Submitted By:

Ammar Jamil

01-134231-010

Zameer Hussain

01-134231-096

Zaryab Ahmed

01-134231-097

Course Registration Conflict Resolution System

Bachelor of Science in Computer Science – 4(A)

Submitted to: Mam Fatima Khalique

Department of Computer Science

Bahria Universtiy, Islamabad

Table of Content

4.1 Normalization of Relational Model

4.1.1 First Normal Form (1NF)

First Normal Form requires that the values in each column of a table are atomic (indivisible). Each table must have a primary key, and there should be no repeating groups or arrays.

- **Given Schema in 1NF**
  - Advisor: (A-ID, A-Name, A-Email)
  - Student: (S-Name, S-Email, S-Enrollment)
  - Course: (Course-ID, Course-Name, Credit-Hours, A-Id, Year, Semester)
  - Course Offering (Year, Semester)
  - Timetable:(Start-Time, End-Time, Day, Section-Name, Course-Id, Room-No, Building)
  - Classroom:(Room-No, Building, Capacity)
  - Section: (Section_Name)
  - Register:(S-Enrollment, Course-Id, Section_Name)

4.1.2 Second Normal Form (2NF)

The first condition of 2NF is that table must be in 1NF. Secondly there should not be any partial dependency (attribute should not depend on part of composite key).

- **Given Schema in 2NF**
  - Advisor: (A-ID, A-Name, A-Email)
  - Student: (S-Name, S-Email, S-Enrollment)
  - Course: (Course-ID, Course-Name, Credit-Hours, A-Id, Year, Semester)
  - Course Offering (Year, Semester)
  - Timetable:(Start-Time, End-Time, Day, Section-Name, Course-Id, Room-No, Building)
  - Classroom:(Room-No, Building, Capacity)
  - Section: (Section_Name)
  - Register:(S-Enrollment, Course-Id, Section_Name)
- **Revised 2NF:**
  - Advisor: (A-ID, A-Name, A-Email)
  - Student: (S-Name, S-Email, S-Enrollment)
  - Course: (Course-ID, Course-Name, Credit-Hours, A-Id, Year, Semester)
  - Course Offering (Year, Semester)
  - Timetable:(Start-Time, End-Time, Day, Section-Name, Course-Id, Room-No, Building)
  - Classroom:(Room-No, Building, Capacity)
  - Section: (Section_Name)

- Register:(S-Enrollment, Course-Id, Section_Name)

4.1.3 Third Normal Form (3NF)

Third Normal form states that there must not be any transitive dependency (non-key attributes should not determine other non-key attributes).

- **Given Schema in 3NF**
- **Removing Transitive Dependencies**
  - Advisor: Transitive dependency present
  - Student: Transitive dependency present
  - Course: Transitive dependency present
  - Course Offering: No Transitive dependency
  - Timetable: No Transitive dependency
  - Classroom: No Transitive dependency
  - Register: No Transitive dependency

  Revised Schema in 3NF
  - Advisor: (A-ID, A-Email)
- Advisor-Information:(A-Email, A-Name)
  - Student:(S-Enrollment, S-Email)
- Student-Information:(S-Email, S-Name)
  - Course: (Course-ID, Course-Name, A-Id, Year, Semester)
- Course-Information:(Course-Name, Credit-Hour)
  - Course Offering (Year, Semester)
  - Timetable:(Start-Time, End-Time, Day, Section-Name, Course-Id, Room-No)
  - Classroom:(Room-No, capacity)
- Building-Info: (RoomNo, Building)
  - Register:(S-Enrollment, Course-Id, Section_Name)

4.1.4 Summary of the Normalization Process

- **1NF**: Ensures atomicity and uniqueness of each column.
- **2NF**: Eliminates partial dependencies; all non-key attributes are fully dependent on the primary key.
- **3NF**: Eliminates transitive dependencies; all attributes are directly dependent on the primary key.

4.2 DDL Script

4.2.1 Script Details

The following scripts will create the necessary tables, define primary keys (PK), and foreign key (FK) constraints.

4.2.2 Execution Instructions

To execute these scripts within your chosen DBMS environment (e.g., MySQL, PostgreSQL, SQL Server), follow these steps:

- **Open the DBMS environment:** Launch your DBMS environment (e.g., MySQL Workbench, pgAdmin, SQL Server Management Studio).
- **Connect to the database:** Connect to the appropriate database where you want to create the tables. If needed, create a new database.
- **Execute the scripts:** Copy the scripts below into the SQL query editor and execute them sequentially.

4.3 DDL Script

4.3.1. Create the Advisor table

```
CREATE TABLE Advisor(
    A_ID INT PRIMARY KEY,
    A_Name VARCHAR(30),
    A_Email VARCHAR(30)
);
```

4.3.2. Create the Student table

```
CREATE TABLE Student(
    S_Enrollment INT PRIMARY KEY,
    S_Name VARCHAR(30),
    S_Email VARCHAR(30)
);
```

4.3.3. Create the CourseOffering table

```
CREATE TABLE CourseOffering(
    Semester VARCHAR(10),
    Year INT,
    PRIMARY KEY (Semester, Year)
);
```

4.3.4. Create the CourseDetails table

```
CREATE TABLE CourseDetails(
    Course_Name VARCHAR(30) PRIMARY KEY,
    Credit_Hour INT
);
```

4.3.5. Create the Course table

```
CREATE TABLE Course (
    C_ID VARCHAR(10) PRIMARY KEY,
    Course_Name VARCHAR(30),
    A_ID INT,
    Semester VARCHAR(10),
    Year INT,
    FOREIGN KEY (A_ID) REFERENCES Advisor(A_ID),
    FOREIGN KEY (Course_Name) REFERENCES CourseDetails(Course_Name)
);
```

4.3.6. Create the ClassRoom table

```
CREATE TABLE ClassRoom(
    RoomNo VARCHAR(5) PRIMARY KEY,
    Capacity INT
);
```

4.3.7. Create the TimeTable table

```
CREATE TABLE TimeTable(
    Day VARCHAR(10),
    Section_Name VARCHAR(5),
    Start_Time Varchar(15),
    End_Time TIME Varchar(15),
    RoomNo VARCHAR(5),
```

```
    C_ID VARCHAR(10),
    PRIMARY KEY (Day, Section_Name, Start_Time),
    FOREIGN KEY (RoomNo) REFERENCES ClassRoom(RoomNo),
    FOREIGN KEY (C_ID) REFERENCES Course(C_ID,
    FOREIGN KEY (Section_Name) REFERENCES Section(Section_Name)
);
```

4.3.8. Create the Register table

```
CREATE TABLE Register(
    S_Enrollment INT,
    C_ID VARCHAR(10),
    Section_Name varchar(5),
    PRIMARY KEY (S_Enrollment, C_ID,Section_Name),
    FOREIGN KEY (S_Enrollment) REFERENCES Student(S_Enrollment),
    FOREIGN KEY (C_ID) REFERENCES Course(C_ID),
    FOREIGN KEY (Section_Name) REFERENCES Section(Section_Name)
);
```

4.3.9. Create the Section table

```
CREATE TABLE SECTION(
    Section_Name varchar(5) PRIMARY KEY
);
```