DBMS PROJECT

Step I

Describing the problem and ER model with necessary assumptions

PROBLEM STATEMENT:

The following Entity Relationship(ER) model pictorially represents the working of a JoSAA (Joint Seat Allocation Authority) database which allots seats to students to premier institutes all over the country. JosAA is set up by the MHRD to allot seats in 100 institutes in India based on the RANK obtained in JEE Mains , CATEGORY and PREFERENCES. Several rounds of allotment take place to ensure that there aren't any vacancies. This database simulates the allotment results after one such round in this procedure.

GROUP MEMBERS:

- MVS Srimanth (Btech, ECE 3rd year)
- Souradeep Sarkar (Msc,Mathematics-2nd Year)

Course Instructor:

Dr. T Ramakrishnudu
Assistant Professor .CSE Department.

Legend for ER Diagram:

- **The -+ headed arrow directed towards an entity indicates cardinality one.**
- **The -< headed arrow directed towards an entity indicates cardinality many.**
- **( ) around an attribute marks it a compound/ composite attribute.**
- **The 'o' beside an attribute means it is optional (can be NULL)**
- **Other formalities follow according to standards mentioned.**

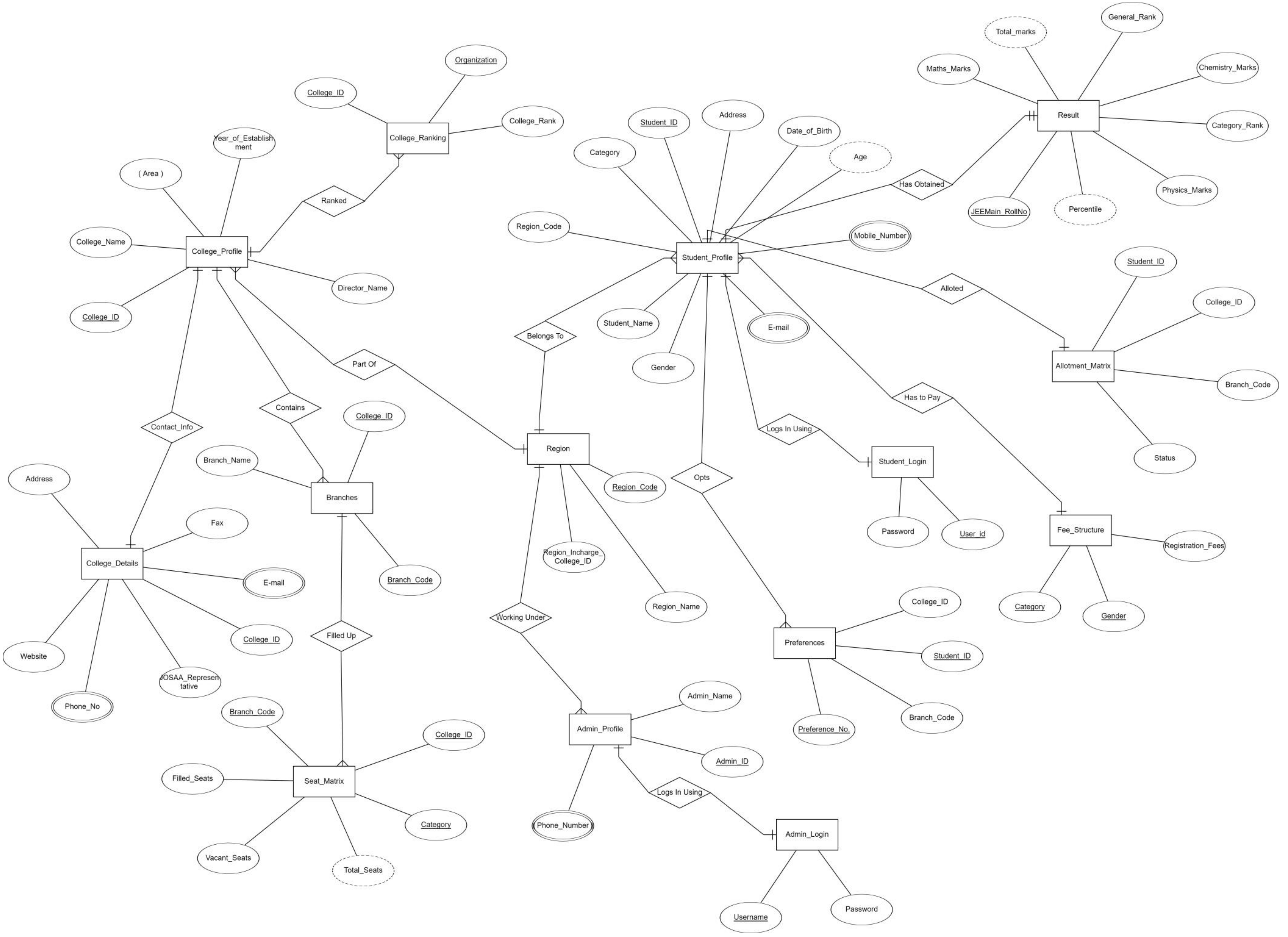
Assumptions

We have made the following assumptions during the creation of our ER Diagram.

- It is assumed that the student has obtained a unique rank in the JEE Mains
- A college can be ranked by multiple organizations so that students can compare different institutes.
- A region comprises of different colleges.
- Students are also categorized into various regions for convenience in administration.
- Administrators are allotted to each region who can log on to their regional part of the Database with their unique User id and password.
- One student is given one username and password only.
- An institute can have many branches.
- Seats in a branch are divided into several categories (EBC/ PWD/GEN)
- Irrespective of the college allotted, all students belonging to a particular category and gender have to pay the same fee at the time of registration.
- A student has the privilege to fill any number of preferences before allotment.
- After all students have filled in their choices they are allotted 1 institute and branch or none.
- They can change their status to FREEZE (Confirm option) or FLOAT (Wait for the next iteration).
