Highschool Database

CIS 3400 ETRA [17216]

Professor Qiang Gao

**GROUP 5**

**Aryan Bousri** (aryan.bousri@baruchmail.cuny.edu) [Group Leader]

**Leonardo Cuapio** (leonardo.cuapio@baruchmail.cuny.edu)

**Pin Yuan Zhu Lee** (pinyuan.zhulee@baruchmail.cuny.edu)

**Fatima** (fatima.tariq@baruchmail.cuny.edu)

**Ranvir Saini** (ranvir.saini@baruchmail.cuny.edu)

SUMMARY

For this project database, we as a group decided to use School/ educational institutions as our topic. Every school has a major system in place to manage the various amounts of data related to each specific student, which shows the relevance of the database when it comes to keeping records, maintaining organization, and allowing specific students to be differentiated from others. Through this database, not only can we keep track of students' performance and efforts, we can also keep track of courses, teachers, and non-academic activities. The database can also help with enrollment, and registration by identifying student’s eligibility based on prior courses. Additionally, the database allows users to identify awards and grants for qualified students. This is done by utilizing student extracurriculars, grades, and academics.

Our team began by building an ERD (entity relationship model) where the relationships between different entities, such as grades and classes were be made. We included a visual of the entity tables as well as the corresponding ERD sentences.

The entities we included are: course, class, grade, student, semester, enrollment, teacher, activity, and ActivtyParticipation.

Our ERD model is then converted into an RDM (relational model) which closes in towards our final goal of implementing a proper database. While the ERD abstractly shows the relationships between entities and attributes, the RDM defines them in a way a database could interpret them. We then entered a normalization phase, where our main objective was to eliminate data redundancy and to produce a clearer data model. The next step is SQL implementation, which allows our team to bring our initially abstract database idea into a completed form, where now we can manipulate our database in Microsoft Access with various scenarios to ensure that it serves its purpose.

The scenarios show exactly how this database can be used. Our examples demonstrate how students can be identified for different scholarships/grants, programs, registration requirements, and class information. From figuring out which students in the robotics team won a money prize to identifying seniors who qualify for the prime scholarship, our database can be used to cover many areas.

**ERD Model**

A diagram of a data flow

Description automatically generated

**ERD Sentences**

**Entities: course, class**

One course has one or many classes. 1..\*

One class belongs to one and only one course. 1..1

**Entities: class, semester**

One semester offers one or many classes. 1..\*

One class is under one and only one semester. 1..1

**Entities: student, enrollment, class**

The student enrolls in one or many times.  1..\*

The enrolled student is one and only one student.  1..1

Class has one or many enrolled students. 1..\*

An enrolled student belongs to one and only one class. 1..1

**Entities: student, award**

A class assigns one or many grades. 1..\*

A grade belongs to one and only one class. 1..1

**Entities: grade, class**

A student has one or many grades. 1..\*

A grade belongs to one and only one student. 1..1

**Entities: teacher, class**

A teacher teaches zero or many classes. 0..\*

A class is taught by one and only one teacher.  1..1

**Entities: ActivityParticipation, student**

A student has participation if the student participates.

A student participation belongs to one and only one student.

**Entities: ActivityParticipation, Activity**

One activity may have one or many participants. 1..\*

One activitypartcipation belongs to one and only one student. 1..1

**Entities. Activity**

Activity can be classified as 3 main categories: club, academic competition, and sport team.

**Converting ERD to RDM**

Course (Course\_ID, Course\_Name, Course\_Description)

Class (Class\_ID, Class\_Number, Class\_Description, Course\_ID(fk), Teacher\_ID(fk), Semester\_Term(fk))

Teacher (Teacher\_ID, Teacher\_Name, Teacher\_PhoneNum, Teacher\_Subject)

Semester (Semester\_Term, Semester\_Date)

Grade (Grade\_ID, Grade\_GPA, Class\_ID(fk))

Enrollment (Enrollment\_ID, Enrollment\_Date, Class\_ID(fk), Student\_ID(fk))

Student (Student\_ID, Student\_FName, Student\_LName, Student\_DOB, Student\_PhoneNum, Student\_Address, Student\_Status)

Award (Award\_ID, Award\_Description, Student\_ID(fk))

ActivityParticipation (ActivityParticipation\_ID, Student\_ID(fk), Activity\_ID(fk))

Activity (Activity\_ID, Club\_Name, Academic\_Subject, Sport\_Basketball, Sport\_Swimming, Sport\_Track, Type\_Activity)

**Normalization**

Course (Course\_ID, Course\_Name, Course\_Description)

