Student Record-Keeping Database - SQL

Date 2023.11.24

Sarah Jafari

TABLE OF CONTENTS

1. Introduction
   1.1. Document Overview
   1.2. Requirements
   1.3. Technology Used
2. Database Design
   2.1. Conceptual Design
      2.1.1. Entity-relationship diagram
   2.2. Relational Schema
      2.2.1. Relation schema diagram
3. Purpose of Tables
   3.1. Student Table (1): attributes; datatype; constraints
   3.2. Course Table (2): attributes; datatype; constraints
   3.3. Grade Table (3): attributes; datatype; constraints
   3.4. Degree Table (4): attributes; datatype; constraints
   3.5. Discipline Table (5): attributes; datatype; constraints
   3.6. Professor Table (6): attributes; datatype; constraints
   3.7. Faculty Table (7): attributes; datatype; constraints
   3.8. Emergency Contact Table (8): attributes; datatype; constraints
   3.9. XREF_student_table (9): attributes; datatype; constraints
   3.10. Summary of Tables
4. Conclusion
5. Reference List

1. Introduction

1.1 Document overview

The purpose of this database design document aims to illustrate important concepts of databases and in the form of a project as part of CP363 course. The project showcases a database representing student information as a Student Record-Keeping Database Management. The project focuses on creating a database with the appropriate information within; entities, attributes, relationships, relational schemas, and correct/complete normalizations. This document serves as a conceptual framework for the created database showcasing readers how data can be efficiently represented and how to avoid data anomalies. Within this document readers will see an entity-relationship diagram, relation schemas, and normalized tables of relevant information relating to students.

1.2 Requirements

The requirements for this project entail students creating a database containing student academic and personal information, as well the associated courses they take within their respective degrees. This information is to be shown in the form of normalized tables.

A more detailed document with the requirements list is referenced and can be found by looking at Professor Shaun Gao's "CP363A Project Requirements" file.

1.3 Technology Used

The following technologies were used in accordance with the requirements for this project; MYSQL, WINDOWS.

2. Database Design

2.1 Conceptual Design

This section will focus on providing readers with an entity-relationship diagram (ERD) to showcase the interconnectivity between all components within the database. Major entities (represented by red squares); Student, Course, Grade, Discipline, Faculty, Degree, Professor, and Emergency Contact, all contain relevant attributes (green circles) associated with their entity. For instance; student -> student_name, student_gender, etc…. Entities are linked together through relationships (gray diamonds), some are strong (single red square) and some are weak (double square) which are distinguishable so that readers are aware which entities existence relies on another, for instance; the Emergency Contact entities' existence solely relies on the Student entity. The double diamond relationships link weak entities to strong entities. Finally, readers can also see the associated keys (primary is underlined) within attributes and the specific type of attributes within entities (composite, multi, and derived). Figure 1 showcases the conceptual entity-relationship diagram for our database.