- In case all the preferences opted by a candidate are filled up, he will be allotted no seat.

Disclaimer: This mini project is not associated with JoSAA or with the MHRD or any of its associates in any respects. It is for student/course purpose only.

*** ER Diagram overleaf ***


General_Rank
Total_marks
Organization
Maths_Marks
Chemistry_Marks
College_ID
Result
College_Rank
College_Ranking
Student_ID
Address
Date_of_Birth
Category_Rank
Year_of_Establishment
Category
Age
(Area)
Has Obtained
Physics_Marks
Ranked
JEEMain_RollNo
Percentile
Region_Code
Mobile_Number
College_Name
College_Profile
Student_Profile
Student_ID
Director_Name
Alloted
College_ID
College_ID
E-mail
Student_Name
Belongs To
Gender
Allotment_Matrix
Part Of
Branch_Code
Has to Pay
Contains
College_ID
Contact_Info
Logs In Using
Region
Branch_Name
Student_Login
Status
Address
Opts
Region_Code
Branches
Fax
Fee_Structure
Password
User_id
Registration_Fees
College_Details
Region_Incharge_College_ID
E-mail
Branch_Code
College_ID
Region_Name
Category
Working Under
Gender
Filled Up
Preferences
College_ID
Student_ID
Website
JOSAA_Representative
Phone_No
Admin_Name
Branch_Code
College_ID
Admin_Profile
Preference_No.
Branch_Code
Filled_Seats
Seat_Matrix
Admin_ID
Logs In Using
Category
Phone_Number
Admin_Login
Vacant_Seats
Total_Seats
Password
Username

Step II:

Creating the Relational Schema from the ER Diagram and Normalizing the Tables so obtained

Overleaf are the expected tables that are created by observing the ER Model on the previous page.

Clearly the ER Model does not result in a 1NF Schema. This is due to non atomic attributes including Admin-Phone Number, StudentEmail_Id , Student Phone , College_Phone and College_Phone .(bearing multiple values)

So separate Relations to be created are:

- **StudentPhone (Student_ID,PhoneNumber)**
- **StudentEmail(Student_ID,Email_ID)**
- **CollegePhone (College_ID,PhoneNumber)**
- **CollegeEmail(College_ID,Email_ID)**
- **AdminPhone(Admin_ID,PhoneNumber)**
- **AdminEmail(Admin_ID,Email_ID)**

Now it is clear that the Relational Schema is at least in 1NF.

Checking for the Normal Form of Relations in Schema Overleaf

Based on our assumptions, the primary keys shown in the figure on the next page determine all the other attributes in all relational tables except in the 'Branch_Details' table.

