# STUDENT RECORD KEEPING SYSTEM DATABASE

Name: Jungsun Eoh

Student Number: 918590990

Github id: jungsun-eoh

| **Milestone/Version** | **Date** |
|---|---|
| Milestone 2 (vs2. M1) | 04/23/2021 |
| Milestone 1 - vs.1 | 03/9/2021 |

# Table of Contents

## Section 1: Project Description

Student Record-Keeping System is a record management design to keep track of current and former students' education records. Education records contain information about a student, such as administration information, course, major, finance (tuition and cost), health and immunization records, discipline reports, etc. The system is expected to have many various fields of data all connected to one student.

The essential purpose of a Student Record-Keeping System is managing its massive amount of variety of students' records and connecting the data with only authorized users. The data in the system should be separated into the specific field and department so that the access permission could be divided into the right purpose. For example, the finance department's employer may read the data of student's tuition and general information, but not health information. This access permission can be changed for a specific reason and season, such as lecturers input their students' grades. Authorized users who have permission to access one department can access only the department they are authorized for.

Unlike other record management designs, the Student Record-Keeping system does not change its data over time. Once the data is on the system, the data will stay. Instead, the system expects to be managed with secured access permission. The record should only allow access from authorized users to read the record of students and only allow access from authorized users to write the record when they have permission to do so. Because student records write their new data on a specific time window, the new Write permission should be always given out to users freshly, even if they were given permission before. Access permission is very important especially in the Student Record-Keeping System, the users should be divided into specific access hierarchies.

This project will cover various educational institutes to maintain records of students, the courses, and their administration information. The data is accessed by academic staff and administrators across the university by their permission status and range.

## Section 2: Use cases

| | |
|---|---|
| Use case title | Grade input |
| Actors | Sally(student employee-grader)<br>Grader, student employee, class section, class roster, assignment |
| Description | Sally is a new grader in English class. She will work as a student employee for this semester in the Eng101-1173 section class. Her name is officially up on the class roster as a grader. She could see there are already few students who already submitted their homework. |

| | |
|---|---|
| Use case title | Get permission to write the data. |
| Actors | Tom(lecturer)<br>lecturer, extend permission, staff, Faculty, professor, input grade with permission, class, department |
| Description | Tom is a lecturer at the University, and he was behind on his grading schedule. His grader didn't respond to his email or slack for 2 weeks. It is all teachers' nightmare. He realizes he won't make the final grade submission date for his class. The grading portal opens right after the final week of the semester, and it closed in few weeks. He could make some modifications when the portal is opened, but he will need another paperwork after the period for new inputs or modifications. He writes a letter to extend his permission on the grade to the staff of the department office. |

|  |  |
|---|---|
| Use case title | Give out write permission to authorized users. -> Instructors |
| Actors | Jim(office manager), Tom(lecturer), Kim(senior office manager)<br>Staff seniority, course, class number, subject, name, description, units, teachers |
| Description | Jim is an office manager who is in the charge of CS department. Upon the final week of the semester, he opens write permission to teachers, so they can input their students' grades. After a few weeks later, he got a mail from Tom that he couldn't finish his grades, and the portal is closed for him. He provided his course name. Jim could easily find Tom's name from course roasters; the course name and number are matched. Jim asks Kim to for extending permission for Tom. Kim allows Jim to extend the permission for Tom. |

|  |  |
|---|---|
| Use case title | Read data |
| Actors | Alice(advisor), Jinny(student)<br>Staff, advisor, academic records, course taken, grades, units, GPA, graduation status |
| Description | Alice is an academic advisor specialized in CS graduation preparation. She is helping her students ensure if they met graduation qualifications before they apply to graduate. Jinny is a senior student, she is about to graduate. She made an appointment with Alice to check her status. Alice reads Jinny's academic records, what courses she took, and their grades. Jinny is in good |

| | |
|---|---|
| | standing on her academics and ready to apply for graduation. |

| | |
|---|---|
| Use case title | Counselor seeing health records |
| Actors | Kevin(school counselor), Vicky(student)<br>Health service, counselor, medical records, immunization, note |
| Description | Kevin worked at university as a school counselor, he helped students to support their social and emotional needs. Today, Vicky made an appointment with Kevin. She is a senior student, and she is still struggling in the final year of her education and job hunting. Kevin could not see her grades, but he could see her Health service records in school and notes from last counseling. |

| | |
|---|---|
| Use case title | Range of permission. |
| Actors | Jenny (professor), Tony (professor)<br>Login, account, logout, class module |
| Description | Jenny and Tony are married who are both working at the same university. The good thing about having a spouse in the same field is they actually help one another. Tony finished his grading early and already posted the grades on the module, but Jenny is way behind her schedule. Tony decides to input the finished grade for her so that she could focus on grading. When he tries to input the data, the system keeps getting a permission error. After a few attempts, he realized he was logging in to his account to input Jenny's class. |

| | |
|---|---|
| | He had to log out from his account and log in to Jenny's to manipulate her class data. |

| | |
|---|---|
| Use case title | Student administration record |
| Actors | Timothy(new student)<br>prerequisite, administration, transcript, student number, name, address, education plan, student account, student resources(health service, finance, financial aid, academics) |
| Description | Timothy is a transfer student. When he applied for a transfer, he was overwhelmed a little with official documents he needed to provide to school; the transcript from the last school, general administration documentation(name, mailing address, etc.), and immunization. But he is glad now the papers are done with him, and he won't have to carry all his documents whenever he goes to staff. He is happy when he got his student ID. Now his record will be stored in the school permanently. His student id number only belongs to him. No one before and after him will get the same number with him. Now he is ready to enroll in his first class. |

| | |
|---|---|
| Use case title | Student union |
| Actors | Sarah(student)<br>Union, union membership, union event, event attendee. |

| | |
|---|---|
| Description | Sarah is a student, and she just noticed a table near the student center. The student unions are tabling for new events. As a third generation of hispanic family, she always wanted to know more about her roots and get an opportunity to celebrate it. She easily finds the desired union she would like to know more about, and they are having an event right now at school. She goes to the room that is holding the event. Before she gets in the room, she signs the attendance sheet. She is happy to see many faces just like her in the room. |

| | |
|---|---|
| Use case title | Student reviewing their academic progress |
| Actors | Kevin(student)<br>Transcript, academics |
| Description | Kevin is a junior student, and he is checking his academic progress. When he goes to his module, he finds academics easily. It says what he is majoring right now, GPA , and classes he took so far. He could check what class he took, what units it was, what grade he got, etc. He decides to print out the information, in order to do that, he needs to order the transcript. He thinks it's good idea to have it, so he could submit the transcription with his internship application. |