2.1.1 Entity-Relationship Diagram

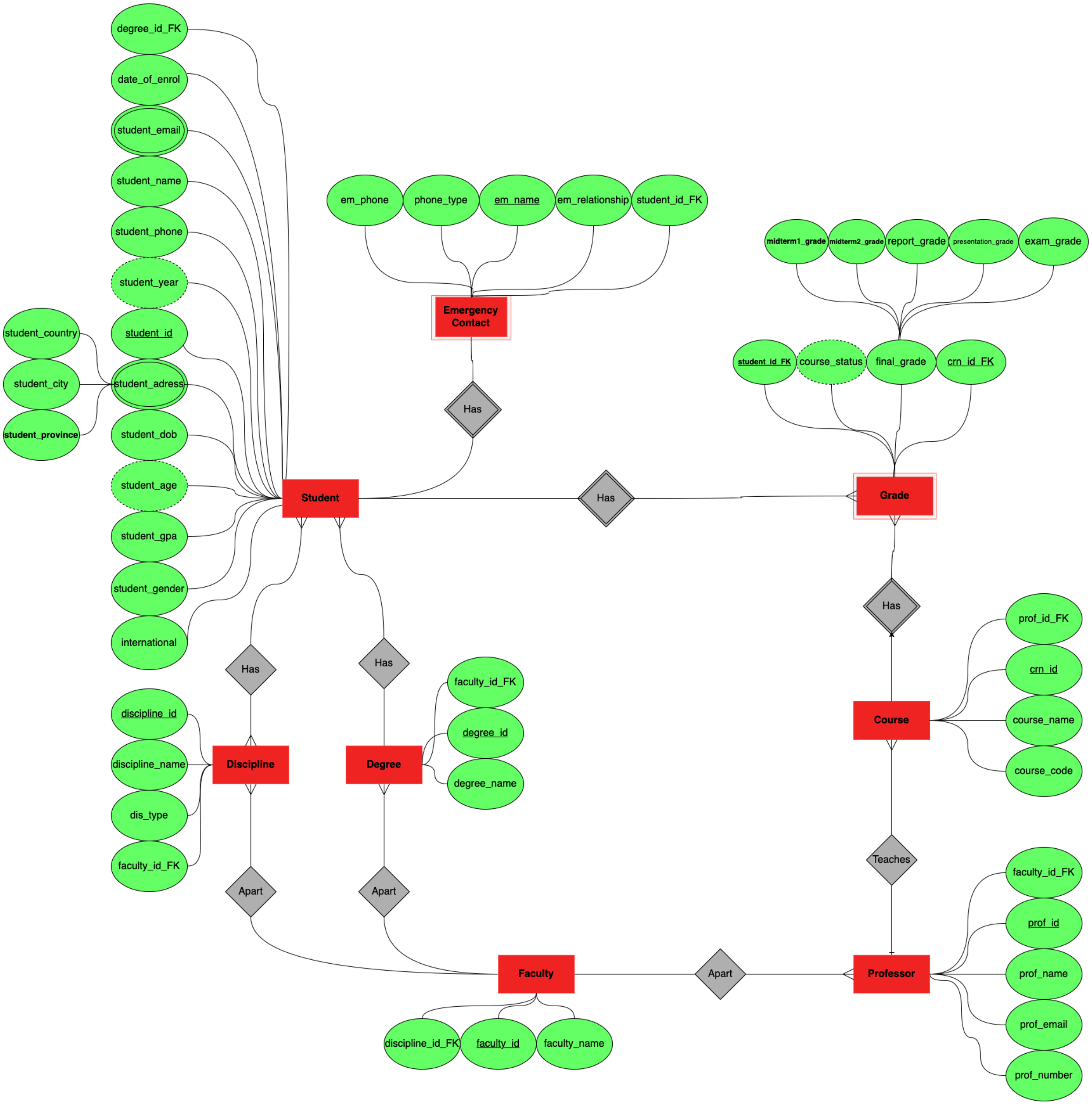