(A detailed list of functional dependencies in all relations is given at the end)

So all the other relations are in BCNF. This will be satisfactory for the present.

In 'Branch_Details' table, we have the following relations.

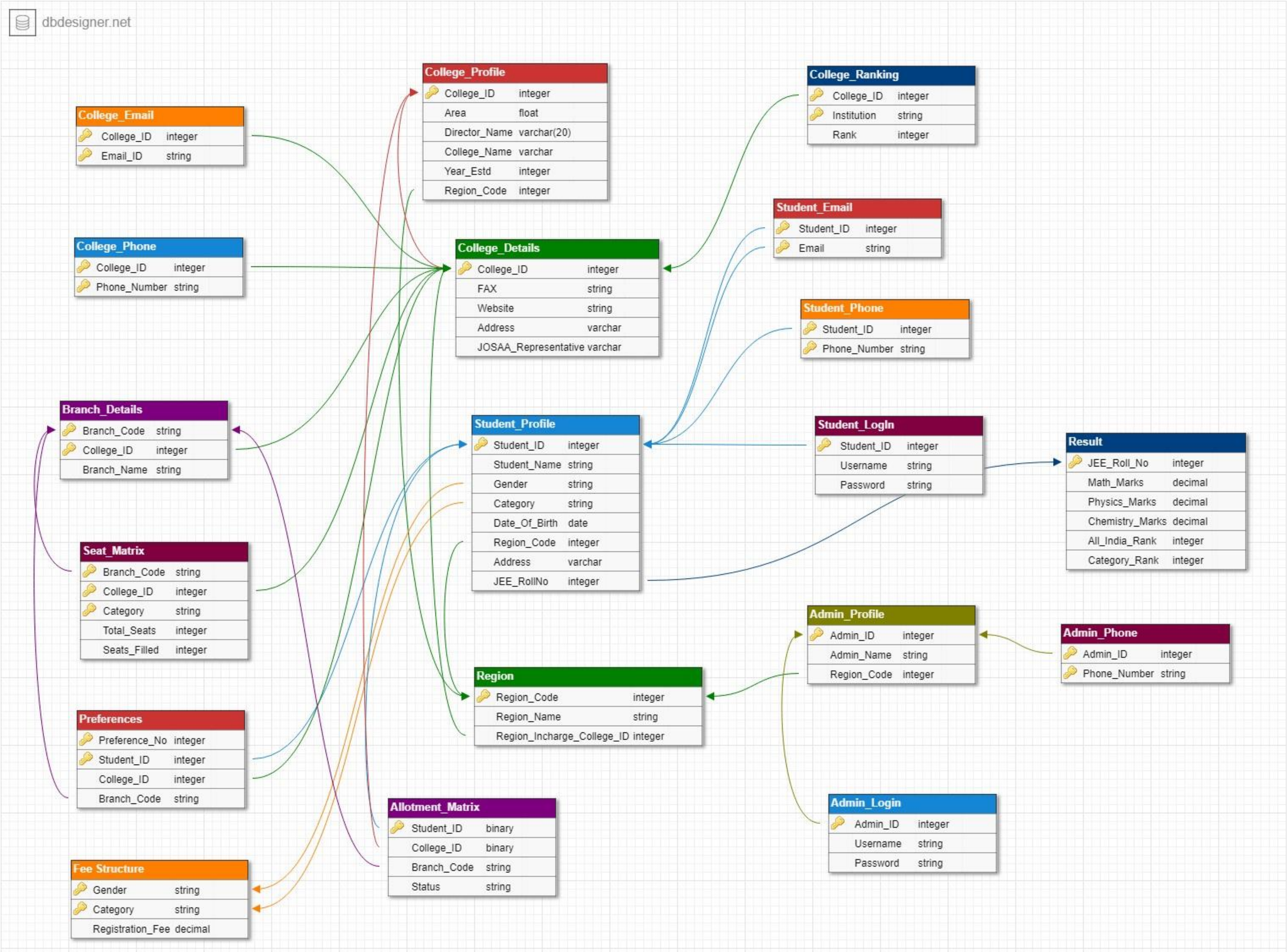
- **Branch_Code->Branch_Name**
- **Branch_Code,College_ID->Branch_Name**

Here there is a partial dependency on key attribute in first dependency.
So there is a need to decompose this relation into two smaller relations

- **BranchIndex(Branch_Code, Branch_Name)**
- **CollegeBranches(College_ID,Branch_Code)**

Now the two relations are in BCNF and 3NF.

