University Course Enrollment: Tables and SQL Transactions Examples

Step 1: Table Creation

```
-- Students Table
CREATE TABLE Students (
    StudentID INT PRIMARY KEY,
    FirstName NVARCHAR(50),
    LastName NVARCHAR(50)
);

-- Courses Table
CREATE TABLE Courses (
    CourseID INT PRIMARY KEY,
    CourseName NVARCHAR(100)
);

-- Enrollments Table
CREATE TABLE Enrollments (
    EnrollmentID INT IDENTITY PRIMARY KEY,
    StudentID INT FOREIGN KEY REFERENCES Students(StudentID),
    CourseID INT FOREIGN KEY REFERENCES Courses(CourseID)
);
```

Step 2: Insert Sample Data

```
-- Insert students
INSERT INTO Students VALUES (1, 'Alice', 'Smith');
INSERT INTO Students VALUES (2, 'Bob', 'Johnson');
INSERT INTO Students VALUES (3, 'Charlie', 'Brown');

-- Insert courses
INSERT INTO Courses VALUES (101, 'Math 101');
INSERT INTO Courses VALUES (102, 'Intro to SQL');
INSERT INTO Courses VALUES (103, 'History 101');
```

Step 3: 10 Transaction Examples

1. Enroll a student in a course

```
BEGIN TRANSACTION;
BEGIN TRY
    INSERT INTO Enrollments (StudentID, CourseID) VALUES (1, 101);
    COMMIT;
END TRY
BEGIN CATCH
    ROLLBACK;
    PRINT 'Error: ' + ERROR_MESSAGE();
END CATCH;
```

2. Attempt to enroll a student in a course they're already enrolled in

```
BEGIN TRANSACTION;
BEGIN TRY
    INSERT INTO Enrollments (StudentID, CourseID) VALUES (1, 101);
    COMMIT;
END TRY
BEGIN CATCH
    ROLLBACK;
    PRINT 'Duplicate enrollment: ' + ERROR_MESSAGE();
END CATCH;
```

3. Enroll Bob in Intro to SQL

```
BEGIN TRANSACTION;
BEGIN TRY
    INSERT INTO Enrollments (StudentID, CourseID) VALUES (2, 102);
    COMMIT;
END TRY
BEGIN CATCH
    ROLLBACK;
    PRINT 'Error enrolling Bob: ' + ERROR_MESSAGE();
END CATCH;
```

4. Rollback manually if course is History 101

```
BEGIN TRANSACTION;
BEGIN TRY
    DECLARE @CourseID INT = 103;
    IF @CourseID = 103
    BEGIN
        PRINT 'Course not allowed, rolling back.';
        ROLLBACK;
    END
    ELSE
    BEGIN
        INSERT INTO Enrollments (StudentID, CourseID) VALUES (2, @CourseID);
        COMMIT;
    END
END TRY
BEGIN CATCH
    ROLLBACK;
    PRINT 'Error: ' + ERROR_MESSAGE();
END CATCH;
```

5. Delete all Bob's enrollments with rollback on error

```
BEGIN TRANSACTION;
BEGIN TRY
    DELETE FROM Enrollments WHERE StudentID = 2;
    COMMIT;
END TRY
BEGIN CATCH
    ROLLBACK;
    PRINT 'Error deleting enrollments: ' + ERROR_MESSAGE();
END CATCH;
```

6. Enroll Charlie in both Math 101 and SQL

```
BEGIN TRANSACTION;
BEGIN TRY
    INSERT INTO Enrollments (StudentID, CourseID) VALUES (3, 101);
    INSERT INTO Enrollments (StudentID, CourseID) VALUES (3, 102);
    COMMIT;
END TRY
BEGIN CATCH
    ROLLBACK;
    PRINT 'Error in multiple enrollments: ' + ERROR_MESSAGE();
END CATCH;
```

7. Simulate error with invalid course ID

```
BEGIN TRANSACTION;
BEGIN TRY
    INSERT INTO Enrollments (StudentID, CourseID) VALUES (1, 999); -- Invalid course
    COMMIT;
END TRY
BEGIN CATCH
    ROLLBACK;
    PRINT 'Invalid course enrollment: ' + ERROR_MESSAGE();
END CATCH;
```

8. Transfer enrollment from Math 101 to History 101

```
BEGIN TRANSACTION;
BEGIN TRY
    DELETE FROM Enrollments WHERE StudentID = 1 AND CourseID = 101;
    INSERT INTO Enrollments (StudentID, CourseID) VALUES (1, 103);
    COMMIT;
END TRY
BEGIN CATCH
    ROLLBACK;
    PRINT 'Transfer failed: ' + ERROR_MESSAGE();
END CATCH;
```

9. Simulate a failure and force rollback using RAISERROR

```
BEGIN TRANSACTION;
BEGIN TRY
    INSERT INTO Enrollments (StudentID, CourseID) VALUES (2, 103);
    RAISERROR('Force rollback for test.', 16, 1);
    COMMIT;
END TRY
BEGIN CATCH
    ROLLBACK;
    PRINT 'Transaction failed on purpose: ' + ERROR_MESSAGE();
END CATCH;
```

10. Check enrollment before inserting (preventing duplicates manually)

```
BEGIN TRANSACTION;
BEGIN TRY
    IF NOT EXISTS (
        SELECT 1 FROM Enrollments WHERE StudentID = 3 AND CourseID = 101
    )
    BEGIN
        INSERT INTO Enrollments (StudentID, CourseID) VALUES (3, 101);
        COMMIT;
    END
    ELSE
    BEGIN
        PRINT 'Already enrolled.';
        ROLLBACK;
    END
END TRY
BEGIN CATCH
    ROLLBACK;
    PRINT 'Error: ' + ERROR_MESSAGE();
END CATCH;
```