Tatiana Ensslin

CSE 581

**Lab 18: Transactions**

*Do this against YOUR own database.*

***For the entire lab, provide all of your SQL as text, inserted at the beginning of the document.***

***For each step also provide a screenshot of execution, showing the SQL and the results.***

***Please make sure both the SQL and the screenshots are marked w/ the question number that they are answering.***

**Steps:**

1. Modify[[1]](#footnote-1) the stored procedure from lab 12, so that way it will behave as follows (*use transactions to accomplish all of this behavior, and print a message stating “Transaction was rolled back”*):

* do the original checks/behaviors as before;
* *(assuming the checks above passed, and the grade was inserted)*, after the modification, **if** all of the students in the course have a grade, **calculate the average** for all of these students. If the average is below 50, **print out a message** “Error: Your grade average is too low. Please check the grades to make sure you have inserted the correct values.”, then **roll back the change** and **then** **quit**.

CREATE PROCEDURE UpdateGrades (@FacultyId AS VARCHAR(20),

@StudentId AS VARCHAR(20),

@CourseId AS INTEGER,

@NumericalGrade AS INTEGER)

as

DECLARE @Enrollment VARCHAR(20)

DECLARE @FinalGrade DECIMAL (5,2)

DECLARE @Answer DECIMAL (5,2)

DECLARE @AvgGrade DECIMAL (5,2)

DECLARE @Grades DECIMAL (5,2)

DECLARE @Sum DECIMAL (7,2)

DECLARE @Count INTEGER

DECLARE @OldGrade INTEGER

SELECT @Enrollment = (SELECT CE.StudentID

FROM CourseEnrollment CE

WHERE CE.StudentId = @StudentID AND CE.CourseID = @CourseID)

SELECT @FinalGrade = (SELECT FinalGrade

FROM CourseEnrollment CE

WHERE CE.CourseID = @CourseID AND CE.StudentID = @StudentID)

IF (@FacultyId = (SELECT Faculty

FROM Courses

WHERE CourseID = @CourseId))

BEGIN

IF @Enrollment IS NOT NULL

BEGIN

BEGIN TRAN Grade

UPDATE CourseENrollment

SET FinalGrade = @NumericalGrade

WHERE CourseID = @CourseID AND StudentID = @StudentID

BEGIN

BEGIN TRAN Average

DECLARE GradeAverageCursor CURSOR FOR

SELECT FinalGrade

FROM CourseEnrollment

WHERE StudentID = @StudentID

OPEN GradeAverageCursor

FETCH NEXT FROM GradeAverageCursor INTO @Grades

WHILE @@Fetch\_Status = 0

BEGIN

IF @Grades IS NULL

BEGIN

SELECT @Grades = 0

SELECT @Sum=@Sum+@Grades

END

ELSE

BEGIN

SELECT @Count = @Count + 1

SELECT @Sum = @Sum + @Grades

END

FETCH NEXT FROM GradeAverageCursor INTO @Grades

END

SELECT @AvgGrade = @Sum/@Count

CLOSE GradeAverageCursor

DEALLOCATE GradeAverageCursor

SELECT @Answer = @AvgGrade

IF @Answer < 50

BEGIN

COMMIT TRAN Average

RETURN @Answer

END

ELSE

BEGIN

ROLLBACK TRAN Average

PRINT 'Error: Your grade average is too low. Please check the grades to make sure you have inserted the correct values.';

END

END

IF @FinalGrade IS NOT NULL

BEGIN

COMMIT TRAN Grade

SET @OldGrade = (SELECT FinalGrade

FROM CourseEnrollment CE

WHERE CE.CourseId = @CourseId and CE.StudentId = @StudentId)

UPDATE CourseEnrollment

SET FinalGrade = @NumericalGrade

WHERE CourseId = @CourseId and StudentId = @StudentId

PRINT ' Success, with a warning- The Students exisiting grade ' + CAST(@OldGrade AS VARCHAR) + 'was changed to' + CAST(@NumericalGrade AS VARCHAR);

END

ELSE

BEGIN

ROLLBACK TRAN Grade

Print ' Success.'