Step 1: Key? Yes, (Course\_ID)

Step 2: Partial key dependency? No

Step 3: Transitive dependency? No

Final relationship: Course (Course\_ID, Course\_Name, Course\_Description)

Class (Class\_ID, Class\_Number, Class\_Description, Course\_ID(fk), Teacher\_ID(fk), Semester\_Term(fk))

Step 1: Key? Yes, (Class\_ID)

Step 2: Partial key dependency? No

Step 3: Transitive dependency? No

Final relationship: Class (Class\_ID, Class\_Number, Class\_Description, Course\_ID(fk), Teacher\_ID(fk), Semester\_Term(fk))

Teacher (Teacher\_ID, Teacher\_Name, Teacher\_PhoneNum, Teacher\_Subject)

Step 1: Key? Yes, (Teacher\_ID)

Step 2: Partial key dependency? No

Step 3: Transitive dependency? No

Final relationship: Teacher (Teacher\_ID, Teacher\_Name, Teacher\_PhoneNum, Teacher\_Subject)

Semester (Semester\_Term, Semester\_Date)

Step 1: Key? Yes, (Semester\_Term)

Step 2: Partial key dependency? No

Step 3: Transitive dependency? No

Final relationship: Semester (Semester\_Term, Semester\_Date)

Grade (Grade\_ID, Grade\_GPA, Class\_ID(fk))

Step 1: Key? Yes, (Grade\_ID)

Step 2: Partial key dependency? No

Step 3: Transitive dependency? No

Final relationship: Grade (Grade\_ID, Grade\_GPA, Class\_ID(fk))

Enrollment (Enrollment\_ID, Enrollment\_Date, Class\_ID(fk), Student\_ID(fk))

Step 1: Key? Yes, (Enrollment\_ID)

Step 2: Partial key dependency? No

Step 3: Transitive dependency? No

Final relationship: Enrollment (Enrollment\_ID, Enrollment\_Date, Class\_ID(fk), Student\_ID(fk))

Student (Student\_ID, Student\_FName, Student\_LName, Student\_DOB, Student\_PhoneNum, Student\_Address, Student\_Status)

Step 1: Key? Yes, (Student\_ID)

Step 2: Partial key dependency? No

Step 3: Transitive dependency? No

Final relationship: Student (Student\_ID, Student\_FName, Student\_LName, Student\_DOB, Student\_PhoneNum, Student\_Address, Student\_Status)

Award (Award\_ID, Award\_Description, Student\_ID(fk))

Step 1: Key? Yes, (Award\_ID)

Step 2: Partial key dependency? No

Step 3: Transitive dependency? No

Final relationship: Award (Award\_ID, Award\_Description, Student\_ID(fk))

ActivityParticipation (ActivityParticipation\_ID, Student\_ID(fk), Activity\_ID(fk))

Step 1: Key? Yes, (ActivityParticipation\_ID)

Step 2: Partial key dependency? No

Step 3: Transitive dependency? No

Final relationship: ActivityParticipation (ActivityParticipation\_ID, Student\_ID(fk), Activity\_ID(fk))

Activity (Activity\_ID, Club\_Name, Academic\_Subject, Sport\_Basketball, Sport\_Swimming, Sport\_Track, Type\_Activity)

Step 1: Key? Yes, (Activity\_ID)

Step 2: Partial key dependency? No

Step 3: Transitive dependency? No

Final relationship: Activity (Activity\_ID, Club\_Name, Academic\_Subject, Sport\_Basketball, Sport\_Swimming, Sport\_Track, Type\_Activity)

**Final Relationships**

Course (Course\_ID, Course\_Name, Course\_Description)

Class (Class\_ID, Class\_Number, Class\_Description, Course\_ID(fk), Teacher\_ID(fk), Semester\_Term(fk))

Teacher (Teacher\_ID, Teacher\_Name, Teacher\_PhoneNum, Teacher\_Subject)

Semester (Semester\_Term, Semester\_Date)

Grade (Grade\_ID, Grade\_GPA, Class\_ID(fk))

Enrollment (Enrollment\_ID, Enrollment\_Date, Class\_ID(fk), Student\_ID(fk))

Student (Student\_ID, Student\_FName, Student\_LName, Student\_DOB, Student\_PhoneNum, Student\_Address, Student\_Status)

Award (Award\_ID, Award\_Description, Student\_ID(fk))

ActivityParticipation (ActivityParticipation\_ID, Student\_ID(fk), Activity\_ID(fk))

Activity (Activity\_ID, Club\_Name, Academic\_Subject, Sport\_Basketball, Sport\_Swimming, Sport\_Track, Type\_Activity)

