Student System Management

Problem Statement:

- Sri Chaitanya college is now having a system for students. That deals with student details, Courses, parents, activities (payment details), reports, performance.
- Basic student details that are name, gender, contact, DOB, address.
- About the courses, course name, faculty, credits, fees, Student feedback, rating for faculty.
- About faculty name, id, course, faculty rating.
- Student have to the pay the fees according to courses. Once fees payment done Student keep the track of fees details with Admission number, student id, last transaction, time, date, upcoming transaction, status and same thing for faculty also.
- Faculty can also keep track of the salary that institute has paid till now.
- Students can check their report cards which contains the details of student id, course, subject grades, overall grade and current status.
- Parents can check student performance regularly: name, course, faculty feedback, rating of the student, improvements.

Entities for student management system:

1. Student(person):

2. Course(concept):

3. Faculty(person):

4. Payment_std(event):

5.Payment_faculty(event):

6. Report card(object):

7. Performance(concept):

Attributes for my entities:

Entity	Attribute	Type
Student (strong)	name, student_id, gender, dob, address, contact	composite Single Simple Composite Multivalued single

Course (strong)	course name, course_id, faculty, credits, fees, student feedback;	Single Single Single single multivalued
Faculty (strong)	name, id, course, rating,	Single Single simple Single
Payment_std (weak)	admission number, student id, last transaction, time, date, pay status, upcoming payments;	Single Single Single single composite simple Single
Payment faculty (weak)	Faculty_id, Last transaction, Date time, Pay status;	Single Single Single simple

Report card: (strong)	student name, student_id, course, overall grade, current status;	Single Single simple Single Single
Performance: (strong)	Student_id, name, course, faculty feedback, rating of the student, improvements.	Single Single Single Single multivalued

The relationships sets in my design are listed below:

std_cour: relating student with course.

std_facul: relating student with faculty.

std_pay: relating students with payments.

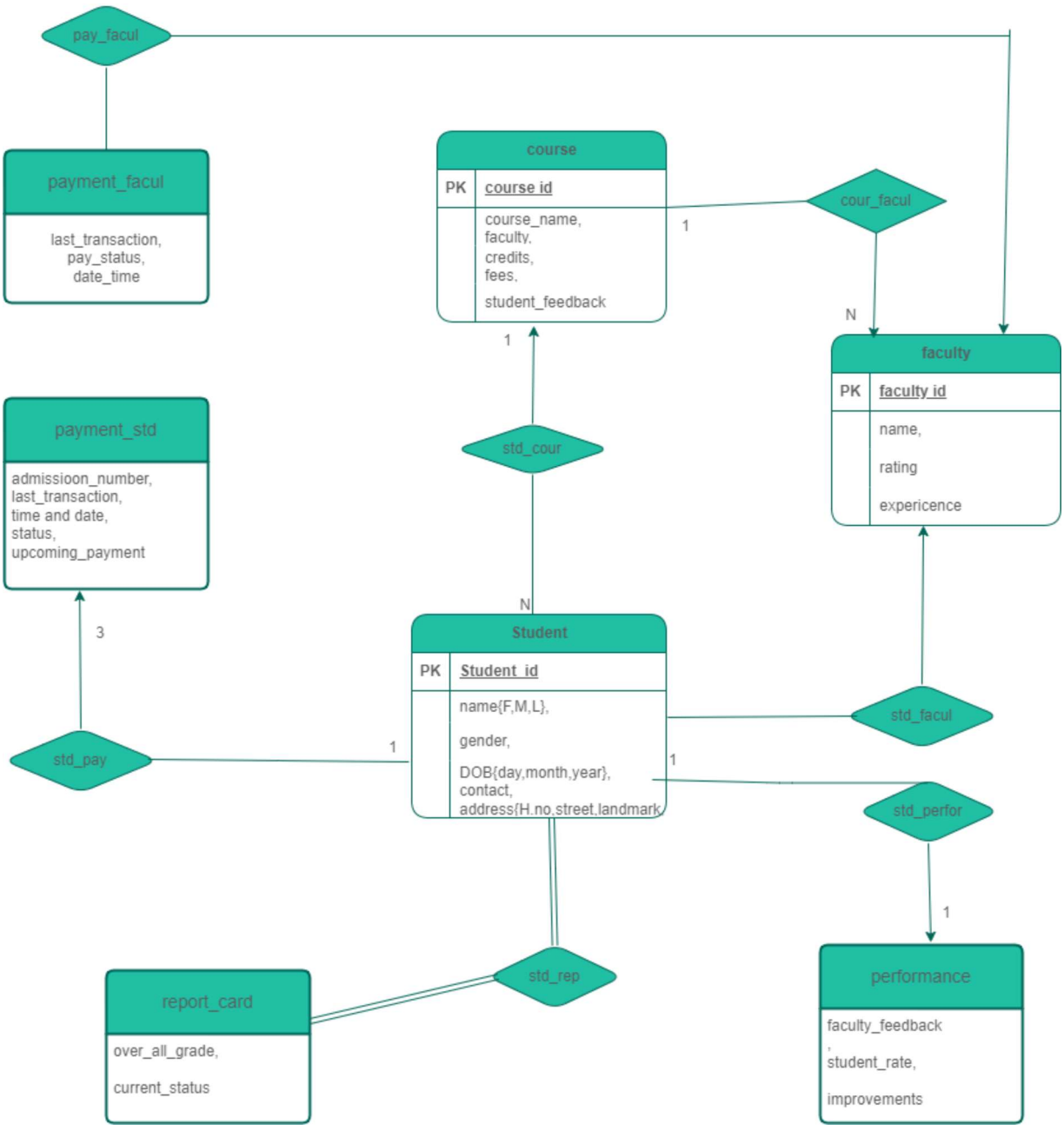
std_perfor: relating students with performance.

