SQL Server Triggers and Stored Procedures

Table and Sample Data

```
-- Sample Table
CREATE TABLE Students (
    StudentID INT PRIMARY KEY,
    Name NVARCHAR(100),
    GPA DECIMAL(3,2),
    EnrollmentDate DATETIME DEFAULT GETDATE()
);

-- Insert Sample Data
INSERT INTO Students (StudentID, Name, GPA) VALUES
(1, 'Alice', 3.5),
(2, 'Bob', 2.8),
(3, 'Charlie', 3.9);
```

10 AFTER/FOR Triggers

Trigger 1 - Log Insert

```
CREATE TRIGGER trg_LogInsert
ON Students
AFTER INSERT
AS
BEGIN
    PRINT 'A new student has been inserted.';
END;
```

Trigger 2 - Log Update

```
CREATE TRIGGER trg_LogUpdate
ON Students
AFTER UPDATE
AS
BEGIN
    PRINT 'A student record has been updated.';
END;
```

Trigger 3 - Log Delete

```
CREATE TRIGGER trg_LogDelete
ON Students
AFTER DELETE
```

```
AS
BEGIN
    PRINT 'A student has been deleted.';
END;
```

Trigger 4 - Prevent GPA > 4.0

```
CREATE TRIGGER trg_ValidateGPA
ON Students
AFTER INSERT, UPDATE
AS
BEGIN
    IF EXISTS (SELECT 1 FROM inserted WHERE GPA > 4.0)
    BEGIN
        RAISERROR('GPA cannot exceed 4.0', 16, 1);
        ROLLBACK;
    END
END;
```

Trigger 5 - Track Enrollment

```
CREATE TABLE EnrollmentLog (StudentID INT, DateLogged DATETIME);
CREATE TRIGGER trg_TrackEnrollment
ON Students
AFTER INSERT
AS
BEGIN
    INSERT INTO EnrollmentLog
    SELECT StudentID, GETDATE() FROM inserted;
END;
```

Trigger 6 - Count Students

```
CREATE TRIGGER trg_CountStudents
ON Students
AFTER INSERT
AS
BEGIN
    DECLARE @Count INT;
    SELECT @Count = COUNT(*) FROM Students;
    PRINT 'Total students: ' + CAST(@Count AS NVARCHAR(10));
END;
```

Trigger 7 - Log GPA Changes

```
CREATE TABLE GPAChanges (StudentID INT, OldGPA DECIMAL(3,2), NewGPA DECIMAL(3,2), ChangedOn DATETIME);
CREATE TRIGGER trg_GPAChange
ON Students
AFTER UPDATE
AS
BEGIN
    INSERT INTO GPAChanges
    SELECT d.StudentID, d.GPA, i.GPA, GETDATE()
    FROM inserted i
    JOIN deleted d ON i.StudentID = d.StudentID
    WHERE i.GPA <> d.GPA;
END;
```

Trigger 8 - Protect Special Name

```
CREATE TRIGGER trg_NoTestName
ON Students
AFTER INSERT
AS
BEGIN
    IF EXISTS (SELECT 1 FROM inserted WHERE Name = 'Test Student')
    BEGIN
        RAISERROR('Cannot insert student named Test Student', 16, 1);
        ROLLBACK;
    END
END;
```

Trigger 9 - Log Deletes to Archive

```
CREATE TABLE ArchivedStudents (StudentID INT, Name NVARCHAR(100), DeletedAt DATETIME);
CREATE TRIGGER trg_ArchiveOnDelete
ON Students
AFTER DELETE
AS
BEGIN
    INSERT INTO ArchivedStudents
    SELECT StudentID, Name, GETDATE() FROM deleted;
END;
```

Trigger 10 - Auto Timestamp

```
ALTER TABLE Students ADD LastUpdated DATETIME;
CREATE TRIGGER trg_AutoTimestamp
```

```
ON Students
AFTER UPDATE
AS
BEGIN
    UPDATE s
    SET LastUpdated = GETDATE()
    FROM Students s
    JOIN inserted i ON s.StudentID = i.StudentID;
END;
```

10 Stored Procedures

Procedure 1 - Get All Students

```
CREATE PROCEDURE sp_GetAllStudents
AS
BEGIN
    SELECT * FROM Students;
END;
```

Procedure 2 - Get Student by ID

```
CREATE PROCEDURE sp_GetStudentByID @StudentID INT
AS
BEGIN
    SELECT * FROM Students WHERE StudentID = @StudentID;
END;
```

Procedure 3 - Insert New Student

```
CREATE PROCEDURE sp_InsertStudent
    @StudentID INT,
    @Name NVARCHAR(100),
    @GPA DECIMAL(3,2)
AS
BEGIN
    INSERT INTO Students (StudentID, Name, GPA)
    VALUES (@StudentID, @Name, @GPA);
END;
```

Procedure 4 - Update Student GPA

```
CREATE PROCEDURE sp_UpdateGPA
```

```
    @StudentID INT,
    @NewGPA DECIMAL(3,2)
AS
BEGIN
    UPDATE Students SET GPA = @NewGPA WHERE StudentID = @StudentID;
END;
```

Procedure 5 - Delete Student

```
CREATE PROCEDURE sp_DeleteStudent
    @StudentID INT
AS
BEGIN
    DELETE FROM Students WHERE StudentID = @StudentID;
END;
```

Procedure 6 - Count Students

```
CREATE PROCEDURE sp_CountStudents
AS
BEGIN
    SELECT COUNT(*) AS TotalStudents FROM Students;
END;
```

Procedure 7 - Get Students Above GPA

```
CREATE PROCEDURE sp_StudentsAboveGPA
    @MinGPA DECIMAL(3,2)
AS
BEGIN
    SELECT * FROM Students WHERE GPA > @MinGPA;
END;
```

Procedure 8 - Log Custom Message

```
CREATE PROCEDURE sp_LogMessage
    @Message NVARCHAR(100)
AS
BEGIN
    PRINT @Message;
END;
```

SQL Server Triggers and Stored Procedures

Procedure 9 - Get Recent Enrollments

```
CREATE PROCEDURE sp_RecentEnrollments
AS
BEGIN
    SELECT * FROM Students WHERE EnrollmentDate >= DATEADD(DAY, -7, GETDATE());
END;
```

Procedure 10 - Bulk GPA Update

```
CREATE PROCEDURE sp_BulkGPAUpdate
    @Delta DECIMAL(3,2)
AS
BEGIN
    UPDATE Students SET GPA = GPA + @Delta;
END;
```