Step 1 Derive relations for logical data model

1.1 Entity Types

For each entity type, create a relation that contains all simple attributes.

Package (packageName, cost)
Primary Key packageName

Tainer (trainerNumber, fname, lname, phoneNumber)
Primary Key trainerNumber

Course (classNmae, maxParticipant, currentParticipant, startDate, endDate)
Primary Key className

Member (memberNumber, fname, lname, phoneNumber)
Primary Key memberNumber

Transaction (transactionNumber, total, transactionDate, type)
Primary Key transactionNumber

Equipment (itemNumber, description, quantity)
Primary Key itemNumber

Level (levelId, minSpending, discount)
Primary Key levelId

1.2 One-to-many binary relationship types

For each 1:* binary relationship, the entity on the one side is considered a parent, and the entity on the many side is considered a child. We use the primary key attribute of the parent entity as a foreign key in a child entity.

Package (packageName, cost)
Primary Key packageName

Tainer (trainerNumber, fname, lname, phoneNumber)
Primary Key trainerNumber

Course (classNmae, maxParticipant, currentParticipant, startDate, endDate, trainerNumber, day, hour, minute)
Primary Key className; Foreign Key: trainerNumber;

Member (memberNumber, fname, lname, phoneNumber, levelId)

Primary Key memberNumber; Foreign Key: levelId

Transaction (transactionNumber, total, transactionDate, type, memberNumber)
Primary Key transactionNumber; Foreign Key: memberNumber

Equipment (itemNumber, description, quantity)
Primary Key itemNumber

Level (levelId, minSpending, discount)
Primary Key levelId

1.3 One-to-one binary relationship types. N/A
1.4 One-to-one recursive relationships. N/A
1.5 Superclass/subclass relationship types. N/A
1.6 Many-to-many binary relationship types

For each many-to-many relationship, create a relation that includes the primary keys of each entity type.

Then, the new relations:

CoursePackage (packageName, className)
Primary Key: packageName className; Foreign Key: packageName, className

Purchase (memberNumber, courseName, transactionNumber)
Primary Key: memberNumber, courseName, transactionNumber;
Foreign Key: memberNumber, courseNumber, transactionNumber;

EquipmentLoan (loanNumber, itemNumber, memberItem, quantity, borrowedDate, returnedDate)
Primary key: loanNumber; Foreign Key: itemNumber, memberItem

Note: the primary key cannot be formed from itemNumber, memberItem and borrowedDate since the same item can be borrowed by the same user several times at the same day.

Then, the final set of relations before the normalization analysis are:

Package (packageName, cost)
Primary Key packageName

Tainer (trainerNumber, fname, lname, phoneNumber)
Primary Key trainerNumber

Course (classNmae, maxParticipant, currentParticipant, startDate, endDate, trainerNumber, day, hour, minute)
Primary Key className; Foreign Key: trainerNumber;

Member (memberNumber, fname, lname, phoneNumber, levelId)

Primary Key memberNumber; Foreign Key: levelId

Transaction (transactionNumber, total, transactionDate, type, memberNumber)
Primary Key transactionNumber; Foreign Key: memberNumber

Equipment (itemNumber, description, quantity)
Primary Key itemNumber

Level (levelId, minSpending, discount)
Primary Key levelId

CoursePackage (packageName, className)
Primary Key: packageName className; Foreign Key: packageName, className

Purchase (memberNumber, courseName, transactionNumber)
Primary Key: memberNumber, courseName, transactionNumber;
Foreign Key: memberNumber, courseNumber, transactionNumber;

EquipmentLoan (loanNumber, itemNumber, memberItem, quantity, borrowedDate, returnedDate)
Primary key: loanNumber; Foreign Key: itemNumber, memberItem