END

END

ELSE

BEGIN

PRINT 'Error: The student is not taking the course you specified.';

END

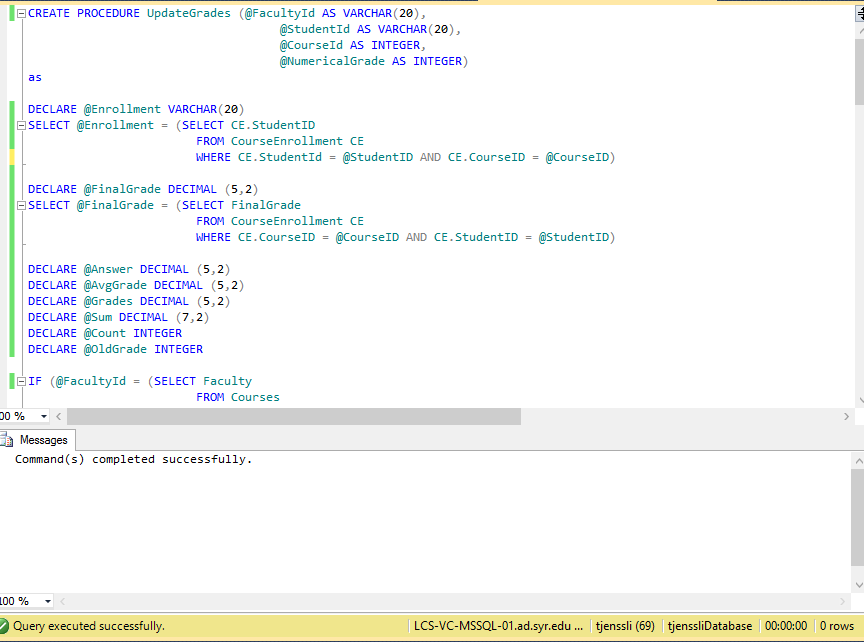
END

ELSE

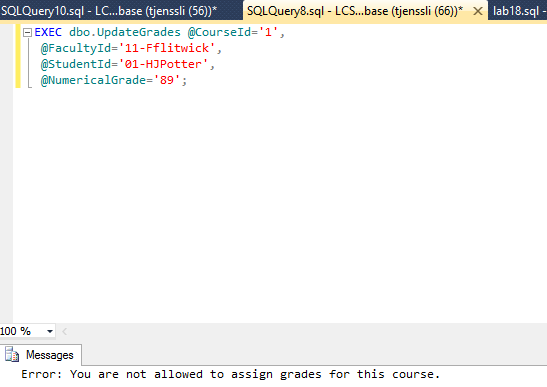
BEGIN

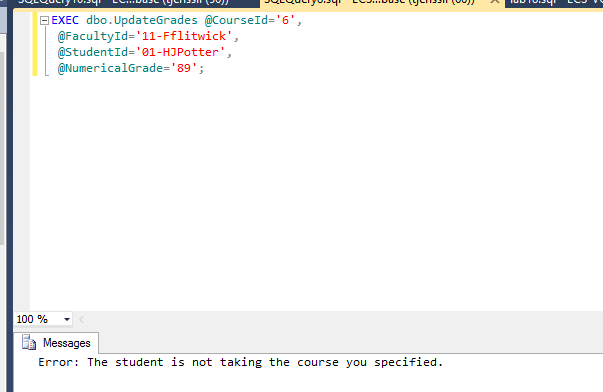
PRINT 'Error: You are not allowed to assign grades for this course.';

