

# Dean H - Employee Performance Management System

## Design Document

<b>Introduction</b>	<b>2</b>
<b>Key Features</b>	<b>2</b>
Tools and Technologies	2
<b>System Architecture</b>	<b>3</b>
Presentation Layer	3
Application Layer	3
Data Layer	3
Scalability and Performance Considerations	3
<b>Database Design</b>	<b>4</b>
<b>End Database Design SQL View</b>	<b>5</b>
<b>SQL Database Creation</b>	<b>6</b>
Table Creation	6
Inserting Data	7
<b>Stored Procedures SQL</b>	<b>11</b>
Insert Employee Info	11
Insert Performance Review	11
Insert Department Info	12
Insert Employee Department Info	12
Add New Employee	12
Update Employee Details	14
Get Employee Performance Reviews	15
Get Top 5 Employees	15
Get Employees That Need a Performance Review	16
<b>Triggers SQL</b>	<b>17</b>
Trigger After Insert Performance Review	17
Trigger After Update Performance Review	17

## **Introduction**

The Employee Performance Management System is an ASP.NET Core MVC application designed to streamline the management of employee records and performance evaluations within an organization. By leveraging Microsoft SQL Server, Entity Framework Core, and SSRS, the system provides a comprehensive solution for tracking employee data, conducting performance reviews, and generating insightful reports. This application aims to enhance HR processes and decision-making by offering an intuitive interface for data management and analysis.

## **Key Features**

- The Employee Records Management feature utilizes SQL Server to manage employee details and departmental assignments. This information is dynamically displayed on Razor pages, providing a user-friendly interface for viewing team details.
- The Performance Reviews feature utilizes SQL Server to record and manage employee performance evaluations. This information is dynamically displayed on Razor pages, providing a user-friendly interface for tracking and analyzing employee performance over time.
- The Department Management feature utilizes SQL Server to manage department details and performance metrics. This information is dynamically displayed on Razor pages, providing a user-friendly interface for monitoring and analyzing department performance.
- The Data Security and Role-Based Access Control feature ensures secure access to sensitive data by implementing role-based access control (RBAC). It utilizes SQL Server to enforce access permissions and maintain data integrity, adhering to proper database design principles. This ensures that only authorized users can view and modify sensitive data, promoting data integrity and consistency.

## **Tools and Technologies**

ASP.NET Core MVC - For building the web application.

Entity Framework Core - For database operations.

Razor - For rendering dynamic HTML content.

SQL Server 2022 Developer Edition - For database management.

SQL Server Reporting Services (SSRS) - For creating and displaying reports.

Javascript - For dynamic displaying data

## **System Architecture**

The Employee Performance Management System follows a three-tier architecture, comprising the presentation layer, the application layer, and Data Layer. Each layer plays a distinct role in ensuring the functionality and performance of the system.

### **Presentation Layer**

The presentation layer encompasses the user interface components responsible for displaying employee data, performance reviews, and department information. This layer includes web pages, views, and client-side scripts that facilitate user interaction with the application. Designed to be intuitive and user-friendly, the interface allows users to easily navigate through employee records and performance metrics.

### **Application Layer**

Serving as the intermediary between the presentation layer and the data layer, the application layer handles business logic and data processing tasks. It comprises controllers, business logic components, and service layers responsible for processing user requests, fetching data from the data layer, and orchestrating the flow of information within the application. This layer ensures seamless integration of employee data and performance metrics retrieved from the data layer.