****Next page is the first set of relational tables followed by Functional Dependencies** (Schema I)**



dbdesigner.net

College_Profile

🔑	College_ID	integer
	Area	float
	Director_Name	varchar(20)
	College_Name	varchar
	Year_Estd	integer
	Region_Code	integer

College_Ranking

🔑	College_ID	integer
🔑	Institution	string
	Rank	integer

College_Email

🔑	College_ID	integer
🔑	Email_ID	string

College_Phone

🔑	College_ID	integer
🔑	Phone_Number	string

College_Details

🔑	College_ID	integer
	FAX	string
	Website	string
	Address	varchar
	JOSAA_Representative	varchar

Student_Email

🔑	Student_ID	integer
🔑	Email	string

Student_Phone

🔑	Student_ID	integer
🔑	Phone_Number	string

Branch_Details

🔑	Branch_Code	string
🔑	College_ID	integer
	Branch_Name	string

Student_Profile

🔑	Student_ID	integer
	Student_Name	string
	Gender	string
	Category	string
	Date_Of_Birth	date
	Region_Code	integer
	Address	varchar
	JEE_RollNo	integer

Student_LogIn

🔑	Student_ID	integer
	Username	string
	Password	string

Result

🔑	JEE_Roll_No	integer
	Math_Marks	decimal
	Physics_Marks	decimal
	Chemistry_Marks	decimal
	All_India_Rank	integer
	Category_Rank	integer

Seat_Matrix

🔑	Branch_Code	string
🔑	College_ID	integer
🔑	Category	string
	Total_Seats	integer
	Seats_Filled	integer

Admin_Profile

🔑	Admin_ID	integer
	Admin_Name	string
	Region_Code	integer

Admin_Phone

🔑	Admin_ID	integer
🔑	Phone_Number	string

Region

🔑	Region_Code	integer
	Region_Name	string
	Region_Incharge_College_ID	integer

Preferences

🔑	Preference_No	integer
🔑	Student_ID	integer
	College_ID	integer
	Branch_Code	string

Allotment_Matrix

🔑	Student_ID	binary
	College_ID	binary
	Branch_Code	string
	Status	string

Admin_Login

🔑	Admin_ID	integer
	Username	string
	Password	string

Fee Structure

🔑	Gender	string
🔑	Category	string
	Registration_Fee	decimal

RELATIONS IN THE DATABASE BEFORE NORMALIZATION

1. **College_Profile (College_ID, Area, Director_Name, College_Name, Year_Estd, Region_Code)**

   FUNCTIONAL DEPENDENCIES:

   College_ID→Area

   College_ID→Director_Name

   College_ID→College_Name

   College_ID→Year_Estd

   College_ID→Region_Code

2. **College Details(College_ID, FAX, Website, Address, JOSAA_Representative)**

   FUNCTIONAL DEPENDENCIES:

   College_ID→FAX

   College_ID→Website

   College_ID→Address

   College_ID→JOSAA_Representative

3. **Student_Profile(Student_ID, Student_Name, Gender, Category, Date_of_Birth, Region_Code, Address, JEE_Roll_No)**

   FUNCTIONAL DEPENDENCIES:

   Student_ID→Student_Name
   Student_ID→Gender
   Student_ID→Category
   Student_ID→Date_of_Birth
   Student_ID→Region_Code
   Student_ID→Address
   Student_ID→JEE_Roll_No

4. **Region(Region_Code,Region_Name,Region_Incharge_College_ID)**

FUNCTIONAL DEPENDENCIES:

Region_Code→Region_Name
Region_Code→ Region_Incharge_College_ID

5. **Allotment_Matrix(Student_ID,College_ID,Branch_Code,Status)**

FUNCTIONAL DEPENDENCIES:

Student_ID→College_ID
Student_ID→Branch_Code Student_ID→Status

6. **College_Ranking(College_ID,Institution, Rank)**

FUNCTIONAL DEPENDENCIES:

{College_ID,Institution}→Rank

7. **Student_Email(Student_ID,Email)**

8. **Student_Phone(Student_ID,Phone_Number)**

9. **Student_Login(Student_ID,Username,Password)**

FUNCTIONAL DEPENDENCIES:

Student_ID→Username
Student_ID→Password

10. **Admin_Profile(Admin_ID,Admin_Name,Region_Code)**

    FUNCTIONAL DEPENDENCIES:

    Admin_ID→Admin_Name
    Admin_ID→Region_Code

11. **Admin_Profile(Admin_ID,Username,Password)**

    FUNCTIONAL DEPENDENCIES:

    Admin_ID→Username
    Admin_ID→Password

12. **Result(JEE_Roll_No, Math_Marks,Physics_Marks,Chemistry_Marks,All_India_Rank,Category_Rank)**

    FUNCTIONAL DEPENDENCIES:

    JEE_Roll_No→Math_Marks
    JEE_Roll_No→Physics_Marks

    JEE_Roll_No→Chemistry_Marks
    JEE_Roll_No→All_India_Rank
    JEE_Roll_No→Category_Rank

13. **Admin_Phone(Admin_ID,Phone_Number)**

14. **College_Email(College_ID,Email_ID)**

15. **College_Phone(College_ID ,Phone_Number)**

    FUNCTIONAL DEPENDENCIES:

    College_ID→Phone_Number

16. **Branch_Details(Branch_Code,College_ID,Branch_Name)**

FUNCTIONAL DEPENDENCIES:

{Branch_Code,College_ID}→ Branch_Name

Branch_Code→Branch_Name

17. **Seat_Matrix(Branch_Code,College_ID,Category,Total_Seats,Seats_Filled)**

FUNCTIONAL DEPENDENCIES:

{Branch_Code,College_ID,Category}→Total_Seats

{Branch_Code,College_ID,Category}→Seats_Filled

18. **Preferences(Preference_No,Student_ID,College_ID,Branch_Code)**

FUNCTIONAL DEPENDENCIES:

{Preference_No,Student_ID}→College_ID

{Preference_No,Student_ID}→Branch_Code

19. **Fee_Structure(Gender,Category,Registration_Fee)**

FUNCTIONAL DEPENDENCIES:

{Gender,Category}→Registration_Fee

Denormalization for Less Redundancy

However the relational schema we developed earlier is in 3NF and BCNF, we still have redundancy.

In case of:

- **Insertions**
- **Updations**
- **Deletions**

This is because one such relation College_Profile sharing a 1-1 relation with College_Details is unnecessary independently. Instead, we can have a merged relation as follows

- **College_Profile (All attributes in College_Profile and College_Details)**
- **The PK of this relation will be College_ID as it is the only unique identifier.**

On similar note Student_Profile and Student_Login can be merged as:

- **Student_Profile(All Attributes of Student_Profile and Student_Login)**
- **The PK of this relation will be Student_ID.**