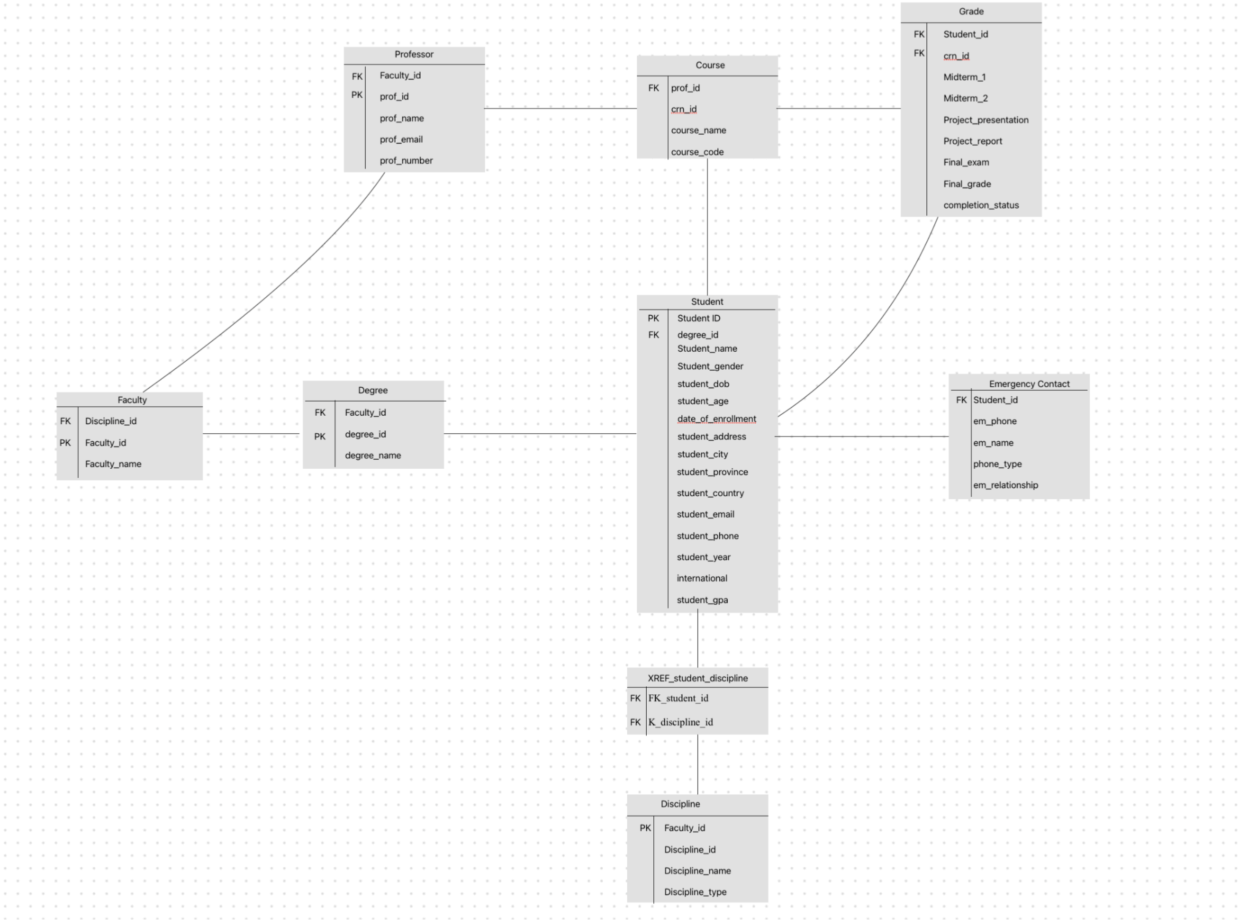