## Section 3: Database requirements

1) Student
    i) A student shall have a unique id.
    ii) A student shall have a unique student number.
    iii) A student shall have a name.
    iv) A student shall have a preferred name
    v) A student shall have a gender.
    vi) A student shall have a date of birth.
    vii) A student shall have ethnicity.
    viii) A student shall have only one status for graduation at a time.
    ix) A student shall have at least one address
    x) A student shall apply to at least one major program.
    xi) A student shall be able to create one and only student account using student id as login id.
    xii) A student shall hold multiple union membership.
    xiii) A student shall be hired as a student employee.
    xiv) A student shall have one and only one academic record.
2) Student Address
    i) A student address shall have a unique id.
    ii) A student address shall have at least one address.
    iii) A student address shall have a street.
    iv) A student address shall have a city.
    v) A student address shall have a state.
    vi) A student address shall have a zip code.
    vii) A student address shall belong to one student.

3) Student_Add
    i) A student_Add shall have student ID.
    ii) A student_Add shall have student address ID.
4) Student_Account
    i) A student account shall be available to one and only one student.
    ii) A student account shall have one unique id.
    iii) A student account shall have a student id.
    iv) A student account shall have a password.
    v) A student account shall have activated status.
    vi) A student account shall have a permission level.
5) Major program
    i) A major program shall have a unique id.
    ii) A major program shall have many options of major to choose at most one for a student.
    iii) A major program shall have a major id.
    iv) A major program shall have approved status..
    v) A major program shall be applied to one student.
6) Major
    i) A major shall have a unique id.
    ii) A major shall have one major name.
    iii) A major shall have a faculty member as head of the major.
    iv) A major shall have multiple courses
7) Courses
    i) Courses shall have a unique id
    ii) Courses shall have subject
    iii) courses shall have a major id.

iv) Courses shall have multiple class sesion

v) Courses shall have a required unit to finish the course.

8) Class section

i) A class section shall have unique id

ii) A class section shall have class number

iii) A class section shall have subject

iv) A class section shall have course number

v) A class section shall have class name

vi) A class section shall have instructor

vii) A class section shall have units

viii) A class section shall have prerequisites

ix) A class section shall have class description

x) A class section shall have course id.

xi) A class section shall have open status

xii) A class section shall have roster id.

9) Class roster student

i) A class roster shall have all the students in the class section.

ii) A class roster shall have student id

iii) A class roster shall have class section id

iv) A class roster shall have unique roster id.

10) Enroll class roster

i) An enroll class roster shall have roster id.

ii) An enroll class roster shall have student account id.

11) Student employee

i) A student employee shall have student id

ii) A student employee shall have department id.

12) TA/Grader

    i) A TA/Grader shall have student id

    ii) A TA/Grader shall have class session id

13) Student_employee_account

    i) Student employee account shall have unique account id

    ii) Student employee account shall have student id

    iii) Student employee account shall have permission level type id

14) Permission level

    i) Permission level shall have unique permission level id

    ii) Permission level shall notify whether permission level granted

    iii) Permission level shall have permission start date

    iv) Permission level shall have permission end date

    v) Permission level shall have permission level type

15) Permission level type

    i) Permission level type shall have unique permission type id

    ii) Permission level type shall have  permission level

    iii) Permission level type shall have  permission type description

16) Access student record

    i) Access student record shall have student records id.

    ii) Access student record shall have permission level id

17) Student records

    i) Student records shall have unique student records id

    ii) Student records shall have student records type id

    iii) Student records shall have student id

18) Student record type

    i) Student records type shall have unique Student records type id

ii) Student records type shall have permission level id.

19) Health Service

i) A health service shall have student id

ii) A health service shall have immunization record

iii) A health service shall have counselor note

iv) A health service shall be able to one student

20) Finance

i) A finance shall have student id

ii) A finance shall have balance

iii) A finance shall have charge

iv) A finance shall have payment

v) A finance shall have pending aid

vi) A finance shall be able to one student

21) Financial aid

i) A financial aid shall have student id

ii) A financial aid shall have balance

iii) A financial aid shall have active years

iv) A finance shall be able to one student

22) Academic records

i) A academics shall have student id

ii) A academics shall have major program id.

iii) A academics shall have a GPA.

iv) A academics shall have graduation status.

v) A academics shall have units taken.

vi) A academics shall  have courses taken.

vii) A academics shall be able to one and only student

23) Courses_taken

   i) A courses taken shall have a unique id.
   ii) A courses taken shall have class id.
   iii) A courses taken shall have grades
   iv) A courses taken shall have semester
   v) A courses taken shall have units

24) Transcripts

   i) A transcripts shall have student id
   ii) A transcripts shall have courses_taken_id
   iii) A transcripts shall have units
   iv) A transcripts shall have gpa.
   v) A transcripts shall be ordered by student multiple time

25) College

   i) A college shall have a unique id.
   ii) A college shall have one name.
   iii) A college shall have a faculty member as a head of department.

26) Staff

   i) A staff shall have a unique id.
   ii) A staff shall have a name
   iii) A staff shall have a gender
   iv) A staff shall have date of birth
   v) A staff shall have email
   vi) A staff shall have department id.
   vii) A staff shall have employee account id
   viii) A staff shall create one employee account.
   ix) A staff shall be belong to department

27) Employee_Account

   i) An Employee account shall have a unique id
   ii) An Employee account shall have employee name
   iii) An Employee account shall have account type
   iv) An Employee account shall have password
   v) An Employee account shall have activated status
   vi) An Employee account shall have permission type id
   vii) An employee account shall have one account type.
   viii) An employee account shall be granted one permission type.

28) Employee_Account_type

   i) An Employee account type shall have a unique id
   ii) An Employee account type shall have department id
   iii) An Employee account type shall have employee description

29) Emp work

   i) Emp work shall have department
   ii) Emp work shall have employee account id

30) Department

   i) A department shall have a unique id.
   ii) A department shall have one name.
   iii) A department shall be only in one college.
   iv) A department shall have a faculty member as a head of department.
   v) A department shall belong to one college

31) Faculty

   i) A faculty shall have a unique id.
   ii) A faculty shall have name
   iii) A faculty shall have gender

iv) A faculty shall have Date of birth
v) A faculty shall have email
vi) A faculty shall have college id
vii) A faculty shall have employee account id
viii) A faculty shall create one employee account.
ix) A faculty shall be belong to department

32) faculty_Account
i) A faculty account shall have a unique id
ii) A faculty account shall have employee name
iii) A faculty account shall have account type
iv) A faculty account shall have password
v) A faculty account shall have activated status
vi) A faculty account shall have permission type id
vii) A faculty account shall have one account type.
viii) A faculty account shall be granted one permission type.