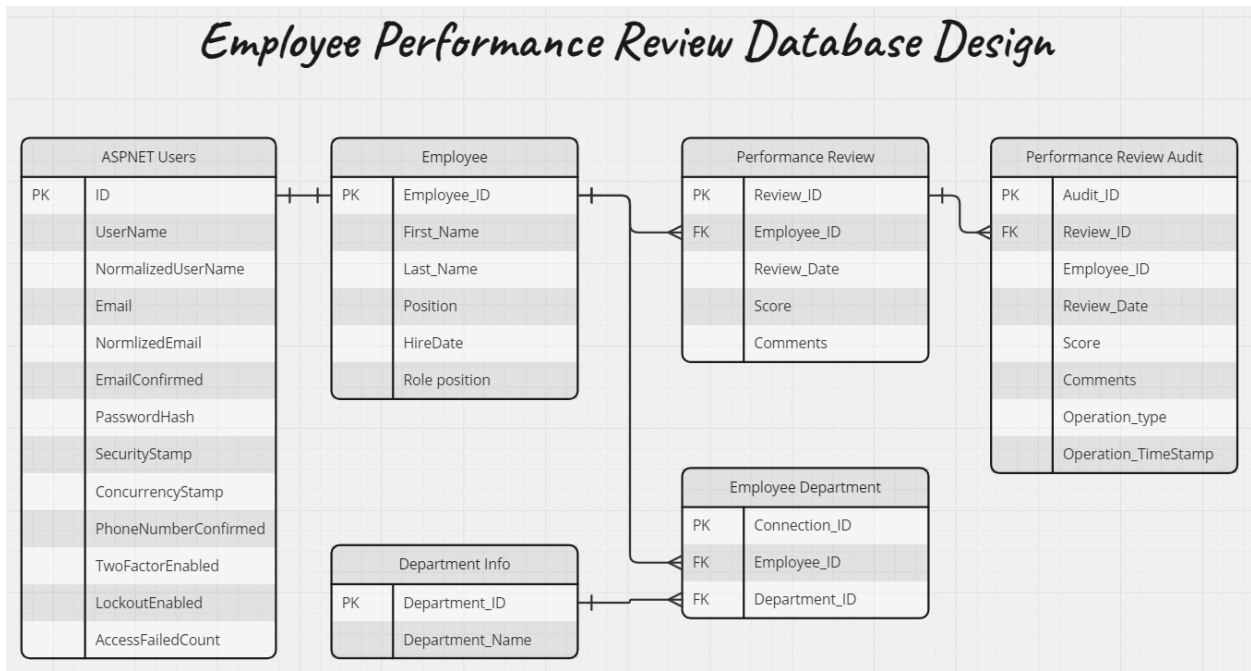
### **Data Layer**

The data layer is responsible for storing and managing persistent data used by the application. It typically consists of a relational database management system (RDBMS), such as SQL Server, which stores employee records, performance reviews, department information, and other relevant data. The data layer ensures data integrity, consistency, and security through proper database design, including the definition of tables, relationships, and constraints. It also provides mechanisms for data retrieval, modification, and deletion, enabling efficient access to and manipulation of data by the application layer.















































### **Scalability and Performance Considerations**

The system architecture of the application is designed to be scalable and performant, capable of handling increasing loads and user interactions. Horizontal scalability is achieved through load balancing and auto-scaling mechanisms, allowing the application to dynamically allocate resources based on demand. Performance optimizations, such as efficient database queries and caching strategies, are implemented to minimize latency and improve responsiveness. Additionally, asynchronous processing and parallelization techniques are employed to enhance throughput and resource utilization, ensuring optimal performance under varying workloads.

Database Design



## End Database Design SQL View

- [-]  EmployeePerformanceDB
  - [+]  Database Diagrams
  - [-]  Tables
    - [+]  System Tables
    - [+]  FileTables
    - [+]  External Tables
    - [+]  Graph Tables
    - [+]  dbo.AspNetRoleClaims
    - [+]  dbo.AspNetRoles
    - [+]  dbo.AspNetUserClaims
    - [+]  dbo.AspNetUserLogins
    - [+]  dbo.AspNetUserRoles
    - [+]  dbo.AspNetUsers
    - [+]  dbo.AspNetUserTokens
    - [+]  Department.DepartmentInfo
    - [+]  Department.EmployeeDepartment
    - [+]  Employee.EmployeeInfo
    - [-]  Employee.PerformanceReview
      - [+]  Columns
      - [+]  Keys
      - [+]  Constraints
      - [-]  Triggers
        -  Trigger\_AfterInsert\_PerformanceReview
        -  Trigger\_AfterUpdate\_PerformanceReview
      - [+]  Indexes
      - [+]  Statistics
    - [+]  Employee.PerformanceReviewAudit
  - [+]  Views
  - [+]  External Resources
  - [+]  Synonyms
  - [-]  Programmability
    - [-]  Stored Procedures
      - [+]  System Stored Procedures
      - [+]  dbo.AddNewEmployee
      - [+]  dbo.GetEmployeeDueForReview
      - [+]  dbo.GetEmployeePerformanceReview
      - [+]  dbo.GetTop5EmployeesAverageScore
      - [+]  dbo.InsertDepartmentInfo
      - [+]  dbo.InsertEmployeeDepartmentInfo
      - [+]  dbo.InsertEmployeeInfo
      - [+]  dbo.InsertPerformanceReview
      - [+]  dbo.UpdateEmployeeDetails
      - [+]  dbo.UpdatePerformanceReview
    - [+]  Functions
    - [+]  Database Triggers
    - [+]  Assemblies