Figure 1: Entity-relationship diagram (conceptual diagram)

2.2 Relational Schema

2.2.1 Relational Schema Diagram



Above is the relational schema which outlines the entities/attributes needed to be implemented in MySQL while showing how each table relates. It also highlights the primary and foreign keys. It takes a deeper dive into the physical design of each table.

3. Database Design

Student

Attribute	DataType	Constraints
Student_id	Integer	NOT NULL, PRIMARY KEY
Student_name	String (Varchar 150)	NOT NULL
Student_gender	String (Varchar 150)	NOT NULL
student_dob	Integer	NOT NULL, CHECK student_dob > 1900
student_age	Integer	NOT NULL
date_of_enrollment	Integer	NOT NULL, CHECK Date_of_enrol > 2010
student_address	String (Varchar 150)	NOT NULL
student_city	String (Varchar 150)	NOT NULL
student_province	String (Varchar 150)	NOT NULL
student_country	String (Varchar 150)	NOT NULL
student_email	String (Varchar 150)	NOT NULL
student_phone	String (Varchar 150)	NOT NULL
degree_id	Integer	NOT NULL, FOREIGN KEY
student_year	Integer	NOT NULL
international	BOOL 1/0	NOT NULL
student_gpa	Float	NOT NULL

Course

Attribute	**DataType**	**Constraint**
crn_id	Integer	NOT NULL
course_name	String (Varchar 150)	NOT NULL
course_code	String (Varchar 150)	NOT NULL
prof_id	Integer	NOT NULL, FOREIGN KEY

Grade

Attribute	**DataType**	**Constraints**
Student_id	Integer	NOT NULL FOREIGN KEY
crn_id	Integer	NOT NULL FOREIGN KEY
Midterm_1	Float	
Midterm_2	Float	
Project_presentation	Float	
Project_report	Float	
Final_exam	Float	NOT NULL
Final_grade	Float	NOT NULL
completion_status	String (Varchar 150)	NOT NULL
		PRIMARY KEY (student_id, crn_id)

Degree

Attribute	Datatype	Constraints
Faculty_id	Integer	NOT NULL FOREIGN KEY
degree_id	Integer	NOT NULL, PRIMARY KEY
degree_name	String (Varchar 150)	NOT NULL

Discipline

Attribute	DataType	Constraints
Faculty_id	Integer	NOT NULL PRIMARY KEY
Discipline_id	Integer	NOT NULL
Discipline_name	String (Varchar 150)	NOT NULL
Discipline_type	ENUM(“Major”, “Minor”)	NOT NULL

Professor

Attribute	DataType	Constraints
Faculty_id	Integer	NOT NULL FOREIGN KEY
prof_id	Integer	NOT NULL, PRIMARY KEY
prof_name	String (Varchar 150)	NOT NULL
prof_email	String (Varchar 150)	NOT NULL
prof_number	String (Varchar 150)	NOT NULL

Faculty

Attribute	DataType	Constraints
Discpline_id	Integer	NOT NULL FOREIGN KEY
Faculty_id	Integer	NOT NULL, PRIMARY KEY
Faculty_name	String (Varchar 150)	NOT NULL

Emergency Contact

Attribute	DataType	Constraints
Student_id	Integer	NOT NULL FOREIGN KEY
em_phone	String (Varchar 150)	NOT NULL
em_name	String (Varchar 150)	NOT NULL
phone_type	String (Varchar 150)	NOT NULL
em_relationship	String (Varchar 150)	NOT NULL

XREF_student_discipline

Attribute	DataType	Constraints
FK_student_id	Integer	NOT NULL FOREIGN KEY
FK_discipline_id	Integer	NOT NULL FOREIGN KEY
		PRIMARY KEY (FK_student_id, FK_discipline_id)

3.10. Summary of Tables

Student Table:
This table is the core of the database, holding essential information about each student. It includes unique identifiers (Student_id), personal details (Student_name, Student_gender, Student_dob, Student_age), contact information (Student_address, Student_city, Student_province, Student_country, Student_email, Student_phone), academic details (Degree_id, Student_year, International status), and performance metrics (Student_gpa). This table serves as the primary source for student-related queries and is fundamental for tracking and managing student data.

Course Table:
The course table lists all courses offered, with each course identified by a unique Course_id. It includes the course name, code, and the associated professor's id. This table is crucial for managing course offerings and linking students to their enrolled courses.

Grade Table:
This table records students' performance in various assessments for each course. It includes Student_id, Course_id, grades for midterms, project presentations, project reports, final exams, the final grade, and the completion status of the course. This table is essential for tracking academic performance and determining eligibility for progression or graduation.

Degree Table:
This table outlines the various degrees offered, with each degree having a unique Degree_id. It includes the degree name and the associated faculty id. This table is used to manage degree programs and link students to their respective degrees.

Discipline Table:
This table contains information about different academic disciplines. Each discipline is identified by a Discipline_id and includes the name and type of discipline. This table is key for categorizing courses and degrees under various academic disciplines.