33) faculty_Account_type
i) A faculty account type shall have a unique id
ii) A faculty account type shall have college id
iii) A faculty account type shall have employee description

34) Account permission granted
i) Account permission granted shall have permission level id
ii) Account permission granted shall have faculty id

## Section 4: Detailed List of Main Entities. Attributes

1) Student(Strong)
   - student_id: key, numeric
   - Name: alphanumeric, composite
     1) First
     2) Last
   - Gender: alphanumeric
   - Ethnicity: alphanumeric
   - addmission_number: numeric
2) EnrolledStudent(Strong)
   - Enrolled_student_id: key, numeric
   - Prefer_name: alphanumeric
   - Date_of_Birth: datetime
3) Student_Address(strong)
   - Address_ID: key, numeric
   - address: alphanumeric, multi-value, composite
     1) Street
     2) City
     3) State
     4) Zipcode
   - Phone: numeric, multi-value, composite
   - Email: alphanumeric
4) Student_Add_Address(weak)
   - addAddress_id: key numeric
   - ernd_Student: weak key, numeric

- Studen_ addresst: weak key, numeric

5) Student_Account(Weak)
    - student_account_ID: numeric, key
    - erd_student: numeric, weak key
    - Activate: boolean
    - Created: numeric, date
    - permission_level: weakkey, numeric
6) Major_program(strong)
    - Major_program_ID: numeric, key
    - Major_ID: weak key, numeric
    - name: alphanumeric
    - advisor: alphanumeric
7) Major(Strong)
    - Major_ID: numeric, key
    - Major name: alphanumeric
    - Head_faculty: alphanumeric,
    - College: weak key
8) Courses(Strong)
    - Course_ID: numeric, key
    - Subject: alphanumeric (CSC)
    - Major: numeric, weak key
9) Class section(Strong)
    - Section_ID: numeric, key
    - Class_number: alphanumeric (11785)
    - Course: weak key, numeric (675)
    - Instructor: weak key

- Units: numeric
- Prerequisites: alphanumeric
- Description: alphanumeric
- Open: Boolean

10) EnrollSection(class) (Weak)
    - enrollid:numeric, key
    - Student_account: numeric, weak key
    - Roster: numeric, weak key
11) class_roster(weak)
    - rosterid : key, numeric
    - class_section: weak key, numeric
    - semester: alphanumeric
12) Student_employee (weak)
    - Student_emp_id: key numeric
    - Student_Acc: numeric, weak key
    - Department: numeric, weak key
    - Salary: numeric
13) TA/Grader (Weak)
    - TA/Grader_id: key numeric
    - Student_Acc: numeric, weak key
    - Class_session: numeric, weak key
    - Salary: numeric
14) Student_employee_account(strong)
    - Student_employee_account_ID: numeric key
    - Created: numeric
    - Activated: numeric

- Student_emp: weak key

15) Permission_level(Strong)
    - Permission level ID: key numeric
    - Permission level granted: numeric
    - Permission_start: multivalue, timestamp
    - Permission_end: multivalue, timestamp
    - permission _level_type: weak key, numeric
16) Permission_level_type(Strong)
    - Permission level type ID: key, numeric
    - Permission level type description: alphanumeric
17) Student Records(Strong)
    - Student Records ID: key, numeric
    - Student Records type: weak key, numeric
    - Student account: weak key, numeric
18) Student Record type(Strong)
    - Student Records type ID: key, numeric
    - Records desc: alphanumeric
19) Finance(Strong)
    - Student_ID: numeric, weak key
    - Balance: numeric, derived
    - Charge: numeric
    - Payment: numeric
    - Pending_Aid: numeric
    - Student Records: weak key, numeric
20) Financial_Aid(Strong)
    - Student_ID: numeric, weak key

- Balance: numeric
- Years: alphanumeric
- Student Records: weak key, numeric

21) Health Service(Strong)
    - Student_ID: numeric, weak key
    - Immunization: alphanumeric
    - Counselor_Note: alphanumeric
    - Student Records: weak key, numeric

22) Academic Records(Strong)
    - academics_ID: numeric, key
    - Major Program: numeric, weak key
    - Student_acc: numeric, weak key
    - Student Records: weak key, numeric
    - GPA: numeric
    - Graduation_status: boolean
    - Units_taken: numeric

23) Courses_taken(weak)
    - Course_taken_ID: numeric, key
    - Grades: numeric
    - Semester: alphanumeric
    - Units: numeric
    - Academic records: numeric, weak key
    - transcripts: weak key, numeric
    - Section: numeric weak key

24) Transcripts(Weak)
    - Transcript ID: key, numeric

- Academic records: numeric, weak key
- ordered: date

25) College(Strong)
    - College_ID: numeric, key
    - College_Head: alphanumeric
    - Name: alphanumeric

26) employee(staff)(Strong)
    - staff_ID: key, numeric
    - Name: alphanumeric, composite
        1) First
        2) Last
    - Gender: alphanumeric
    - Date_of_Birth: date, composite
        1) Year
        2) Month
        3) Day
    - Email: alphanumeric

27) Employee_Account (Strong)
    - Employee_Account_ID: numeric, key
    - account_Type: numeric, weak key
    - Activate: boolean
    - Permission_level: numeric, weak key
    - Employee : numeric weak key

28) employee_Account_type(weak)
    - employee_Account_Type_ID: numeric, key
    - employee_type_description: alphanumeric

29) Department(Strong)

- Department_id: numeric
- Name: alphanumeric
- Head_ID: numeric, weak key

30) Faculty(Strong)

- Faculty_ID: key, numeric
- Name: alphanumeric, composite
    1) First
    2) Last
- Gender: alphanumeric
- Date_of_Birth: date, composite
    1) Year
    2) Month
    3) Day
- Email: alphanumeric

