Entities:

Department(name, abbreviation)
    name unique   /*Since Abbreviation and name are keys, but “abbreviation” is our primary key. We have to add a constraint stating that name should be unique */

StudyProgramme(name, abbreviation)

Branch(spName, name)
    spName -> StudyProgramme.name

Course(courseName, creditPoints, code, departName, departAbb)
    departAbb -> Department.abbreviation
    departName -> Department.Name

LimitedCourse(courseCode, #maxParticipants)
    courseCode -> Course.code

Student(name, id, spName, bName, spAbb)
    spName -> StudyProgramme.Name
    bName -> Branch.name
    spAbb -> StudyProgramme.Abbreviation

Classification(name)

Relationships:

HostBy(dAbbreviation, spName)
    dAbbreviation -> Department.abbreviation
    spName -> StudyProgramme.name

Has(studentID, bName, bProgramme)
    (bName, bProgramme) -> Branch.(name, programme)
    (studentID, bProgramme) -> Student.(studentID, programme)

/*In order to prevent a cyclic relationship, we have added this contraint. This constraint states that the StudyProgramme of the student’s branch has to be the same as the StudyProgramme of the student. */

IsMandatory(courseCode, spName)
    courseCode -> Course.code
    spName -> StudyProgramme.name

IsRecommended(courseCode, bName, bProgramme)
    courseCode -> Course.code
    (bName, bProgramme) -> Branch.(name, programme)

IsAddMandatory(courseCode, bName, bProgramme)
    courseCode -> Course.code
    (bName, bProgramme) -> Branch.(name, programme)

IsRegistered(courseCode, studentID)
courseCode -> Course.code
studentID -> Student.id

HasCompleted(courseCode, studentID, grade)
courseCode -> Course.code
studentID -> Student.id

IsWaiting(sinceDate, studentID, courseCode)
studentID -> Student.id
courseCode -> LimitedCourse.code
(sinceDate, courseCode) unique /*In order to make it impossible for two students to have the same sinceDate, we have to add a constraint*/

HasClassification(courseCode, className)
courseCode -> Course.code
className -> Classification.name