**Step 1: identifying entities and primary keys based on problem description**

We were searching for the potential entities - people, things, and events which are represented as nouns in the problem description. Here are all entities that have been identified. Here it is assumed that each class may have multiple exams.

**Parent**

ParentID

ParentName

ParentPhoneNo

**Payment**

PaymentID

StudentID

TotalAmount

PaymentDate

**Exam**

ClassID

ExamID

ExamDate

**Student**

StudentID

StudentName

StudentAddress

Balance

**Registration**

StudentID

ClassID

RegistrationDate

**Class**

ClassID

ClassName

ClassFee

ClassStartTime

ClassDuration

**Step 2: identifying relationships and foreign keys**

We identified the following relationships:

(1) Class registration relationship between “student” and “class”. It is a many-to-many

relationship, and we have already built “registration” as the associative entity with

studentID and classID as the composite primary key.

(2) Family relationship between “student” and “parent”. Another M:N relationship, and we need to build a new associative entity “familyRelationship”.

(3) “Paid-for” relationship between “student” and “payment”, and “paying” relationship

between “parent” and “payment”, both one-to-many relationships, and we put the foreign key on the many side, which is “payment” in this case.

(4) One-to-many relationship between “class” and “exam” since one class may have multiple exams, such as midterm and final. “exam” is a weak entity.

(5) Many-to-many relationship between “student” and “exam” since a student might have to take several exams in a class.



**Step 3: check whether the data model conforms to 3NF**

Now we should check if each entity conforms to 3NF. It seems all entities have primary keys (1NF), and no partial and transitive dependencies.

A final review of this ERD revealed one problem: when entering a grade to the database, the current solution would not check whether the student is actually registered in that class.

The only way to verify is to associate the performance table with the registration table, not the student table.

In the following revised solution, the “student Performance” table has two foreign keys, both are composite foreign keys. The first one (studentID, classID) references the registration table, checking if the student has registered for this class. The second foreign key (examID, classID) references the exam table, checking if the exam actually exists.

