**1. Introduction to Stored Procedures**

A **stored procedure** is a precompiled collection of one or more SQL statements stored in a database.  
Instead of sending multiple SQL queries from the application, the client can call the procedure by name, passing parameters if needed.

**Purpose:** To improve performance, security, and maintainability of database operations.

**2. Why Use Stored Procedures in ASP.NET Core Projects**

|  |  |
| --- | --- |
| **Feature** | **Explanation** |
| **Performance** | Stored procedures are precompiled and optimized by SQL Server. |
| **Security** | Prevents SQL injection as parameters are handled safely. |
| **Reusability** | Common database logic can be reused across applications. |
| **Maintainability** | Easier to update database logic without modifying application code. |
| **Separation of Concerns** | Keeps business logic separate from data access logic. |

**3. Structure of a Stored Procedure**

**Syntax:**

CREATE PROCEDURE procedure\_name

@parameter1 datatype,

@parameter2 datatype = default\_value

AS

BEGIN

-- SQL Statements

END;

**Example:**

CREATE PROCEDURE sp\_GetStudents

AS

BEGIN

SELECT \* FROM Students;

END;

To execute:

EXEC sp\_GetStudents;

**4. Creating Stored Procedures for StudentCourseDB**

Below are examples of CRUD stored procedures used in the Student Management API.

**1. Get All Students**

CREATE PROCEDURE sp\_GetStudents

AS

BEGIN

SELECT s.Id, s.Name, s.Age, s.Grade, c.CourseName

FROM Students s

INNER JOIN Courses c ON s.CourseId = c.CourseId;

END;

**2. Get Student by ID**

CREATE PROCEDURE sp\_GetStudentById

@Id INT

AS

BEGIN

SELECT \* FROM Students WHERE Id = @Id;

END;

**3. Add Student**

CREATE PROCEDURE sp\_AddStudent

@Name NVARCHAR(100),

@Age INT,

@Grade NVARCHAR(10),

@CourseId INT

AS

BEGIN

INSERT INTO Students (Name, Age, Grade, CourseId)

VALUES (@Name, @Age, @Grade, @CourseId);

END;

**4. Update Student**

CREATE PROCEDURE sp\_UpdateStudent

@Id INT,

@Name NVARCHAR(100),

@Age INT,

@Grade NVARCHAR(10),

@CourseId INT

AS

BEGIN

UPDATE Students

SET Name = @Name, Age = @Age, Grade = @Grade, CourseId = @CourseId

WHERE Id = @Id;

END;

**5. Delete Student**

CREATE PROCEDURE sp\_DeleteStudent

@Id INT

AS

BEGIN

DELETE FROM Students WHERE Id = @Id;

END;

**5. Integrating Stored Procedures in ASP.NET Core (EF Core)**

There are multiple ways to call stored procedures in ASP.NET Core using EF Core.  
The most common methods are:

**1. Using FromSqlRaw()**

Example to call stored procedure that returns data:

var students = \_context.Students

.FromSqlRaw("EXEC sp\_GetStudents")

.ToList();

**2. Using Database.ExecuteSqlRaw()**

Used for stored procedures that do not return data (like Insert/Update/Delete).

Example:

\_context.Database.ExecuteSqlRaw(

"EXEC sp\_AddStudent @p0, @p1, @p2, @p3",

new object[] { name, age, grade, courseId });

**3. Using SqlParameter for Safety**

var idParam = new SqlParameter("@Id", 2);

var student = \_context.Students

.FromSqlRaw("EXEC sp\_GetStudentById @Id", idParam)

.ToList();

**6. API Controller Integration Example**

Example StudentsController methods using stored procedures:

[ApiController]

[Route("api/[controller]")]

public class StudentsController : ControllerBase

{

private readonly StudentCourseContext \_context;

public StudentsController(StudentCourseContext context)

{

\_context = context;

}

[HttpGet]

public IActionResult GetAllStudents()

{

var students = \_context.Students.FromSqlRaw("EXEC sp\_GetStudents").ToList();

return Ok(students);

}

[HttpGet("{id}")]

public IActionResult GetStudentById(int id)

{

var idParam = new SqlParameter("@Id", id);

var student = \_context.Students.FromSqlRaw("EXEC sp\_GetStudentById @Id", idParam).FirstOrDefault();

return Ok(student);

}

[HttpPost]

public IActionResult AddStudent(Student s)

{

\_context.Database.ExecuteSqlRaw(

"EXEC sp\_AddStudent @p0, @p1, @p2, @p3",

s.Name, s.Age, s.Grade, s.CourseId);

return Ok("Student added successfully.");

}

[HttpPut("{id}")]

public IActionResult UpdateStudent(int id, Student s)

{

\_context.Database.ExecuteSqlRaw(

"EXEC sp\_UpdateStudent @p0, @p1, @p2, @p3, @p4",

id, s.Name, s.Age, s.Grade, s.CourseId);

return Ok("Student updated successfully.");

}

[HttpDelete("{id}")]

public IActionResult DeleteStudent(int id)

{

\_context.Database.ExecuteSqlRaw("EXEC sp\_DeleteStudent @p0", id);

return Ok("Student deleted successfully.");

}

}

**7. Advantages of Using Stored Procedures in EF Core**

1. **Performance** – SQL Server precompiles and caches the query plan.
2. **Security** – Parameters prevent SQL injection.
3. **Simpler Maintenance** – Database logic can be changed independently.
4. **Reduced Data Transfer** – Only the required data is sent back.
5. **Logging and Auditing** – Stored procedures can include tracking logic.

**8. Common Issues & Fixes**

|  |  |  |
| --- | --- | --- |
| **Issue** | **Reason** | **Solution** |
| Invalid object name | Database context connected to wrong DB | Check connection string |
| Procedure not found | Procedure not created or name mismatch | Verify stored procedure name in SQL Server |
| Parameter count mismatch | Incorrect number of parameters | Match parameters exactly as defined |
| No data returned | Result set not mapped | Check your entity model and SQL SELECT columns |

**Mini Exercise**

* Create a stored procedure to retrieve students **above a certain age**.
* Call it in your controller using FromSqlRaw.

**Snapshots :**

A screenshot of a computer

AI-generated content may be incorrect.

StudentCourseDB created successfully in SQL Server.

A screenshot of a computer

AI-generated content may be incorrect.

Students and Courses tables created with sample data in SSMS.

A screenshot of a computer program

AI-generated content may be incorrect.

All stored procedures (CRUD) created and visible in SSMS under Programmability → Stored Procedures.

A screenshot of a computer

AI-generated content may be incorrect.

ASP.NET Core Web API project structure showing Controllers, Models, Data, and Program.cs files.

A screen shot of a computer

AI-generated content may be incorrect.

Connection string configured in appsettings.json to link API with SQL Server database.

A screenshot of a computer

AI-generated content may be incorrect.

StudentCourseContext.cs showing DbSet<Student> and DbSet<Course> for EF Core integration.

A screen shot of a computer program

AI-generated content may be incorrect.

StudentsController configured to call SQL stored procedures using FromSqlRaw and ExecuteSqlRaw.

A screenshot of a computer program

AI-generated content may be incorrect.

EF Core migration and database update commands executed successfully in terminal.

A screenshot of a computer

AI-generated content may be incorrect.

GET /api/students tested successfully in Swagger fetching student data from stored procedure.

A screenshot of a computer

AI-generated content may be incorrect.

POST /api/students adds a new student record to SQL database.

A screenshot of a computer

AI-generated content may be incorrect.DELETE /api/students/{id} successfully deletes a record using stored procedure.