```
Create procedure InsertStudents (@StudentID Varchar(10),

@StudentName Varchar(50),


@Gender Varchar (6), @DOB Date,

@DepartmentID Varchar (10), @Email Varchar (100))

As

Begin

Insert into Students (StudentID, Name, Gender, DOB, DepartmentID,
Email)

Values(@StudentID, @StudentName, @Gender, @DOB,
@DepartmentID, @Email)

End;
```

The **InsertStudents** stored procedure is designed to add new students records into the students table in our students database. It provides an efficient way to insert all necessary information in a single operation.

Parameters

The parameters declared for this procedure include:

@StudentID: This parameter is intended to store the unique identifier for each new student and requires string values, such as 'S156'.

@StudentName: This parameter stores the full name of every student and requires string values to be provided.

@Gender: This stores the gender of the student

@DOB: This stores the student's date of birth and would require dates in standardized formats YYY-MM-DD.

@ DepartmentID: This parameter specifies the identifier of the department each student belongs to.

@Email: This parameter stores the student's email address and requires string values.

The **InsertStudents** procedure is run using **EXEC InsertStudents** by taking the values provided and inserting them as a new row into the **Students** table.

```
Create procedure NewEnrollment (@EnrollmentID Varchar (10),
@StudentID Varchar (10), @CourseID Varchar (10), @EnrollmentDate
Date)

As

Begin

        Insert into Enrollments (EnrollmentID, StudentID, CourseID,
EnrollmentDate)

        Values (@EnrollmentID, @StudentID, @CourseID,
@EnrollmentDate)

End;
```

The **NewEnrollment** procedure inserts a new course enrollment for a specific student on the **Enrollments** table.

The parameters declared include:

@EnrollmentID, @StudentID, @CourseID & @ENrollmentDate. These parameters specify the unique identifier for each course enrollment, the unique student and course identifier, and the date on which the enrollment record is created.

```
Create procedure StudentsInCourse (@CourseCode Varchar(10))
As
Begin
        Select Distinct S.StudentID, S.Name, C.CourseName,
        S.DepartmentID
        From Students S
        Join Enrollments E
        on S.StudentID = E.StudentID
        Join Courses C
        on E.CourseID = C.CourseID
        Where C.CourseID = @CourseCode
        Order by S.StudentID
End;
```

The **StudentsInCourse** stored procedure returns a list of students enrolled in a specific course. The parameter declared in this procedure is the **@CourseID,** which serves as a reference for whatever CourseID is specified in the **EXEC StudentsInCourse** call statement. A sample call statement would appear like this - **Exec StudentsInCourse @CourseCode = 'PSY351'** and return an output as shown below

	StudentID	Name	CourseName	DepartmentID	EnrollmentID
1	S3017	Morgan Smith	Research on social interactions	DPT-SOC	E163
2	S3019	Susan Knight	Research on social interactions	DPT-POL	E190
3	S3029	Flora Shaw	Research on social interactions	DPT-EDU	E181

```
Create Procedure Studentstaught (@InstructorName Varchar (50))
As
Begin
Select I.InstructorID, InstructorName,
Count (Distinct S.StudentID) As TotalStudents,
STRING_AGG(S.Name, ',') As StudentsTaught,
STRING_AGG(S.StudentID, ',') As StudentsID
From Instructors I
Join Courses C
on I.InstructorID = C.InstructorID
Join Enrollments E
on C.CourseID = E.CourseID
Join Students S
on E.StudentID = S.StudentID
Where InstructorName = @InstructorName
Group by I.InstructorID, InstructorName
Order by TotalStudents;
End;
```

The **Studentstaught** stored procedure returns a list of students grouped by the instructor who taught them. The declared parameter is **@InstructorName,** which serves as a reference in the **EXEC** statement to identify the instructor whose list of students should be retrieved. The statement **Exec Studentstaught @InstructorName = 'Professor Bankole'** will return the output below.

	InstructorID	InstructorName	TotalStudents	StudentsTaught	StudentsID
1	I1013	Professor Bankole	6	Labyb Adejuwon,Favour Taiwo,Robin Ahmed,Robin Ah...	S2013,S2021,S3013,S3013,S2037,S3027,S3023

```
Create procedure UpdateStudentMail (@StudentID Varchar(10), @Email Varchar (100))
As
Begin
Update Students
Set Email = @Email
Where StudentID = @StudentID
End;
```

The **UpdateStudentMail** stored procedure updates preexisting student records by inserting the email address for the specified student using the declared parameters **@StudentID & @Email.** The statement **Exec UpdateStudentMail 'S3041', 'Wlyon@stu.edu.com'** will insert the email provided to the student with id **S3041.**

```sql
Create procedure StudentCourseEnrollment (@StudentID Varchar (10))
As
Begin
    Select s.StudentID, s.Name AS StudentName,
    Count(E.CourseID) AS NumberOfCoursesEnrolled,
    String_agg(C.CourseName, ', ') AS CoursesTaken
from Students As s
Join Enrollments As e
on s.StudentID = e.StudentID
Join Courses As c
on e.CourseID = c.CourseID
Where s.StudentID = @StudentID
Group by s.StudentID, s.Name
Order by NumberOfCoursesEnrolled;
End;
```

The **StudentCourseEnrollment** stored procedure returns a list of all courses enrolled in by a specific student using the parameter **@StudentID.**

The call statement **Exec StudentCourseEnrollment @StudentID = 'S3029'** will return the output in the diagram below.

	StudentID	StudentName	NumberOfCoursesEnrolled	CoursesTaken
1	S3029	Flora Shaw	2	Entreprenurial skills, Research on social intera...

```sql
Create Procedure CourseEnrollment (@CourseID Varchar (10))
As
Begin
    Select C.CourseID, CourseName , Count(E.courseID) As EnrollmentCount
    From Courses C
    Join Enrollments E
    on C.CourseID = E.CourseID
    Where C.CourseID = @CourseID
    Group by C.CourseID, CourseName
    Order by EnrollmentCount
End;
```

The **CourseEnrollment** stored procedure shows a count of students enrolled in a specific course. The parameter declared is **@StudentID,** which specifies which course's enrollment count should be retrieved.

The call statement **Exec CourseEnrollment @CourseID = 'Eco302'** will return the output below.

	CourseID	CourseName	EnrollmentCount
1	ECO302	Microeconomics II	2