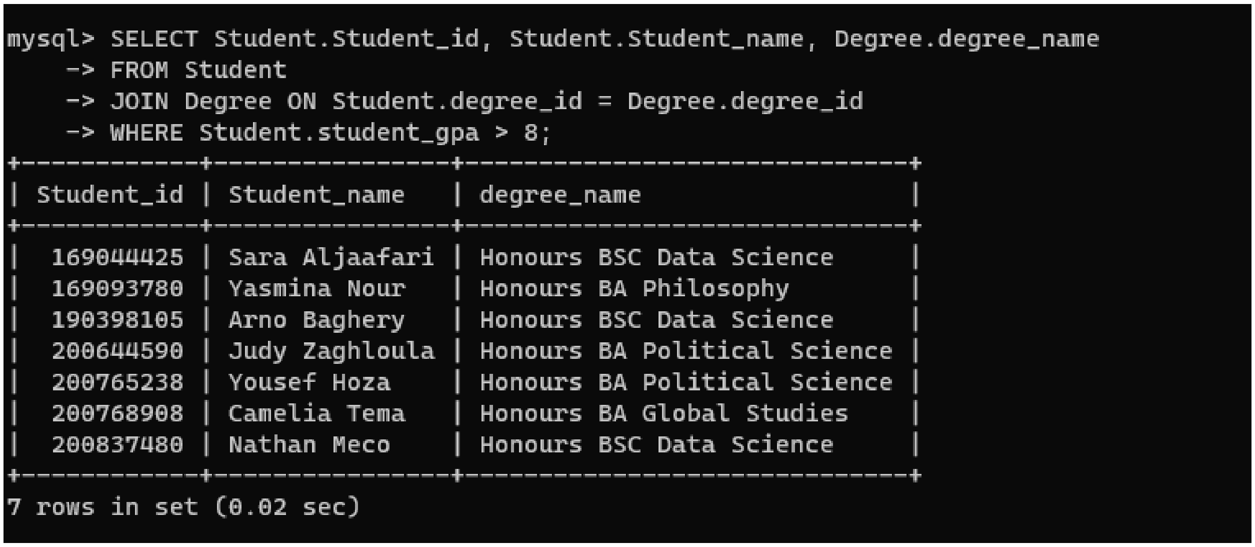
Professor Table:
The professor table includes information about faculty members. Each professor is identified by a unique Prof_id and includes their name, email, and contact number. This table is crucial for managing faculty details and linking them to the courses they teach.

Faculty Table:
This table holds information about different faculties within the institution, identified by Faculty_id. It includes the faculty name and the disciplines associated with each faculty. This table is important for organizing the institution's academic structure.

Emergency Contact Table:
This table stores emergency contact information for students. It includes the Student_id, emergency contact's phone number, name, phone type, and relationship to the student. This table is critical for addressing student welfare and emergency situations.

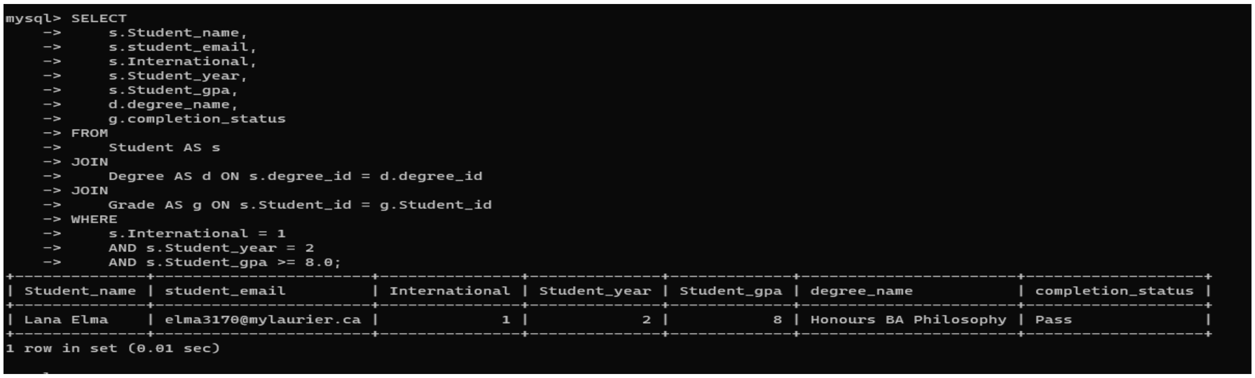
XREF_student_discipline Table:
This table stores the foreign keys of students and discipline. This table is the cross reference table between the 2 entities student and disciple so their many to many relationships can be recorded.