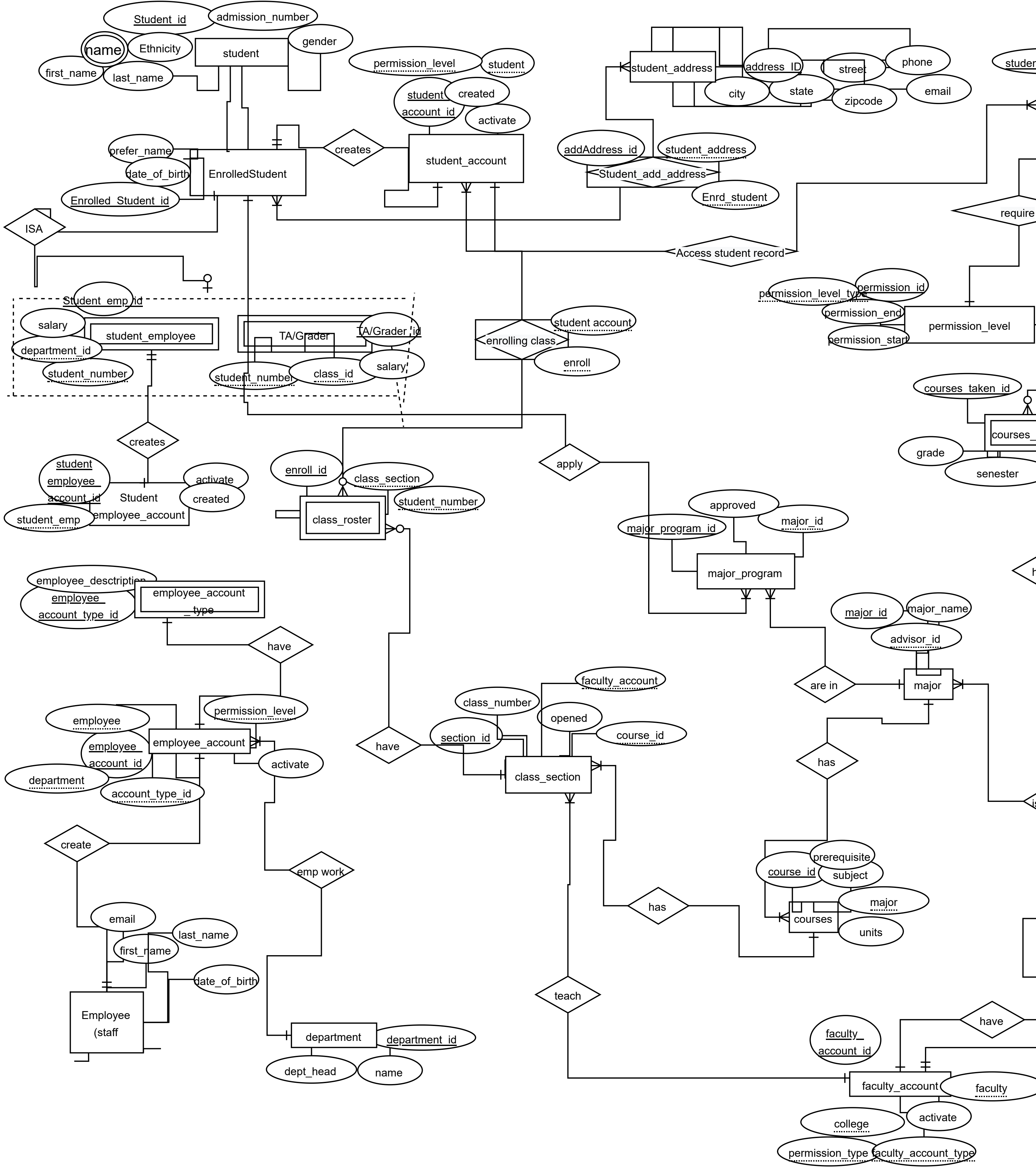
31) faculty_Account (Strong)

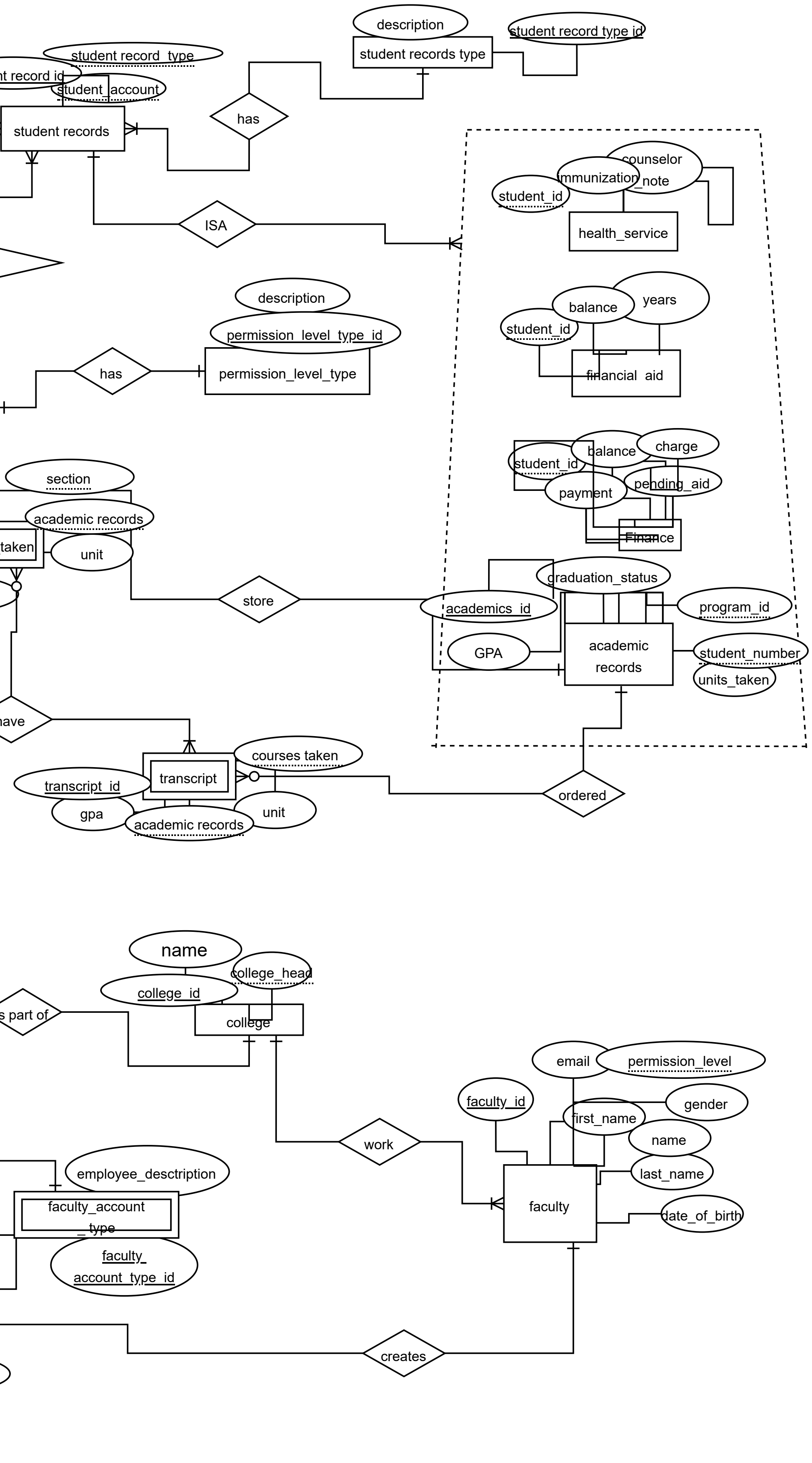
- faculty_Account_ID: numeric, key
- faculty_account_Type: numeric, weak key
- Activate: boolean
- Permission_level: numeric, weak key
- College: numeric, weak key
- faculty:numeric, weak key

