BITF21M003	AYESHA	12/16/2023	S005	SI-21X	MODELING AND SIMULATION
BITF21M003	AYESHA	12/16/2023	S003	CC-212	SOFTWARE ENGINEERING
BITF21M003	AYESHA	12/16/2023	S002	CC-215	DATABASE SYSTEMS
BITF21M003	AYESHA	12/16/2023	S004	MS-252	LINEAR ALGEBRA
BITF21M003	AYESHA	12/16/2023	S001	CC-213	DATA STRUCTURE AND ALGORITHM
BITF21M009	WARDA	12/16/2023	S005	SI-21X	MODELING AND SIMULATION
BITF21M009	WARDA	12/16/2023	S004	MS-252	LINEAR ALGEBRA
BITF21M009	WARDA	12/16/2023	S003	CC-212	SOFTWARE ENGINEERING
BITF21M009	WARDA	12/16/2023	S002	CC-215	DATABASE SYSTEMS
BITF21M009	WARDA	12/16/2023	S006	HQ-282	QURAN TRANSLATION
BITF21M009	WARDA	12/16/2023	S001	CC-213	DATA STRUCTURE AND ALGORITHM
BITF21M010	UMER	12/16/2023	S005	SI-21X	MODELING AND SIMULATION
BITF21M010	UMER	12/16/2023	S001	CC-213	DATA STRUCTURE AND ALGORITHM
BITF21M010	UMER	12/16/2023	S002	CC-215	DATABASE SYSTEMS
BITF21M010	UMER	12/16/2023	S003	CC-212	SOFTWARE ENGINEERING

- **TEACHER TEACHING COURSES**

```
SELECT T.TMEM_ID,M.MEMBER_NAME ,S.C_CODE,C.C_TITLE FROM
TEACHING T JOIN MEMBERS M ON T.TMEM_ID = M.MEMBER_ID JOIN
SECTION S ON S.TMEM_ID = T.TMEM_ID
JOIN COURSE C ON C.C_CODE = S.C_CODE;
```

TMEM_ID	MEMBER_NAME	C_CODE	C_TITLE
M006	KHAWAR BUTT	HQ-282	QURAN TRANSLATION
M001	SYED WAQAR	CC-213	DATA STRUCTURE AND ALGORITHM
M002	ASIF SOHAIL	CC-215	DATABASE SYSTEMS
M004	ANZAR	CC-212	SOFTWARE ENGINEERING
M005	MAM ZARA	SI-21X	MODELING AND SIMULATION
M003	KASHIF MURTAZA	MS-252	LINEAR ALGEBRA

- **TEACHERS TOTAL COURSES**

```
SELECT M.MEMBER_NAME,C.C_TITLE,COUNT(S.SECTION_ID) AS "COURSES"
    FROM SECTION S JOIN MEMBERS M ON S.TMEM_ID = M.MEMBER_ID
JOIN COURSE C ON S.C_CODE = C.C_CODE
    GROUP BY M.MEMBER_NAME, C.C_TITLE;
```

MEMBER_NAME	C_TITLE	COURSES
ASIF SOHAIL	DATABASE SYSTEMS	1
KHAWAR BUTT	QURAN TRANSLATION	1
ANZAR	SOFTWARE ENGINEERING	1
MAM ZARA	MODELING AND SIMULATION	1
SYED WAQAR	DATA STRUCTURE AND ALGORITHM	1
KASHIF MURTAZA	LINEAR ALGEBRA	1

- **RETRIEVE COURSE AND NUMBER OF STUDENTS ENROLLED IN IT**

```
SELECT SE.SECTION_ID,C.C_TITLE,COUNT(SE.SECTION_ID) AS "STUDENTS
    ENROLLED",M.MEMBER_NAME AS "TEACHER" FROM
    SECTION_ENROLLMENT SE JOIN SECTION S ON SE.SECTION_ID =
    S.SECTION_ID
JOIN COURSE C ON S.C_CODE = C.C_CODE JOIN MEMBERS M ON S.TMEM_ID =
    M.MEMBER_ID
    GROUP BY SE.SECTION_ID,C.C_TITLE, M.MEMBER_NAME;
```