Relevant Test Cases:



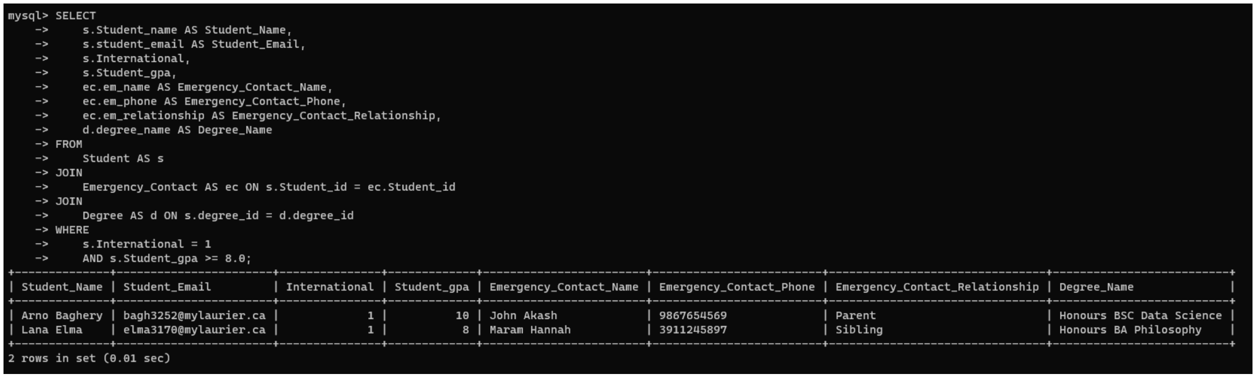
```
mysql> SELECT Student.Student_id, Student.Student_name, Degree.degree_name
    -> FROM Student
    -> JOIN Degree ON Student.degree_id = Degree.degree_id
    -> WHERE Student.student_gpa > 8;
+------------+----------------+------------------------------+
| Student_id | Student_name   | degree_name                  |
+------------+----------------+------------------------------+
|  169044425 | Sara Aljaafari | Honours BSC Data Science     |
|  169093780 | Yasmina Nour   | Honours BA Philosophy        |
|  190398105 | Arno Baghery   | Honours BSC Data Science     |
|  200644590 | Judy Zaghloula | Honours BA Political Science |
|  200765238 | Yousef Hoza    | Honours BA Political Science |
|  200768908 | Camelia Tema   | Honours BA Global Studies    |
|  200837480 | Nathan Meco    | Honours BSC Data Science     |
+------------+----------------+------------------------------+
7 rows in set (0.02 sec)
```

Retrieve the names and email addresses of international students who are enrolled in the second year (Student_year = 2) and have a GPA greater than or equal to 8.0. Additionally, include their degree names and the completion status from the Grade table.



