# Student Allocation Using SQL Stored Procedure

## Objective

To allocate elective subjects to students based on their GPA and subject preferences using a stored procedure in SQL Server. Students with higher GPA are given priority during allocation. If none of the five preferences can be fulfilled for a student, they are marked as unallotted.

## Database Tables

### 1. StudentDetails

Columns:  
- StudentId (VARCHAR): Unique ID of the student  
- StudentName (VARCHAR): Name of the student  
- GPA (FLOAT): Grade Point Average  
- Branch (VARCHAR): Department  
- Section (VARCHAR): Section

### 2. SubjectDetails

Columns:  
- SubjectId (VARCHAR): Unique subject ID  
- SubjectName (VARCHAR): Name of the subject  
- MaxSeats (INT): Maximum number of seats  
- RemainingSeats (INT): Remaining seats for allotment

### 3. StudentPreference

Columns:  
- StudentId (VARCHAR): FK to StudentDetails  
- SubjectId (VARCHAR): FK to SubjectDetails  
- Preference (INT): Preference number (1 to 5)

### 4. Allotments

Columns:  
- SubjectId (VARCHAR): FK to SubjectDetails  
- StudentId (VARCHAR): FK to StudentDetails

### 5. UnallotedStudents

Columns:  
- StudentId (VARCHAR): FK to StudentDetails for unallocated students

## Stored Procedure: AllocateSubjects

Below is the full stored procedure that handles the allocation logic:

IF OBJECT\_ID('AllocateSubjects', 'P') IS NOT NULL  
 DROP PROCEDURE AllocateSubjects;  
GO  
  
CREATE PROCEDURE AllocateSubjects  
AS  
BEGIN  
 SET NOCOUNT ON;  
  
 DELETE FROM Allotments;  
 DELETE FROM UnallotedStudents;  
  
 DECLARE @StudentId VARCHAR(20), @SubjectId VARCHAR(20), @RemainingSeats INT;  
 DECLARE @Preference INT;  
  
 DECLARE student\_cursor CURSOR FOR  
 SELECT StudentId FROM StudentDetails ORDER BY GPA DESC;  
  
 OPEN student\_cursor;  
 FETCH NEXT FROM student\_cursor INTO @StudentId;  
  
 WHILE @@FETCH\_STATUS = 0  
 BEGIN  
 DECLARE @Allotted BIT = 0;  
 SET @Preference = 1;  
  
 WHILE @Preference <= 5 AND @Allotted = 0  
 BEGIN  
 SELECT @SubjectId = SubjectId  
 FROM StudentPreference  
 WHERE StudentId = @StudentId AND Preference = @Preference;  
  
 IF @SubjectId IS NOT NULL  
 BEGIN  
 SELECT @RemainingSeats = RemainingSeats  
 FROM SubjectDetails  
 WHERE SubjectId = @SubjectId;  
  
 IF @RemainingSeats > 0  
 BEGIN  
 INSERT INTO Allotments (SubjectId, StudentId)  
 VALUES (@SubjectId, @StudentId);  
  
 UPDATE SubjectDetails  
 SET RemainingSeats = RemainingSeats - 1  
 WHERE SubjectId = @SubjectId;  
  
 SET @Allotted = 1;  
 END  
 END  
  
 SET @Preference = @Preference + 1;  
 END  
  
 IF @Allotted = 0  
 BEGIN  
 INSERT INTO UnallotedStudents (StudentId)  
 VALUES (@StudentId);  
 END  
  
 FETCH NEXT FROM student\_cursor INTO @StudentId;  
 END  
  
 CLOSE student\_cursor;  
 DEALLOCATE student\_cursor;  
END;  
GO

## Steps to Execute

1. Populate all five tables with required data.  
2. Run the procedure using:  
EXEC AllocateSubjects;  
3. View results using:  
SELECT \* FROM Allotments;  
SELECT \* FROM UnallotedStudents;

## Verification Checklist

- Do all students have 5 preferences?  
- Are remaining seats properly decreased?  
- Are students allotted based on GPA?  
- Do unallotted students appear in UnallotedStudents?  
- No subject has over-allotment (RemainingSeats < 0)?

## Sample Result View

SELECT   
 s.StudentId, s.StudentName, s.GPA, a.SubjectId, sd.SubjectName  
FROM Allotments a  
JOIN StudentDetails s ON a.StudentId = s.StudentId  
JOIN SubjectDetails sd ON a.SubjectId = sd.SubjectId  
ORDER BY s.GPA DESC;