std_rep: relating student with report card.

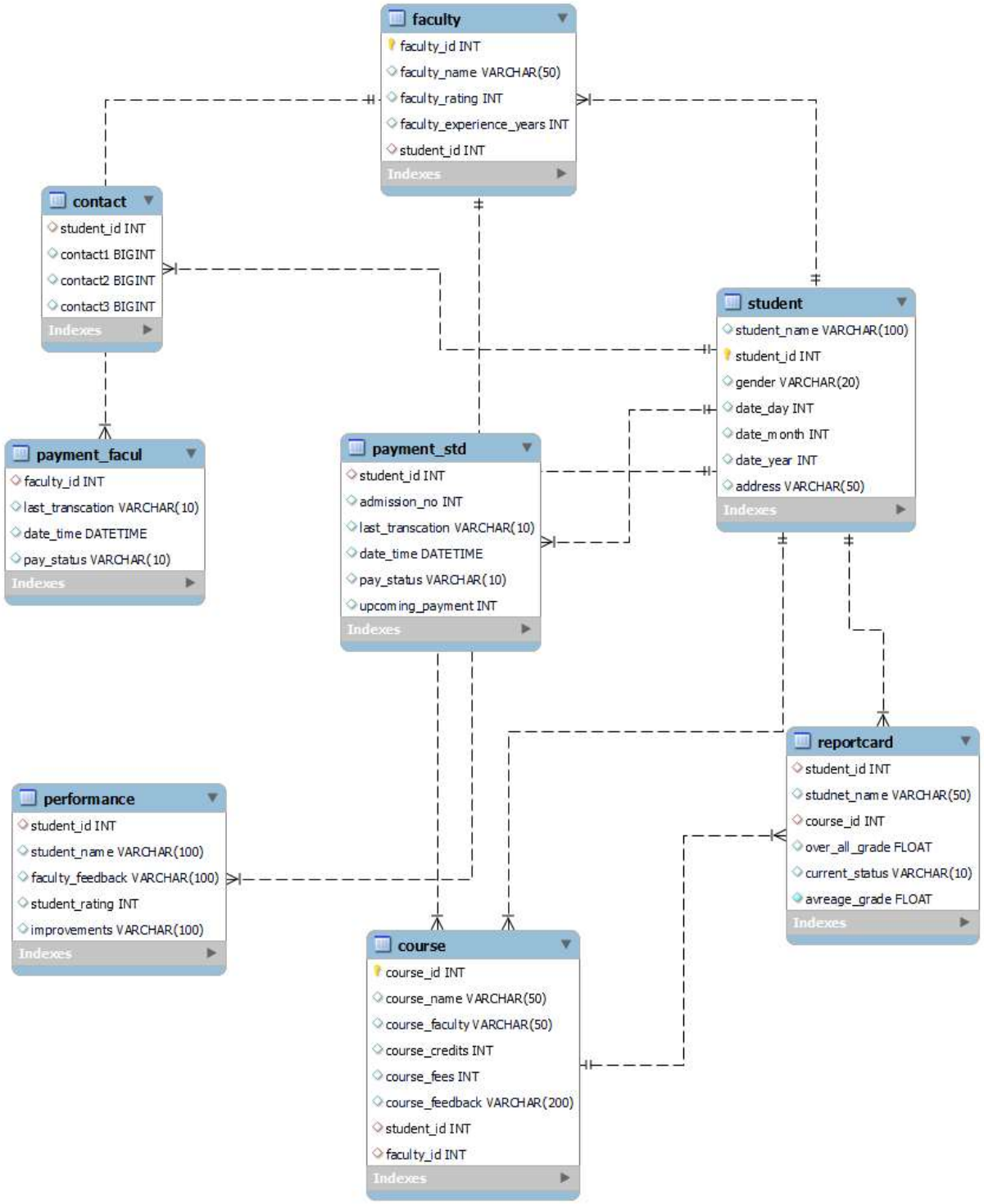
cour_facul: relating course with faculty.