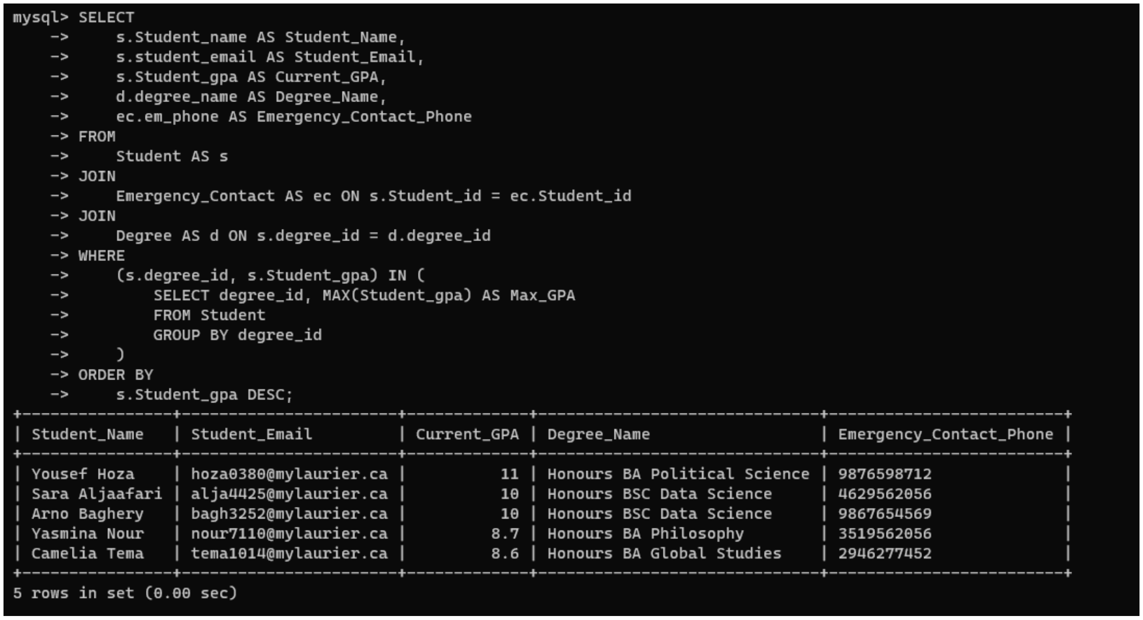
```
mysql> SELECT
    ->     s.Student_name,
    ->     s.student_email,
    ->     s.International,
    ->     s.Student_year,
    ->     s.Student_gpa,
    ->     d.degree_name,
    ->     g.completion_status
    -> FROM
    ->     Student AS s
    -> JOIN
    ->     Degree AS d ON s.degree_id = d.degree_id
    -> JOIN
    ->     Grade AS g ON s.Student_id = g.Student_id
    -> WHERE
    ->     s.International = 1
    ->     AND s.Student_year = 2
    ->     AND s.Student_gpa >= 8.0;
+--------------+-----------------------+---------------+--------------+-------------+-----------------------+-------------------+
| Student_name | student_email         | International | Student_year | Student_gpa | degree_name           | completion_status |
+--------------+-----------------------+---------------+--------------+-------------+-----------------------+-------------------+
| Lana Elma    | elma3170@mylaurier.ca |             1 |            2 |           8 | Honours BA Philosophy | Pass              |
+--------------+-----------------------+---------------+--------------+-------------+-----------------------+-------------------+
1 row in set (0.01 sec)
```

Retrieve the names, email addresses, and emergency contact information (name, phone, relationship) of international students with a GPA greater than or equal to 8.0. Additionally, include the degree name of each student.



```
mysql> SELECT
    ->     s.Student_name AS Student_Name,
    ->     s.student_email AS Student_Email,
    ->     s.International,
    ->     s.Student_gpa,
    ->     ec.em_name AS Emergency_Contact_Name,
    ->     ec.em_phone AS Emergency_Contact_Phone,
    ->     ec.em_relationship AS Emergency_Contact_Relationship,
    ->     d.degree_name AS Degree_Name
    -> FROM
    ->     Student AS s
    -> JOIN
    ->     Emergency_Contact AS ec ON s.Student_id = ec.Student_id
    -> JOIN
    ->     Degree AS d ON s.degree_id = d.degree_id
    -> WHERE
    ->     s.International = 1
    ->     AND s.Student_gpa >= 8.0;
+--------------+-----------------------+---------------+-------------+------------------------+-------------------------+--------------------------------+--------------------------+
| Student_Name | Student_Email         | International | Student_gpa | Emergency_Contact_Name | Emergency_Contact_Phone | Emergency_Contact_Relationship | Degree_Name              |
+--------------+-----------------------+---------------+-------------+------------------------+-------------------------+--------------------------------+--------------------------+
| Arno Baghery | bagh3252@mylaurier.ca |             1 |          10 | John Akash             | 9867654569              | Parent                         | Honours BSC Data Science |
| Lana Elma    | elma3170@mylaurier.ca |             1 |           8 | Maram Hannah           | 3911245897              | Sibling                        | Honours BA Philosophy    |
+--------------+-----------------------+---------------+-------------+------------------------+-------------------------+--------------------------------+--------------------------+
2 rows in set (0.01 sec)
```