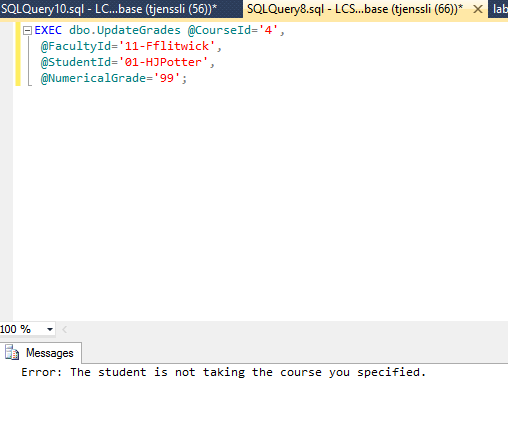
END



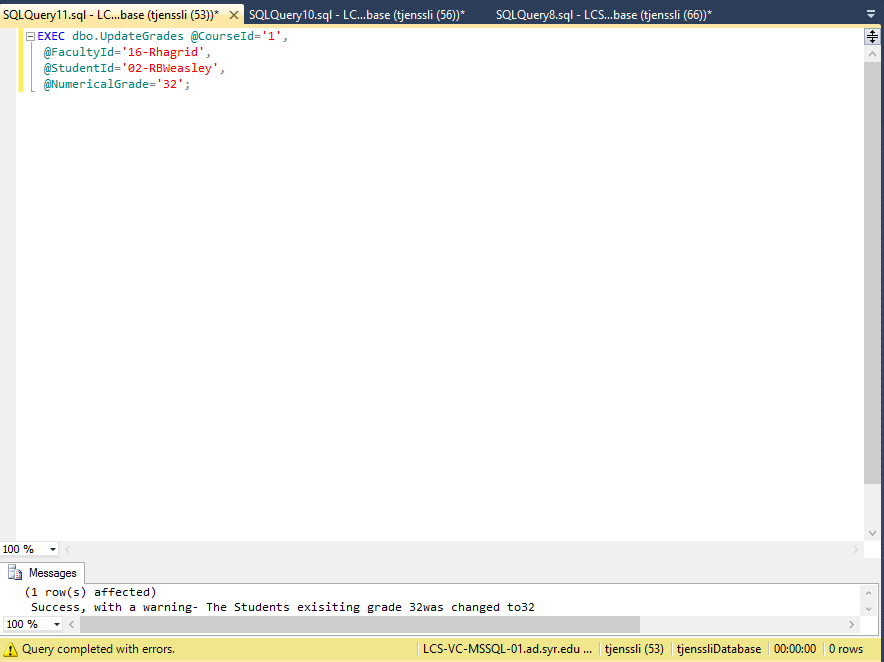
1. Prove2 that if (any) one of the original checks fails, it does so via a transaction3.

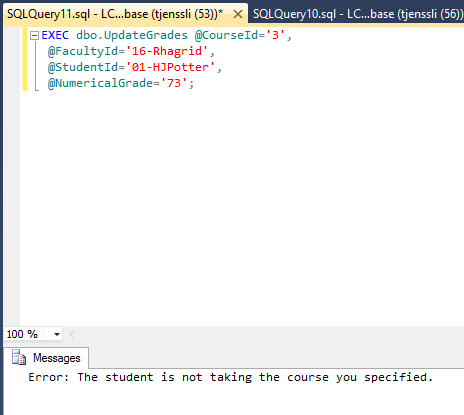






1. Prove2 that the new check fails when the conditions are met, and does so via a transaction3



1. Prove[[2]](#footnote-2) that a successful data modification works correctly[[3]](#footnote-3).

1. You can either ALTER, or DROP and CREATE [↑](#footnote-ref-1)
2. You have to prove to me that it works exactly as expected. That means you need to prove that the right messages were printed, and that the data was or was not changed, depending on the case. The only way you can prove this to me is via screenshots. [↑](#footnote-ref-2)
3. If you need to modify the data in order to be able to execute all of the cases, feel free to do so. I do not need to see these, just make sure that the data prior to the SP’s execution is showing the valid input. [↑](#footnote-ref-3)