32) faculty_Account_type(weak)

- faculty_Account_Type_ID: numeric, key
- Faculty_type description: alphanumeric

## Section 5: Entity-relationship diagram


Student_id
admission_number
name
Ethnicity
student
gender
first_name
last_name
permission_level
student
student_address
address_ID
street
phone
city
state
zipcode
email
studen
student
account_id
created
activate
prefer_name
EnrolledStudent
creates
student_account
addAddress_id
student_address
Student_add_address
date_of_birth
Enrolled_Student_id
Enrd_student
require
ISA
Access student record
permission_id
permission_level_type
permission_end
permission_level
permission_start
Student_emp_id
salary
student_employee
TA/Grader
TA/Grader_id
student account
enrolling class
enroll
department_id
student_number
student_number
class_id
salary
courses_taken_id
courses_
creates
grade
senester
student
employee_
account_id
Student
employee_account
activate
created
student_emp
enroll_id
class_section
student_number
class_roster
apply
approved
major_program_id
major_id
major_program
employee_description
employee_
account_type_id
employee_account
_type
have
major_id
major_name
advisor_id
are in
major
faculty_account
class_number
opened
employee
permission_level
employee_
account_id
employee_account
activate
section_id
course_id
have
class_section
has
department
account_type_id
create
emp work
course_id
prerequisite
subject
major
courses
units
has
email
last_name
first_name
date_of_birth
Employee
(staff
teach
have
faculty_
account_id
department
department_id
dept_head
name
faculty_account
faculty
activate
college
permission_type
faculty_account_type


description
student records type
student record type id
student record_type
nt record id
student_account
student records
has
ISA
counselor
immunization
_note
student_id
health_service
description
permission_level_type_id
has
permission_level_type
balance
years
student_id
financial aid
balance
charge
student_id
payment
pending_aid
Finance
section
academic records
taken
unit
store
graduation_status
academics_id
program_id
GPA
academic records
student_number
units_taken
have
courses taken
transcript
transcript_id
gpa
unit
academic records
ordered
name
college_head
college_id
s part of
college
email
permission_level
faculty_id
gender
first_name
name
work
last_name
employee_desctription
faculty
faculty_account _type
date_of_birth
faculty_ account_type_id
creates

## Section 6: Testing Table

| | Entity A | relation | Entity B | cardinality | | |
|---|---|---|---|---|---|---|
| 1 | student | apply | Financial aid | 1-to-many | fail | Student might not apply financial aid |
| 2 | student | apply | Financial aid | 0-to-many | pass | |
| 3 | student | have | student_address | 1-to-1 | fail | Student can have multiple address |
| 4 | student | have | student_address | 1-to-many | pass | |
| 5 | student | create | student_account | 1-to-1 | fail | Student can create only one account, and it will store in the DB permanently. No one can use same one. |
| 6 | student | create | student_account | Only and only one | pass | |
| 7 | Student | have | academics | 1-to-1 | fail | Student can have only one academics, and it will store in the DB permanently. |
| 8 | Student | have | academics | Only and | pass | |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | | only one | | |
| 9 | course | have | class_section | 1-to-many | fail | Course exist without any class. |
| 10 | course | have | class_section | 1-to-many(zero) | pass | Now the course itself will exist even though school stop carrying specific classes. |
| 11 | union | have | union_membership | many-to-many | fail | Union membership can not be strong entity |
| 12 | union | have | union_membership | 1-to-many | pass | |
| 13 | union | organize | union event | many-to-many | fail | Union might not organize any event at all |
| 14 | union | organize | union event | many-to-many(zero) | pass | |
| | | | | | | |

M1 - ver.2

| | Entity A | relation | Entity B | cardinality | | |
|---|---|---|---|---|---|---|
| 15 | Student account | Enrolling class | Enrolled student | many-to-one | fail | There might be class no student enrolled yet. |
| 16 | Student | Enrollin | Enrolled | many(zero | pass | |

| | | | | | | |
|---|---|---|---|---|---|---|
| | account | g class | student | )-to-many( zero) | | |
| 17 | Enrolled student | | | | fail | It should be change it to weak entity |
| 18 | Student account | | | | fail | It should be change it to weak entity |
| 19 | | | | | pass | |
| 20 | Student account | granted | Permission level | 1-to-1 | fail | Student account is weak entity, |
| 21 | Student account | granted | Permission level | 1(zero)-to-1 | pass | |
| 22 | student | creates | Student account | 1-to-1 | fail | Student account is weak entity, |
| 23 | student | creates | Student account | 1-to-1 | pass | |
| 24 | Class section | | courses_take n | | fail | Course taken should connected to class section |
| 25 | Class section | isRecor dedTo | courses_take n | many-to-m any | pass | |
| 26 | college | | faculty | | fail | College and faculty should be |