Selecting contact info of students who have achieved the highest GPA in their respective degree programs, so that other students can reach out to them



```
mysql> SELECT
    ->     s.Student_name AS Student_Name,
    ->     s.student_email AS Student_Email,
    ->     s.Student_gpa AS Current_GPA,
    ->     d.degree_name AS Degree_Name,
    ->     ec.em_phone AS Emergency_Contact_Phone
    -> FROM
    ->     Student AS s
    -> JOIN
    ->     Emergency_Contact AS ec ON s.Student_id = ec.Student_id
    -> JOIN
    ->     Degree AS d ON s.degree_id = d.degree_id
    -> WHERE
    ->     (s.degree_id, s.Student_gpa) IN (
    ->         SELECT degree_id, MAX(Student_gpa) AS Max_GPA
    ->         FROM Student
    ->         GROUP BY degree_id
    ->     )
    -> ORDER BY
    ->     s.Student_gpa DESC;
+----------------+-----------------------+-------------+------------------------------+-------------------------+
| Student_Name   | Student_Email         | Current_GPA | Degree_Name                  | Emergency_Contact_Phone |
+----------------+-----------------------+-------------+------------------------------+-------------------------+
| Yousef Hoza    | hoza0380@mylaurier.ca |          11 | Honours BA Political Science | 9876598712              |
| Sara Aljaafari | alja4425@mylaurier.ca |          10 | Honours BSC Data Science     | 4629562056              |
| Arno Baghery   | bagh3252@mylaurier.ca |          10 | Honours BSC Data Science     | 9867654569              |
| Yasmina Nour   | nour7110@mylaurier.ca |         8.7 | Honours BA Philosophy        | 3519562056              |
| Camelia Tema   | tema1014@mylaurier.ca |         8.6 | Honours BA Global Studies    | 2946277452              |
+----------------+-----------------------+-------------+------------------------------+-------------------------+
5 rows in set (0.00 sec)
```

4. Conclusion

In creating this database, we have successfully implemented a streamlined method aligning with key database creation principles. Our design ensures no data duplication or loss and maintains system organization. Centered around the student entity, it seamlessly connects related elements like student records, emergency contact info, grades, course, degree, discipline and professor details.

Key highlights include adherence to the third normal form for normalization, providing data consistency. Utilizing entity-relationship diagrams for a clear conceptual framework and detailed relational schemas for practicality, our design incorporates primary and foreign keys for robust data integrity. This adaptable database not only meets current needs but anticipates future expansion, reflecting our commitment to an efficient and forward-thinking system. Our report showcases our group's ability to translate complex requirements into a coherent, functional, and user-friendly database system, offering a structured foundation for future development.

5. References

Gao, S. (2023 Sep). CP363A Project Requirements

Gao, S. (2023 Sep). Database-I-week-1-2-relational-model

Gao, S. (2023 Oct). Database-I-week-5-2-DB-design-1

Gao, S. (2023 Oct). Database-I-week-6-1-DB-design-2

Gao, S. (2023 Oct). Database-I-week-6-2-DB-design-3

Gao, S. (2023 Nov). Database-I-week-7-1-DB-design-4

"MySQL :: MySQL 8.0 Reference Manual :: 13.1 Data Definition Statements." *Dev.mysql.com*, dev.mysql.com/doc/refman/8.0/en/sql-data-definition-statements.html.