select \* from student;

select \* from unitofstudy;

select \* from whenoffered;

select \* from transcript;

select \* from uosoffering;

select \* from requires;

create procedure login(in studentID char(20), in studentPassword char(20))

select id

from student

where Name = studentID and Password = studentPassword;

select \* from transcript

where StudId = '3213' and grade is null;

create procedure currentCourses(in studentID char(20), in thisYear int(11), in thisQuarter char(2))

select UosCode, semester,year,semester

from transcript

where StudId = studentID and year = thisYear and Semester = thisQuarter and grade is null;

create procedure transcript(in studentID char(20))

select UosCode, semester,year,Grade

from transcript

where StudId = studentID;

select StudId,UoSCode, Semester, Year, Enrollment, MaxEnrollment, Name, Grade

From uosoffering natural join transcript, faculty

where uosoffering.InstructorId = faculty.Id;

select UoSCode, Semester, Year, Enrollment, MaxEnrollment, Name

From uosoffering, faculty

where uosoffering.InstructorId = faculty.Id;

create procedure courseDetail(in studentID char(20), in courseNum char(8), in thisQuarter char(2), in thisYear int(11))

select StudId,UoSCode, Semester, Year, Enrollment, MaxEnrollment, Name, Grade, UoSName

From uosoffering natural join transcript natural join unitofstudy, faculty

where uosoffering.InstructorId = faculty.Id

and StudId = studentID

and UoSCode = courseNum

and year =thisYear

and Semester = thisQuarter;

create procedure AllCourses(in thisyear int(11),in thisQuarter char(2), in nextQuarter char(2),in studentID char(20))

select UoSCode, Semester, Year, Enrollment, MaxEnrollment, Name,UoSName

From uosoffering natural join unitofstudy, faculty

where uosoffering.InstructorId = faculty.Id

and ((year= thisyear and semester = thisQuarter) or

(thisQuarter !="Q1" and year= thisyear and semester = nextQuarter) or

(thisQuarter = "Q1" and year = thisyear +1))

and not exists(select \*

from transcript

where StudId = studentID

and transcript.UoSCode = uosoffering.UoSCode

and transcript.semester = uosoffering.Semester

and transcript.year = uosoffering.year

);

DELIMITER $$

create procedure Enroll(in studentID char(20), in classNum char(8), in requiredQuarter char(2), in requiredYear int(11))

begin

start transaction;

if not exists (select \* from requires where UoSCode = classNum) then

BEGIN

if exists (select \* from uosoffering where Enrollment< MaxEnrollment and UoSCode =classNum and year = requiredYear and semester = requiredQuarter) then

BEGIN

update uosoffering

set Enrollment = Enrollment +1

where UoSCode =classNum and year = requiredYear and semester = requiredQuarter;

insert into transcript

value (studentID,classNum,requiredQuarter, requiredYear, NULL);

commit;

END;

end if;

END;

else

BEGIN

if not exists (select \*from requires where UoSCode = classNum and

PrereqUoSCode in (select distinct UoSCode

from uosoffering

where UoSCode not in(select UoSCode

from transcript

where StudId = studentID and (Grade is not NULL and Grade !="F")))) then

BEGIN

if exists (select \* from uosoffering where Enrollment< MaxEnrollment and UoSCode =classNum and year = requiredYear and semester = requiredQuarter) then

BEGIN

update uosoffering

set Enrollment = Enrollment +1

where UoSCode =classNum and year = requiredYear and semester = requiredQuarter;

insert into transcript

value (studentID,classNum,requiredQuarter, requiredYear, NULL);

commit;

END;

end if;

END;

else

BEGIN

select PrereqUoSCode from requires where UoSCode = classNum and

PrereqUoSCode not in (select UoSCode

from transcript

where StudId = studentID and (Grade is not NULL or Grade !="F"));

END;

end if;

END;

end if;

end