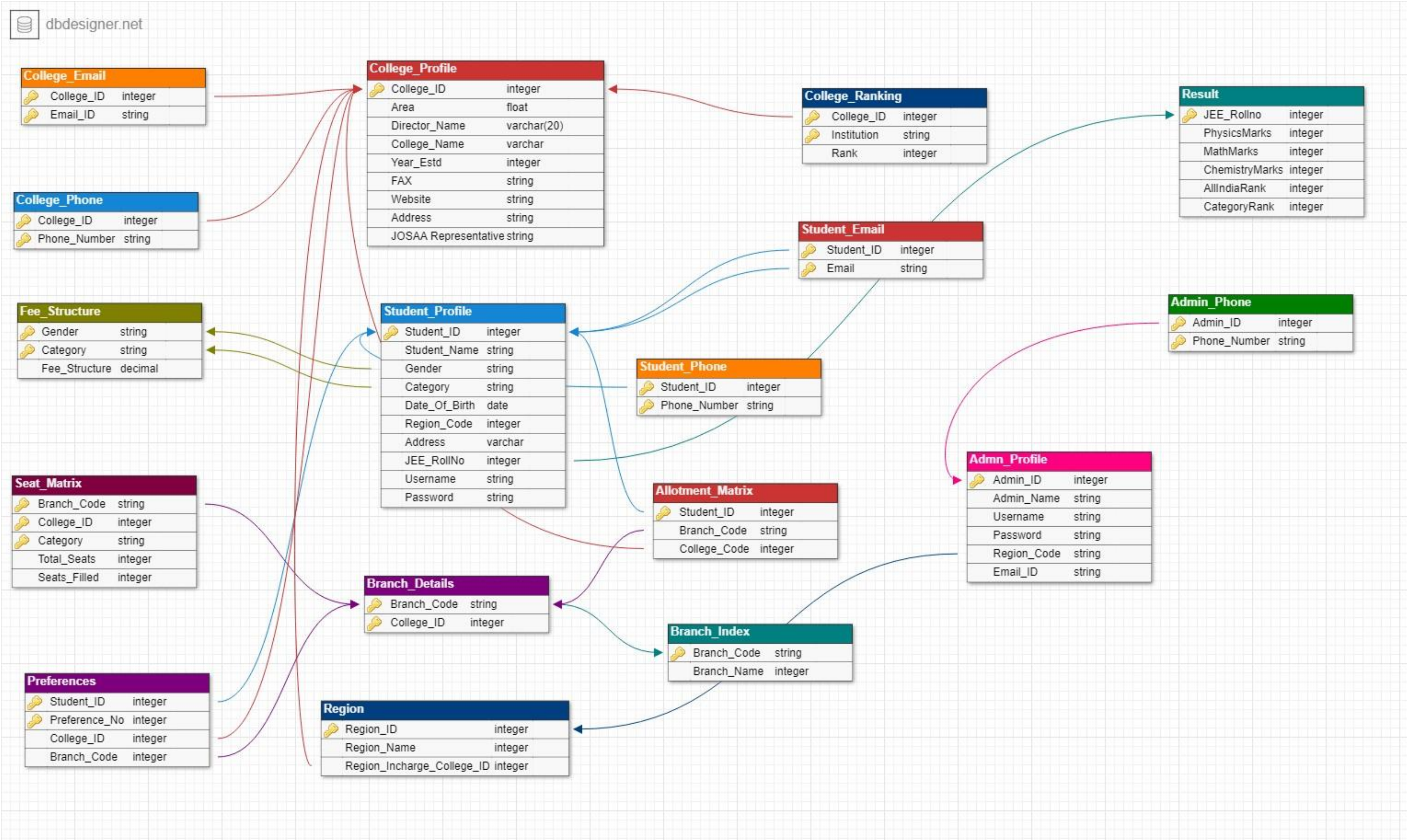
Also Admin_Profile and Admin_Login can be merged as :

- **Admin_Profile(All Attributes of Admin_Profile and Admin_Login)**
- **The PK of this relation will be Admin_ID clearly.**

Finally we have a relational schema with minimum redundancy.

This is an example where denormalization leads to better efficiency.

****Next page is the updated Relational schema along with the FD's** (Schema II)**



dbdesigner.net

College_Email	
🔑 College_ID	integer
🔑 Email_ID	string

College_Phone	
🔑 College_ID	integer
🔑 Phone_Number	string

Fee_Structure	
🔑 Gender	string
🔑 Category	string
Fee_Structure	decimal

Seat_Matrix	
🔑 Branch_Code	string
🔑 College_ID	integer
🔑 Category	string
Total_Seats	integer
Seats_Filled	integer

Preferences	
🔑 Student_ID	integer
🔑 Preference_No	integer
College_ID	integer
Branch_Code	integer

College_Profile	
🔑 College_ID	integer
Area	float
Director_Name	varchar(20)
College_Name	varchar
Year_Estd	integer
FAX	string
Website	string
Address	string
JOSAA Representative	string

Student_Profile	
🔑 Student_ID	integer
Student_Name	string
Gender	string
Category	string
Date_Of_Birth	date
Region_Code	integer
Address	varchar
JEE_RollNo	integer
Username	string
Password	string

Branch_Details	
🔑 Branch_Code	string
🔑 College_ID	integer

Region	
🔑 Region_ID	integer
Region_Name	integer
Region_Incharge_College_ID	integer