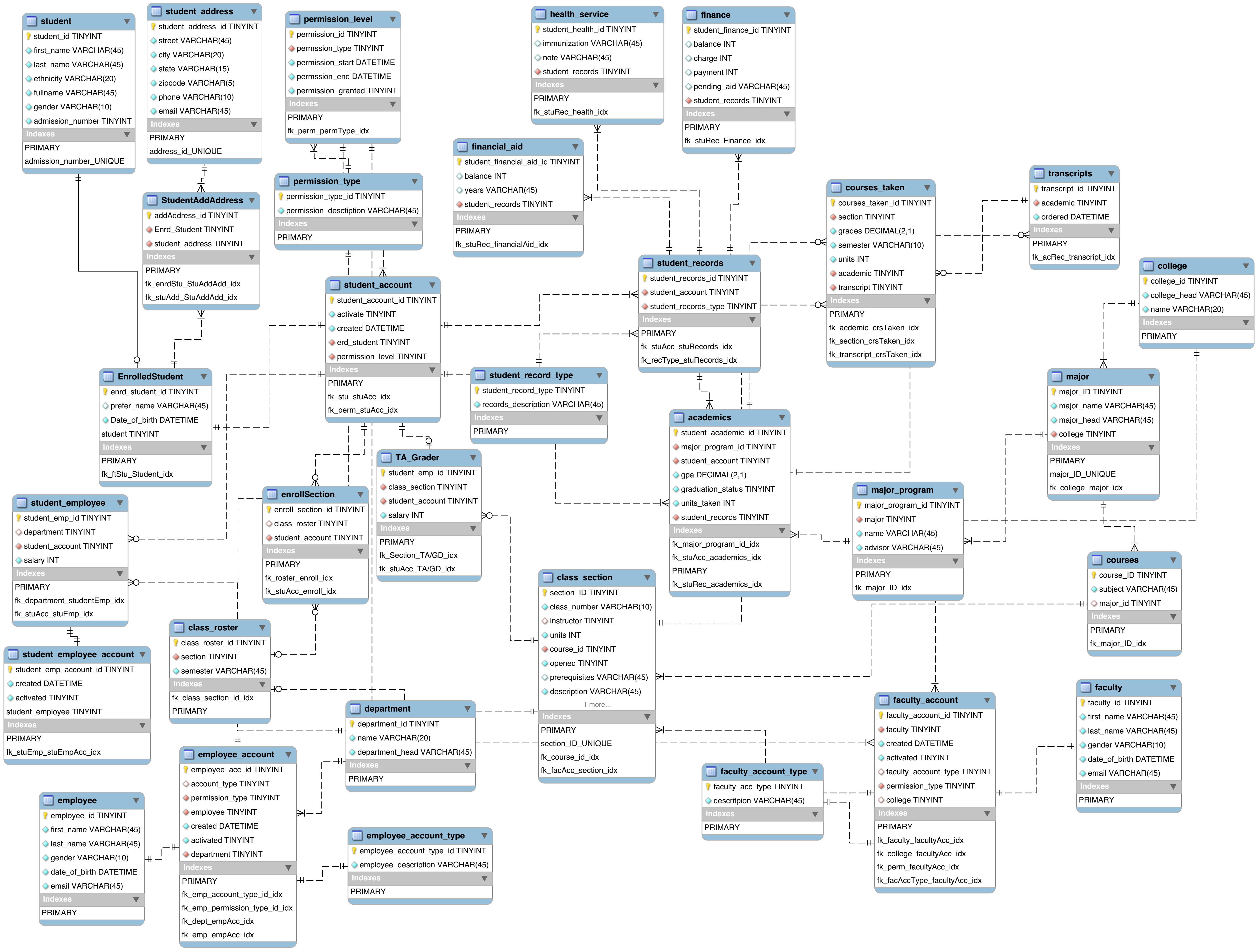
| | | | | | | |
|---|---|---|---|---|---|---|
| | | | | | | connected each other. |
| 27 | college | work | faculty | Many-to-many | pass | |
| 28 | student_employee | creates | Student_employee_account | Only one-to-one | fail | Student employee, TA/Grader entities should be weak. |
| 29 | student_employee | creates | Student_employee_account | one(zero)-to-one | pass | |
| | | | | | | |

## Section 7: Database Model/EER

| Table | Fk | On update | On delete | comment |
|---|---|---|---|---|
| academics | majorprogram | No action | No action | Even Though the major program has changed over time, the student record should keep the original. |
| academics | student Account | cascade | restrict | If a student account should not be deleted from the academic table, but if the account is changed, the record should move along. |
| academics | student Record | cascade | restrict | Student records should not be allowed to be deleted. |
| Class roster | section | cascade | cascade | If a class section is deleted, class is cancel, so the roster should be deleted too. |
| Class section | course | cascade | cascade | If the course(csc675) is cancelled, it means all sections(csc675-1, csc675-2) are cancelled. |
| Class section | Faculty account | cascade | Set null | If the faculty rescinds from class, the instructor should remain empty until finding a new instructor. |
| courses | major | cascade | cascade | If a school decides to no longer teach a certain major, all classes in the major are cancelled. |
| Course taken | acadenic | cascade | restrict | Academic records keep all the student academic records. These records should not be allowed to delete. |
| Course taken | section | No action | No action | Even Though the section has changed over time, the course taken should keep the original. |
| Course taken | transcript | No action | No action | The course taken should keep the original. |
| Employee account | Account type | cascade | Set null | Account type can be changed over time, but it does not mean fire every certain type of employee. They will be assigned into new roles. |
| Employee account | Permission type | cascade | cascade | Permission type can be changed, and it should be move along with account |
| Employee | departm | cascade | cascade | department can be changed, and it should be |

| | | | | |
|---|---|---|---|---|
| account | ent | | | move along with account |
| Employee account | employ ee | cascade | No action | Even though the employee leave their job, Employee account records will be stored for future reference. |
| Enrolled student | student | cascade | No action | Even though the student leave school, their records will be stored for future reference. |
| Enroll section | roster | cascade | cascade | If a roster is deleted, enroll section no longer exists. |
| Enroll section | Student account | cascade | cascade | If student account is deleted, enroll section no longer exists. |
| Faculty account | faculty | cascade | No action | Even though the faculty leave their job, Employee account records will be stored for future reference. |
| Faculty account | college | cascade | Set null | college can be changed over time, but employee record should be stored. |
| Faculty account | Permiss ion type | cascade | cascade | Permission type can be changed, and it should be move along with account |
| Faculty account | Faculty account type | cascade | Set null | Account type can be changed over time, but it does not mean fire every certain type of employee. They will be assigned into new roles. |
| finance | Student records | cascade | No action | Student records should not be deleted. |
| Financial aid | Student records | cascade | No action | Student records should not be deleted. |
| Health service | Student records | cascade | No action | Student records should not be deleted. |
| major | college | cascade | cascade | If a school decides to no longer have a certain college, all majors in the college are cancelled. |
| Major program | major | cascade | cascade | If a school decides to no longer have a certain major, all major program in the major are cancelled. |
| Permissio n level | Permiss ion type | cascade | cascade | Permission type can be changed due to different needs, it permission level table should follow the changes. |
| Student | Enrolled | cascade | No | Even though the student leave school, their |