SECTION_ID	C_TITLE	STUDENTS ENROLLED	TEACHER
S004	LINEAR ALGEBRA	7	KASHIF MURTAZA
S003	SOFTWARE ENGINEERING	7	ANZAR
S006	QURAN TRANSLATION	7	KHAWAR BUTT
S002	DATABASE SYSTEMS	7	ASIF SOHAIL
S005	MODELING AND SIMULATION	7	MAM ZARA
S001	DATA STRUCTURE AND ALGORITHM	7	SYED WAQAR

VIEWS

- **STUDENT COURSES RESULT VIEW**

```
CREATE OR REPLACE VIEW STUDENTS_RESULT_VIEW
AS SELECT
    STUDENT.STUDENT_NAME,COURSE_RESULT.ROLLNO,COURSE_RESULT.C_
    MARKS,GRADING.GRADE,GRADING.GRADE_POINTS,COURSE.C_TITLE,COU
    RSE.C_HOURS,STUDENT.STU_BATCH,DEGREE_PROGRAM.DEGREE_NAME,
    SECTION.SEMISTER FROM COURSE_RESULT JOIN COURSE ON
    COURSE_RESULT.C_CODE = COURSE.C_CODE JOIN STUDENT ON
    STUDENT.ROLLNO = COURSE_RESULT.ROLLNO JOIN GRADING ON
    COURSE_RESULT.C_MARKS BETWEEN GRADING.LOWMARKS AND
    GRADING.HIMARKS
    JOIN DEGREE_PROGRAM ON STUDENT.DEGREE_ID =
    DEGREE_PROGRAM.DEGREE_ID JOIN SECTION ON SECTION.SECTION_ID =
    COURSE_RESULT.SECTION_ID  ORDER BY COURSE_RESULT.ROLLNO;
```

STUDENT_NAME	ROLLNO	C_MARKS	GRADE	GRADE_POINTS	C_TITLE	C_HOURS	STU_BATCH	DEGREE_NAME	SEMISTER
AYESHA	BITF21M003	78	B+	3.3	DATABASE SYSTEMS	3	F21	BSIT	1
AYESHA	BITF21M003	89	A+	4	QURAN TRANSLATION	.5	F21	BSIT	1
AYESHA	BITF21M003	80	A-	3.7	LINEAR ALGEBRA	3	F21	BSIT	1
AYESHA	BITF21M003	95	A+	4	DATA STRUCTURE AND ALGORITHM	3	F21	BSIT	1
AYESHA	BITF21M003	86	A+	4	MODELING AND SIMULATION	3	F21	BSIT	1
AYESHA	BITF21M003	86	A+	4	SOFTWARE ENGINEERING	3	F21	BSIT	1
WARDA	BITF21M009	89	A+	4	QURAN TRANSLATION	.5	F21	BSIT	1
WARDA	BITF21M009	90	A+	4	DATA STRUCTURE AND ALGORITHM	3	F21	BSIT	1
WARDA	BITF21M009	76	B+	3.3	LINEAR ALGEBRA	3	F21	BSIT	1
WARDA	BITF21M009	81	A-	3.7	SOFTWARE ENGINEERING	3	F21	BSIT	1
WARDA	BITF21M009	82	A-	3.7	MODELING AND SIMULATION	3	F21	BSIT	1
WARDA	BITF21M009	84	A-	3.7	DATABASE SYSTEMS	3	F21	BSIT	1
UMER	BITF21M010	88	A+	4	DATA STRUCTURE AND ALGORITHM	3	F21	BSIT	1
UMER	BITF21M010	89	A+	4	QURAN TRANSLATION	.5	F21	BSIT	1
UMER	BITF21M010	80	A-	3.7	DATABASE SYSTEMS	3	F21	BSIT	1
UMER	BITF21M010	80	A-	3.7	LINEAR ALGEBRA	3	F21	BSIT	1
UMER	BITF21M010	81	A-	3.7	MODELING AND SIMULATION	3	F21	BSIT	1