**Database Implementation and Inserting Data**

CREATE TABLE Course (

Course\_ID NUMBER NOT NULL,

Course\_Name VARCHAR(250),

Course\_Description VARCHAR(250),

CONSTRAINT pk\_Course PRIMARY KEY (Course\_ID)

)

INSERT INTO Course VALUES (1, “Calculus 1”, “Limit & derivative”)

CREATE TABLE Teacher (

Teacher\_ID NUMBER NOT NULL,

Teacher\_Name VARCHAR(250),

Teacher\_PhoneNum NUMBER,

Teacher\_Subject VARCHAR(250),

CONSTRAINT pk\_Teacher PRIMARY KEY (Teacher\_ID)

)

INSERT INTO Teacher VALUES (1, "Jake", 5687459631, " Math")

CREATE TABLE Semester (

Semester\_Term VARCHAR(250),

Semester\_Date DATE,

CONSTRAINT pk\_Semester PRIMARY KEY (Semester\_Term)

)

INSERT INTO Semester VALUES ("Fall 2023", "7/20/2023")

CREATE TABLE Class (

Class\_ID NUMBER NOT NULL,

Class\_Number VARCHAR(250),

Class\_Description VARCHAR(250),

Course\_ID NUMBER,

Teacher\_ID NUMBER,

Semester\_Term VARCHAR(250),

CONSTRAINT pk\_Class PRIMARY KEY (Class\_ID),

CONSTRAINT fk1\_Class FOREIGN KEY (Course\_ID) REFERENCES  Course(Course\_ID),

CONSTRAINT fk2\_Class FOREIGN KEY (Teacher\_ID) REFERENCES Teacher(Teacher\_ID),

CONSTRAINT fk3\_Class FOREIGN KEY (Semester\_Term) REFERENCES Semester(Semester\_Term)

)

INSERT INTO Class VALUES (1, "MTH 1001", "Math", 1, 1, "Fall 2023")

CREATE TABLE Grade (

Grade\_ID NUMBER NOT NULL,

Grade\_GPA NUMBER,

Class\_ID NUMBER,

CONSTRAINT pk\_Grade PRIMARY KEY (Grade\_ID),

CONSTRAINT fk\_Grade FOREIGN KEY (Class\_ID) REFERENCES  Class(Class\_ID)

)

INSERT INTO Grade VALUES (101, 62, 1)

CREATE TABLE Student (

Student\_ID NUMBER NOT NULL,

Student\_FName VARCHAR(250),

Student\_LName VARCHAR(250),

Student\_DOB DATE,

Student\_PhoneNum NUMBER,

Student\_Address VARCHAR(250),

Student\_Status VARCHAR(250),

CONSTRAINT pk\_Student PRIMARY KEY (Student\_ID)

)

INSERT INTO Student VALUES (101, "Jayce", "Thomas", "9/11/2006", 5461268463, "155 W. Prospect Ave.", "Freshman")

CREATE TABLE Enrollment (

Enrollment\_ID NUMBER NOT NULL,

Enrollment\_Date DATE,

Class\_ID NUMBER,

Student\_ID NUMBER,

CONSTRAINT pk\_Enrollment PRIMARY KEY (Enrollment\_ID),

CONSTRAINT fk1\_Enrollment FOREIGN KEY (Class\_ID) REFERENCES  Class(Class\_ID),

CONSTRAINT fk2\_Enrollment FOREIGN KEY (Student\_ID) REFERENCES  Student(Student\_ID)

)

INSERT INTO Enrollment VALUES(1, "6/25/2023", 4, 101)

CREATE TABLE Award (

Award\_ID NUMBER NOT NULL,

Award\_Description VARCHAR(250),

Student\_ID NUMBER,

CONSTRAINT pk\_Award PRIMARY KEY (Award\_ID),

CONSTRAINT fk\_Award FOREIGN KEY (Student\_ID) REFERENCES  Student (Student\_ID)

)

INSERT INTO Award VALUES (1001, "Math Team 3rd Place", 103)

CREATE TABLE Activity (

Activity\_ID NUMBER NOT NULL,

Club\_Name VARCHAR(250),

Academic\_Subject VARCHAR(250),

Sport\_Basketball VARCHAR(250),

Sport\_Swimming VARCHAR(250),

Sport\_Track VARCHAR(250),

Type\_Activity VARCHAR(250),

CONSTRAINT pk\_Activity PRIMARY KEY (Activity\_ID)

)

INSERT INTO Activity(Activity\_ID, Sport\_Basketball) VALUES (1, "Yes")