| | | | | |
|---|---|---|---|---|
| account | student | | action | records will be stored for future reference. |
| Student account | Permission level | cascade | cascade | Permission level can be changed, it should move along with changes. |
| Student add address | Student address | cascade | cascade | If a student address is deleted, student add address no longer exists. |
| Student add address | Enrd student | cascade | cascade | If a enrd student is deleted, student add address no longer exists. |
| Student employee | department | cascade | Set null | Student employee will reassign to new department. |
| Student employee | Student account | cascade | No action | Store it for future reference. |
| Student employee account | Student employee | cascade | No action | Store it for future reference. |
| Student records | Student account | cascade | No action | Student records should never be deleted. |
| Student records | Student records type | cascade | No action | Student records should never be deleted. |
| ta/grader | section | cascade | cascade | If a section is canceled, grader no longer needed. |
| ta/grader | Student account | cascade | No action | Store it for future reference. |
| transcripts | academic | No action | No action | Transcript already printed should not be changed over time. |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |


student
student_id TINYINT
first_name VARCHAR(45)
last_name VARCHAR(45)
ethnicity VARCHAR(20)
fullname VARCHAR(45)
gender VARCHAR(10)
admission_number TINYINT
Indexes
PRIMARY
admission_number_UNIQUE
student_address
student_address_id TINYINT
street VARCHAR(45)
city VARCHAR(20)
state VARCHAR(15)
zipcode VARCHAR(5)
phone VARCHAR(10)
email VARCHAR(45)
Indexes
PRIMARY
address_id_UNIQUE
permission_level
permission_id TINYINT
permssion_type TINYINT
permission_start DATETIME
permssion_end DATETIME
permission_granted TINYINT
Indexes
PRIMARY
fk_perm_permType_idx
health_service
student_health_id TINYINT
immunization VARCHAR(45)
note VARCHAR(45)
student_records TINYINT
Indexes
PRIMARY
fk_stuRec_health_idx
finance
student_finance_id TINYINT
balance INT
charge INT
payment INT
pending_aid VARCHAR(45)
student_records TINYINT
Indexes
PRIMARY
fk_stuRec_Finance_idx
financial_aid
student_financial_aid_id TINYINT
balance INT
years VARCHAR(45)
student_records TINYINT
Indexes
PRIMARY
fk_stuRec_financialAid_idx
permission_type
permission_type_id TINYINT
permission_desctiption VARCHAR(45)
Indexes
PRIMARY
StudentAddAddress
addAddress_id TINYINT
Enrd_Student TINYINT
student_address TINYINT
Indexes
PRIMARY
fk_enrdStu_StuAddAdd_idx
fk_stuAdd_StuAddAdd_idx
transcripts
transcript_id TINYINT
academic TINYINT
ordered DATETIME
Indexes
PRIMARY
fk_acRec_transcript_idx
courses_taken
courses_taken_id TINYINT
section TINYINT
grades DECIMAL(2,1)
semester VARCHAR(10)
units INT
academic TINYINT
transcript TINYINT
Indexes
PRIMARY
fk_acdemic_crsTaken_idx
fk_section_crsTaken_idx
fk_transcript_crsTaken_idx
student_records
student_records_id TINYINT
student_account TINYINT
student_records_type TINYINT
Indexes
PRIMARY
fk_stuAcc_stuRecords_idx
fk_recType_stuRecords_idx
college
college_id TINYINT
college_head VARCHAR(45)
name VARCHAR(20)
Indexes
PRIMARY
student_account
student_account_id TINYINT
activate TINYINT
created DATETIME
erd_student TINYINT
permission_level TINYINT
Indexes
PRIMARY
fk_stu_stuAcc_idx
fk_perm_stuAcc_idx
EnrolledStudent
enrd_student_id TINYINT
prefer_name VARCHAR(45)
Date_of_birth DATETIME
student TINYINT
Indexes
PRIMARY
fk_ftStu_Student_idx
student_record_type
student_record_type TINYINT
records_description VARCHAR(45)
Indexes
PRIMARY
major
major_ID TINYINT
major_name VARCHAR(45)
major_head VARCHAR(45)
college TINYINT
Indexes
PRIMARY
major_ID_UNIQUE
fk_college_major_idx
academics
student_academic_id TINYINT
major_program_id TINYINT
student_account TINYINT
gpa DECIMAL(2,1)
graduation_status TINYINT
units_taken INT
student_records TINYINT
Indexes
fk_major_program_id_idx
fk_stuAcc_academics_idx
PRIMARY
fk_stuRec_academics_idx
TA_Grader
student_emp_id TINYINT
class_section TINYINT
student_account TINYINT
salary INT
Indexes
PRIMARY
fk_Section_TA/GD_idx
fk_stuAcc_TA/GD_idx
major_program
major_program_id TINYINT
major TINYINT
name VARCHAR(45)
advisor VARCHAR(45)
Indexes
PRIMARY
fk_major_ID_idx
student_employee
student_emp_id TINYINT
department TINYINT
student_account TINYINT
salary INT
Indexes
PRIMARY
fk_department_studentEmp_idx
fk_stuAcc_stuEmp_idx
enrollSection
enroll_section_id TINYINT
class_roster TINYINT
student_account TINYINT
Indexes
PRIMARY
fk_roster_enroll_idx
fk_stuAcc_enroll_idx
class_section
section_ID TINYINT
class_number VARCHAR(10)
instructor TINYINT
units INT
course_id TINYINT
opened TINYINT
prerequisites VARCHAR(45)
description VARCHAR(45)
1 more...
Indexes
PRIMARY
section_ID_UNIQUE
fk_course_id_idx
fk_facAcc_section_idx
courses
course_ID TINYINT
subject VARCHAR(45)
major_id TINYINT
Indexes
PRIMARY
fk_major_ID_idx
class_roster
class_roster_id TINYINT
section TINYINT
semester VARCHAR(45)
Indexes
fk_class_section_id_idx
PRIMARY
student_employee_account
student_emp_account_id TINYINT
created DATETIME
activated TINYINT
student_employee TINYINT
Indexes
PRIMARY
fk_stuEmp_stuEmpAcc_idx
department
department_id TINYINT
name VARCHAR(20)
department_head VARCHAR(45)
Indexes
PRIMARY
faculty
faculty_id TINYINT
first_name VARCHAR(45)
last_name VARCHAR(45)
gender VARCHAR(10)
date_of_birth DATETIME
email VARCHAR(45)
Indexes
PRIMARY
faculty_account
faculty_account_id TINYINT
faculty TINYINT
created DATETIME
activated TINYINT
faculty_account_type TINYINT
permission_type TINYINT
college TINYINT
Indexes
PRIMARY
fk_faculty_facultyAcc_idx
fk_college_facultyAcc_idx
fk_perm_facultyAcc_idx
fk_facAccType_facultyAcc_idx
employee_account
employee_acc_id TINYINT
account_type TINYINT
permission_type TINYINT
employee TINYINT
created DATETIME
activated TINYINT
department TINYINT
Indexes
PRIMARY
fk_emp_account_type_id_idx
fk_emp_permission_type_id_idx
fk_dept_empAcc_idx
fk_emp_empAcc_idx
faculty_account_type
faculty_acc_type TINYINT
descritpion VARCHAR(45)
Indexes
PRIMARY
employee
employee_id TINYINT
first_name VARCHAR(45)
last_name VARCHAR(45)
gender VARCHAR(10)
date_of_birth DATETIME
email VARCHAR(45)
Indexes
PRIMARY
employee_account_type
employee_account_type_id TINYINT
employee_description VARCHAR(45)
Indexes
PRIMARY