- **STUDENTS GPA VIEW**

```
CREATE OR REPLACE VIEW STUDENTS_GPA_VIEW
AS SELECT DISTINCT
    STUDENT_NAME,DEGREE_NAME,SECTION.SEMISTER,GPA_CALCULATOR(SE
    CTION_ENROLLMENT.ROLLNO,SECTION.SEMISTER) AS "GPA" FROM
    STUDENT JOIN DEGREE_PROGRAM ON STUDENT.DEGREE_ID =
    DEGREE_PROGRAM.DEGREE_ID JOIN SECTION_ENROLLMENT
    ON SECTION_ENROLLMENT.ROLLNO = STUDENT.ROLLNO JOIN SECTION ON
    SECTION.SECTION_ID = SECTION_ENROLLMENT.SECTION_ID;
```

STUDENT_NAME	DEGREE_NAME	SEMISTER	GPA
WARDA	BSIT	1	3.7
HUZAIFA RAUF	BSIT	1	3.7
MUHAMMAD HASEEB	BSIT	1	3.9
AYESHA	BSIT	1	3.8
AQDAS SHABBIR	BSIT	1	4
SYED JAFFAR ALI NAQVI	BSIT	1	3.7
UMER	BSIT	1	3.8

FUNCTIONS

- **Function to calculate the gpa**

```
CREATE OR REPLACE FUNCTION GPA_CALCULATOR(STU_ROLLNO
COURSE_RESULT.ROLLNO%TYPE,STU_SEMISTER SECTION.SEMISTER%TYPE)
RETURN NUMBER
IS

TOTAL_COURSE_MARKS COURSE_RESULT.C_MARKS%TYPE;
STU_GRADE_POINTS GRADING.GRADE_POINTS%TYPE;
COURSE_HOURS COURSE.C_HOURS%TYPE;
CURSOR STUDENT_CURSOR IS SELECT C_MARKS,GRADE_POINTS,C_HOURS
    FROM COURSE_RESULT JOIN GRADING ON C_MARKS BETWEEN
    LOWMARKS AND HIMARKS
JOIN SECTION ON SECTION.SECTION_ID = COURSE_RESULT.SECTION_ID JOIN
    COURSE ON COURSE.C_CODE = COURSE_RESULT.C_CODE
WHERE COURSE_RESULT.ROLLNO = STU_ROLLNO AND SECTION.SEMISTER =
    STU_SEMISTER;

TOTAL_POINTS_COURSE NUMBER(4,2) := 0;

TOTAL_GRADE_POINTS_SUM NUMBER(4,2) := 0;

BEGIN
OPEN STUDENT_CURSOR;
LOOP
FETCH STUDENT_CURSOR INTO
    TOTAL_COURSE_MARKS,STU_GRADE_POINTS,COURSE_HOURS;
EXIT WHEN STUDENT_CURSOR%NOTFOUND;
TOTAL_POINTS_COURSE := TOTAL_POINTS_COURSE + (STU_GRADE_POINTS *
    COURSE_HOURS);
TOTAL_GRADE_POINTS_SUM  := TOTAL_GRADE_POINTS_SUM +
    COURSE_HOURS;
END LOOP;

CLOSE STUDENT_CURSOR;
```

```
IF TOTAL_GRADE_POINTS_SUM = 0 THEN
RETURN 0;
END IF;
RETURN ROUND(TOTAL_POINTS_COURSE / TOTAL_GRADE_POINTS_SUM,1);
END;
```