CREATE TABLE ActivityParticipation (

ActivityParticipation\_ID NUMBER NOT NULL,

Student\_ID NUMBER,

Activity\_ID NUMBER,

CONSTRAINT pk\_ActivityParticipation PRIMARY KEY (ActivityParticipation\_ID),

CONSTRAINT fk1\_ActivityParticipation  FOREIGN KEY (Student\_ID) REFERENCES  Student (Student\_ID),

CONSTRAINT fk2\_ActivityParticipation  FOREIGN KEY (Activity\_ID) REFERENCES  Activity (Activity\_ID)

)

INSERT INTO ActivityParticipation VALUES (1, 101, 5)

**Scenario**

Seniors are qualified for a $5,000 Prime Scholarship. Write a query to display the students (first and last name) who qualify for this scholarship.

SELECT Student\_ID, Student\_FNAME, Student\_Lname

FROM Student

WHERE Student\_Status = "Senior"

A screenshot of a computer

Description automatically generated

The Swimming Organization is giving out $1,000 for students (first and last name)who participate in swimming. Write a query to display the students who can get the $1,000.

SELECT A.Student\_ID, S.Student\_FNAME, S.Student\_Lname

FROM Student S INNER JOIN ActivityParticipation A

ON A.Student\_ID = S.Student\_ID

WHERE A.Activity\_ID = 2

A screenshot of a computer

Description automatically generated

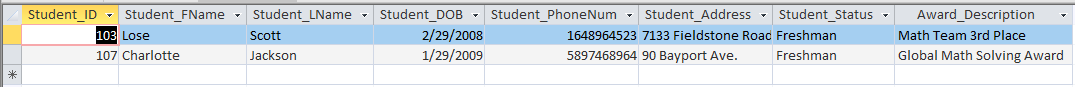
Students who have participated in  a math competition and gotten an award are invited to participate in the MathHouse Internship Program. Write a query to display the information of students who can apply for this internship and the award they achieve.

SELECT S.Student\_ID, S.Student\_FName, S.Student\_LName, S.Student\_DOB, S.Student\_PhoneNum, S.Student\_Address, S.Student\_Status, A.Award\_Description

FROM Student S INNER JOIN Award A

ON A.Student\_ID = S.Student\_ID

WHERE A.Award\_Description LIKE "\*Math\*"



The semester is ending. Only students who are taking calculus 1 can register for calculus 2. Write a query to display the first and last name of the students who can register for calculus 2.

SELECT S.Student\_ID, S.Student\_FName, S.Student\_LName

FROM Student S INNER JOIN Enrollment E

ON E.Student\_ID = S.Student\_ID

WHERE E.Class\_ID IN (1,2)

A screenshot of a computer

Description automatically generated

NASA is hosting a robot competition. Participants can win up to $10,000 and NASA will cover the tuition of all students of the winning team. Write a query to display the member(s) of the Robotic Team.

SELECT A.Academic\_Subject, S.Student\_FName, S.Student\_LName

FROM Activity A, ActivityParticipation P, Student S

WHERE A.Activity\_ID=P.Activity\_ID AND P.Student\_ID=S.Student\_ID AND  A.Academic\_Subject = "Robotic Team"

A close-up of a computer screen

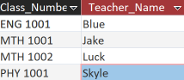
Description automatically generated

During the enrollment period, students want to know the teacher of the class they want to enroll in. Write a query to display the class number and teacher's name.

SELECT DISTINCT Class\_Number, Teacher\_Name

FROM class c, teacher t

WHERE c.Teacher\_ID = t.Teacher\_ID



Conclusion

The most challenging part of the project we as a group experienced was the creation of the ERD. It required us to consider which entities to include while thinking about how they would link. Our group had many discussions and brainstorming sessions on this matter. This is what made the process difficult since everyone had different and clashing ideas. Once we agreed on which entities we would use for our school database, we needed to figure out the attributes and the relationships between the entities. After multiple revisions and discussions, we created our ERD model. The project’s easiest part was converting the ERD to RDM. This is because as long as the rules of the different relationships are followed, it is an easy and smooth process. A set of relations was the result of the process which was derived by converting each entity using the relationships.

If we were to do this project again, it is essential to think ahead of the possible scenarios and create links and relationships for them. For example, the student entity and grade entity were not linked, so there was no way to make a scholarship/grant for students with high grades/GPA.

Our school database system works as intended, and as proposed. It identifies specific students that are eligible for different scholarships/grants. Additionally it allows users to track students' performance and efforts, while also keeping track of courses, teachers, and non-academic activities.