## Section 8: forward Engineering

## Section 9: inserting Data

## Section 10: Testing

## Section 11: Testing Table

| Entity | SQLQuery | pass/fail | Error description | solution |
|---|---|---|---|---|
| academics | UPDATE | pass | none | none |
| academics | DELETE | pass | none | none |
| Class roster | UPDATE | pass | none | none |
| Class roster | DELETE | pass | none | none |
| class_section | UPDATE | FAIL | CONSTRAINT issue | Making Changes in foreign key |
| class_section | DELETE | fail | CONSTRAINT issue | Making Changes in foreign key |
| College | UPDATE | pass | none | none |
| College | DELETE | fail | CONSTRAINT issue | Making Changes in foreign key |
| courses | UPDATE | pass | none | none |
| courses | DELETE | pass | none | none |
| courses_taken | UPDATE | pass | none | none |
| courses_taken | DELETE | pass | none | none |
| department | UPDATE | pass | none | none |
| department | DELETE | pass | none | none |
| employee | UPDATE | pass | none | none |
| employee | DELETE | pass | none | none |
| employee_account | UPDATE | pass | none | none |
| employee_acco | DELETE | pass | none | none |

| | | | | |
|---|---|---|---|---|
| unt | | | | |
| employee_account_type | UPDATE | pass | none | none |
| employee_account_type | DELETE | pass | none | none |
| EnrolledStudent | UPDATE | pass | none | none |
| EnrolledStudent | UPDATE | FAIL | Shouldn't update the fk student id, but it updated. | Making Changes in foreign key |
| EnrolledStudent | DELETE | fail | CONSTRAINT issue | Making Changes in foreign key |
| enrollSection | UPDATE | FAIL | Shouldn't update the fk, but it updated. | Making Changes in foreign key |
| enrollSection | DELETE | FAIL | Shouldn't DELETE the fk, but it DELTED. | Making Changes in foreign key |
| faculty | UPDATE | pass | none | none |
| faculty | DELETE | pass | Shouldn't DELETE the fk, and it didnt delete it, but I wonder if it is how the no response works. | Making Changes in foreign key |
| faculty_account | UPDATE | pass | none | none |
| faculty_account | DELETE | FAIL | Shouldn't DELETE the fk, but it DELETED. | Making Changes in foreign key |
| faculty_account_type | UPDATE | pass | none | none |
| faculty_account_type | DELETE | pass | none | none |

| | | | | |
|---|---|---|---|---|
| finance | UPDATE | pass | none | none |
| finance | DELETE | FAIL | Shouldn’t DELETE the fk, but it DELETED. | Making Changes in foreign key |
| financial_aid | UPDATE | pass | none | none |
| financial_aid | DELETE | FAIL | Shouldn’t DELETE the fk, but it DELETED. | Making Changes in foreign key |
| health_service | UPDATE | pass | none | none |
| health_service | DELETE | FAIL | Shouldn’t DELETE the fk, but it DELETED. | Making Changes in foreign key |
| major | UPDATE | pass | none | none |
| major | DELETE | pass | none | none |
| major_program | UPDATE \| | fail | | none |
| major_program | DELETE | FAIL | CONSTRAINT issue | Making Changes in foreign key |
| permission_level | UPDATE | pass | none | none |
| permission_level | DELETE | FAIL | CONSTRAINT issue | Making Changes in foreign key |
| permission_type | UPDATE | pass | none | none |
| permission_type | DELETE | FAIL | CONSTRAINT issue | Making Changes in foreign key |
| student | UPDATE | pass | none | none |
| student | DELETE | FAIL | Shouldn’t DELETE the fk, and it didnt delete it BUT | Making Changes in foreign key |

|  |  |  |  |  |
|---|---|---|---|---|
|  |  |  | with error |  |
| student_account | UPDATE | FAIL | Shouldn't UPDATE the fk, and it DID | Making Changes in foreign key |
| student_account | DELETE | FAIL | Shouldn't DELETE the fk, and it didnt delete it BUT with error | Making Changes in foreign key |
| StudentAddAddress | UPDATE | FAIL | Shouldn't UPDATE the fk, but it DID. | Making Changes in foreign key |
| StudentAddAddress | DELETE | FAIL | Shouldn't DELETE the fk, but it DELETED. | Making Changes in foreign key |
| student_address | UPDATE | pass | none | none |
| student_address | DELETE | pass | none | none |
| student_employee | UPDATE | pass | none | none |
| student_employee | DELETE | FAIL | Shouldn't DELETE the fk, and it didnt delete it BUT with error | Making Changes in foreign key |
| student_employee_account | UPDATE | pass | none | none |
| student_employee_account | DELETE | FAIL | Shouldn't DELETE the fk, but it DELETED. | Making Changes in foreign key |
| student_records | UPDATE | pass | none | none |
| student_records | DELETE | FAIL | Shouldn't DELETE the fk, and it didnt delete it BUT with error | Making Changes in foreign key |

| | | | | |
|---|---|---|---|---|
| student_record_type | UPDATE | pass | none | none |
| student_record_type | DELETE | FAIL | Shouldn’t DELETE the fk, and it didnt delete it BUT with error | Making Changes in foreign key |
| TA_Grader | DELETE | pass | none | none |
| TA_Grader | UPDATE | pass | none | none |
| transcripts | UPDATE | FAIL | Shouldn’t UPDATE the fk, but it DID. | Making Changes in foreign key |
| transcripts | DELETE | FAIL | Shouldn’t DELETE the fk, but it DELETED. | Making Changes in foreign key |
| | | | | |
| | | | | |