- FUNCTION TO GET THE NUMBER OF COURSES A STUDENT IS ENROLLED

```
CREATE OR REPLACE FUNCTION get_enrolled_courses_count(
   p_rollno  STUDENT.ROLLNO%TYPE
) RETURN NUMBER
IS
   v_enrolled_count NUMBER;
BEGIN

   SELECT COUNT(DISTINCT SECTION_ID)
   INTO v_enrolled_count
   FROM SECTION_ENROLLMENT
   WHERE ROLLNO = p_rollno
   AND NOT EXISTS (SELECT C_CODE FROM COURSE_RESULT WHERE ROLLNO
    =p_rollno AND SECTION_ID = SECTION_ENROLLMENT.SECTION_ID );


   RETURN v_enrolled_count;
END get_enrolled_courses_count;
```


PROCEDURES

- PROCEDURE TO CALCULATE THE ATTENDANCE IN A

PARTICULAR SECTION

```
CREATE OR REPLACE PROCEDURE TOTAL_ATTENDANCE(
   S_ROLLNO IN SECTION_ATTENDANCE.ROLLNO%TYPE,
   S_SECTION_ID IN SECTION_ATTENDANCE.SECTION_ID%TYPE
) IS
   COUNT_ATT NUMBER(10) := 0;
   ENROLLED NUMBER(10) := 0;
   SECTION_EXISTS NUMBER(2) := 0;
   STUDENT_EXISTS NUMBER(2) := 0;
BEGIN
   SELECT COUNT(*)
   INTO SECTION_EXISTS
   FROM SECTION
   WHERE SECTION_ID = S_SECTION_ID;

   SELECT COUNT(*)
   INTO STUDENT_EXISTS
   FROM STUDENT
   WHERE ROLLNO = S_ROLLNO;



   SELECT COUNT(*)
   INTO ENROLLED
   FROM SECTION_ENROLLMENT
   WHERE ROLLNO = S_ROLLNO AND SECTION_ID = S_SECTION_ID;

   IF SECTION_EXISTS = 0 THEN
     DBMS_OUTPUT.PUT_LINE('SECTION DOES NOT EXIST');
   ELSIF STUDENT_EXISTS = 0 THEN
      DBMS_OUTPUT.PUT_LINE('STUDENT DOES NOT EXIST');
   ELSIF ENROLLED = 0 THEN
      DBMS_OUTPUT.PUT_LINE('STUDENT IS NOT ENROLLED IN SECTION');
   ELSE
```

```
    SELECT COUNT(*)
    INTO COUNT_ATT
    FROM SECTION_ATTENDANCE
    WHERE ROLLNO = S_ROLLNO AND SECTION_ID = S_SECTION_ID;

    DBMS_OUTPUT.PUT_LINE('Total attendance for student ' || S_ROLLNO || '
  in Section ' || S_SECTION_ID || ': ' || COUNT_ATT);
  END IF;
END;
```

```
Total attendance for student BITF21M022 in Section S001: 1
```

- PROCEDURE TO VIEW THE CGPA OF THE STUDENT

```
CREATE OR REPLACE PROCEDURE CALCULATE_CGPA(S_ROLLNO
    STUDENT.ROLLNO%TYPE)
IS

  S_RECORD STUDENTS_GPA_VIEW%ROWTYPE;
  CURSOR STUDENT_CURSOR  IS SELECT * FROM STUDENTS_GPA_VIEW
   WHERE ROLLNO = S_ROLLNO;
  CGPA NUMBER(4,1) := 0;
  COUNT_SEMISTER NUMBER(1) := 0;
BEGIN


  OPEN STUDENT_CURSOR;
  LOOP
    FETCH STUDENT_CURSOR INTO S_RECORD;
    EXIT WHEN STUDENT_CURSOR%NOTFOUND;
    CGPA := CGPA + S_RECORD.GPA;
    COUNT_SEMISTER := COUNT_SEMISTER + 1;
  END LOOP;
  CLOSE STUDENT_CURSOR;
```

```
    IF COUNT_SEMISTER = 0 THEN
    DBMS_OUTPUT.PUT_LINE('RECORD NOT FOUND IN VIEW');
    ELSE
    DBMS_OUTPUT.PUT_LINE('CGPA = ' || CGPA/COUNT_SEMISTER);
    END IF;

END;
```

```
CGPA = 3.775

Statement processed.
```