pay_facul: relating faculty with payments.

ER Diagram For Student Management System



ER Diagram Provided By Mysql Workbench



RELATIONS/TABLES:

Student: (student_id, student_name, gender, date_day, date_month, date_year, address)

Course: (course_id, course_name, course_faculty, course credits, course fees, course_feedback)

Faculty: (faculty_id, faculty_name, faculty_rating, faculty_experience_years)

Payment_std: (admission_no, last_transaction, date_time, pay status, upcoming_payment)

Payment_faculty: (last_transaction, date_time, pay status)

Report card: (over_all_grade, current_status, average_grade)

Performance: (faculty_feedback, student_rating, improvement)

Functional Dependencies & Normalization:

Table selected to find normalization is the ”Faculty” relation.

faculty_id	faculty_name	student_id	faculty_experience_years	faculty_rating
214201	nand	201	12	5
214202	nick	202	5	4
214203	mick	203	8	4
214204	jack	204	15	3
214205	van	205	4	2
214206	vanes	206	6	4

Checking for 1NF:

- R={faculty_id,faculty_name,student_id,faculty_experience_years,faculty_rating}
- This relation is already in 1-NF form because there is no multivalued attribute(i.e.,they are atomic in nature).

Checking for 2NF:

Steps:

1. identify the prime and non-prime attributes
2. check if the candidate key is a composite key.
3. check if the non-prime attributes are fully dependent on the prime attributes.
4. check if the non-prime attribute is partially dependent on the prime attributes.

- faculty_id= A**(primary key)**, faculty_name= B, student_id= C**(foreign key)**, faculty_experience_years=D, faculty_rating=E
- The key for the given relation is (Guest_id, emp_id, service_id) where (emp_id, service_id) are foreign keys and guest_id is a primary key of relation room where three of them combine to form candidate key.
- The set of functional dependencies are f = {ac -> bde, ac -> d, ac->b, ac->e}
- {bde} may have redundant values so they cannot form the key.
- From the relation room prime attributes are {ac}
- From the relation non-prime attributes are{bde}
- From the relation non-prime attributes are{def}

Checking For Partial Dependency

{A -> B , A -> D, A-> E}

{C -> B , C -> D, C -> E}

The non-prime attributes {B, D, E} are partially dependent on the Part of the primary key.

2nf Decomposition

{A -> B , A -> D, A-> E}

{C -> B , C -> D, C -> E}

CONVERTING INTO 2NF

All the above 6 functional dependencies are creating problems. Because, There are non-prime attributes that are partially dependent on the candidate key.

So all these 6 functional dependencies are broken into 9 new different relations.

So The New 9 Relations are

1. (faculty_id, faculty_name)
2. (faculty_id, faculty_experience_years)
3. (faculty_id, faculty_rating)
4. (student_id, faculty_name)
5. (student_id, faculty_experience_years)
6. (student_id, faculty_rating)

3NF FORM: -

Already in 2NF

NO transitive Dependency

In the above relation there are no transitive dependency relations so the given relation is in 3nf form.