College_Ranking	
🔑 College_ID	integer
🔑 Institution	string
Rank	integer

Student_Email	
🔑 Student_ID	integer
🔑 Email	string

Student_Phone	
🔑 Student_ID	integer
🔑 Phone_Number	string

Allotment_Matrix	
🔑 Student_ID	integer
Branch_Code	string
College_Code	integer

Branch_Index	
🔑 Branch_Code	string
Branch_Name	integer

Result	
🔑 JEE_Rollno	integer
PhysicsMarks	integer
MathMarks	integer
ChemistryMarks	integer
AllIndiaRank	integer
CategoryRank	integer

Admin_Phone	
🔑 Admin_ID	integer
🔑 Phone_Number	string

Admn_Profile	
🔑 Admin_ID	integer
Admin_Name	string
Username	string
Password	string
Region_Code	string
Email_ID	string

RELATIONS IN THE DATABASE AFTER NORMALIZATION

1. **College_Profile (College_ID, Area, Director_Name, College_Name, Year_Estd, Region_Code,Websiite,Address,JOSAA_Representative,FAX)**

   FUNCTIONAL DEPENDENCIES:

   College_ID→Area

   College_ID→Director_Name

   College_ID→College_Name

   College_ID→Year_Estd

   College_ID→Region_Code

   College_ID->FAX

   College_ID->Website

   College_ID->JOSAA Representative

   College_ID->Address

2. **Student_Profile(Student_ID, Student_Name, Gender, Category, Date_of_Birth, Region_Code, Address, JEE_Roll_No,Username,Password)**

   FUNCTIONAL DEPENDENCIES:

   Student_ID→Student_Name
   Student_ID→Gender
   Student_ID→Category
   Student_ID→Date_of_Birth
   Student_ID→Region_Code
   Student_ID→Address
   Student_ID→JEE_Roll_No
   Student_ID->Username
   Student_ID->Password

3. **Region(Region_Code,Region_Name,Region_Incharge_College_ID)**

   FUNCTIONAL DEPENDENCIES:

   Region_Code→Region_Name
   Region_Code→ Region_Incharge_College_ID

4. **Allotment_Matrix(Student_ID,College_ID,Branch_Code,Status)**

   FUNCTIONAL DEPENDENCIES:

   Student_ID→College_ID
   Student_ID→Branch_Code
   Student_ID→Status

5. **College_Ranking(College_ID,Institution, Rank)**

   FUNCTIONAL DEPENDENCIES:

   {College_ID,Institution}→Rank

6. **Student_Email(Student_ID,Email)**

7. **Student_Phone(Student_ID,Phone_Number)**

8. **Admin_Profile(Admin_ID,Admin_Name,Region_Code,Username,Password)**

   FUNCTIONAL DEPENDENCIES:

   Admin_ID→Admin_Name
   Admin_ID→Region_Code
   Admin_ID->Username
   Admin_ID->Password

9. **Result(JEE_Roll_No, Math_Marks,Physics_Marks,Chemistry_Marks,All_India_Rank,Category_Rank)**

FUNCTIONAL DEPENDENCIES:

JEE_Roll_No→Math_Marks
JEE_Roll_No→Physics_Marks

JEE_Roll_No→Chemistry_Marks
JEE_Roll_No→All_India_Rank
JEE_Roll_No→Category_Rank

10. Admin_Phone(Admin_ID,Phone_Number)

11. College_Email(College_ID,Email_ID)

12. College_Phone(College_ID ,Phone_Number)

FUNCTIONAL DEPENDENCIES:

College_ID→Phone_Number

13. nch_Details(Branch_Code,College_ID)

14. Branch_Index(Branch_Code,Branch_Name)

FUNCTIONAL DEPENDENCIES

Branch_Code->Branch_Name

15. Seat_Matrix(Branch_Code,College_ID,Category,Total_Seats,Seats_Filled)

FUNCTIONAL DEPENDENCIES:

{Branch_Code,College_ID,Category}→Total_Seats

{Branch_Code,College_ID,Category}→Seats_Filled

16. Preferences(Preference_No,Student_ID,College_ID,Branch_Code)

FUNCTIONAL DEPENDENCIES:

{Preference_No,Student_ID}→College_ID

{Preference_No,Student_ID}→Branch_Code

17. Fee_Structure(Gender,Category,Registration_Fee)

FUNCTIONAL DEPENDENCIES:

{Gender,Category}→Registration_Fee