# SQL Database Creation

## Table Creation

```
CREATE SCHEMA Employee;
go

CREATE SCHEMA Department;
go

-- Create EmployeeInfo Table
CREATE TABLE Employee.EmployeeInfo (
    employee_id INT IDENTITY (1,1) PRIMARY KEY,
    first_name VARCHAR(255) NOT NULL,
    last_name VARCHAR(255) NOT NULL,
    position VARCHAR(255) NOT NULL,
    role_position VARCHAR(255) NOT NULL,
    hire_date DATE NOT NULL
);

-- Create PerformanceReview Table
CREATE TABLE Employee.PerformanceReview (
    review_id INT IDENTITY (1,1) PRIMARY KEY,
    employee_id INT NOT NULL,
    review_date DATE NOT NULL,
    score INT NOT NULL,
    comments VARCHAR(255),
    CONSTRAINT FK_Employee_PerformanceReview FOREIGN KEY (employee_id)
        REFERENCES Employee.EmployeeInfo(employee_id) ON DELETE CASCADE
);

-- Create DepartmentInfo Table
CREATE TABLE Department.DepartmentInfo (
    department_id INT IDENTITY (1,1) PRIMARY KEY,
    department_name VARCHAR(255) NOT NULL
);

-- Create EmployeeDepartment Table
CREATE TABLE Department.EmployeeDepartment (
    connection_id INT IDENTITY (1,1) PRIMARY KEY,
    employee_id INT NOT NULL,
    department_id INT NOT NULL,
    CONSTRAINT FK_Employee_EmployeeDepartment FOREIGN KEY (employee_id)
        REFERENCES Employee.EmployeeInfo(employee_id) ON DELETE CASCADE,
    CONSTRAINT FK_Department_EmployeeDepartment FOREIGN KEY (department_id)
        REFERENCES Department.DepartmentInfo(department_id) ON DELETE CASCADE
);

CREATE TABLE Employee.PerformanceReviewAudit (
    audit_id INT IDENTITY (1,1) PRIMARY KEY,
    review_id INT NOT NULL,
    employee_id INT NOT NULL,
    review_date DATE NOT NULL,
    score INT NOT NULL,
    comments VARCHAR(255),
    operation_type VARCHAR(10) NOT NULL, -- 'INSERT' or 'UPDATE'
    operation_timestamp DATETIME DEFAULT GETDATE(),
    CONSTRAINT FK_Employee_PerformanceReviewAudit FOREIGN KEY (review_id)
        REFERENCES Employee.PerformanceReview(review_id) ON DELETE CASCADE
);
```

## Inserting Data

```
use EmployeePerformanceDB;
go

SET IDENTITY_INSERT Employee.EmployeeInfo ON;
go

CREATE PROCEDURE InsertEmployeeInfo
    @EmployeeID INT,
    @FirstName VARCHAR(255),
    @LastName VARCHAR(255),
    @Position VARCHAR(255),
    @HireDate DATE,
    @UserName VARCHAR(255),
    @Role VARCHAR(255)
AS
BEGIN
    INSERT INTO Employee.EmployeeInfo (employee_id, first_name, last_name, position, role_position, hire_date)
    VALUES (@EmployeeID, @FirstName, @LastName, @Position, @Role, @HireDate);

    -- Insert user into AspNetUsers
    -- Default Password (YourSecurePassword123!)
    INSERT INTO AspNetUsers (Id, UserName, NormalizedUserName, Email, NormalizedEmail, EmailConfirmed, PasswordHash,
    SecurityStamp, ConcurrencyStamp, PhoneNumberConfirmed, TwoFactorEnabled, LockoutEnabled, AccessFailedCount)
    VALUES (@EmployeeID, @UserName, UPPER(@UserName), @UserName + '@example.com', UPPER(@UserName + '@example.com'), 1,
    'AQAAAAIAAYagAAAAEC9LifLJYADjCODAqX2wGnmCnA1cACbMkhsEArIPy4ZXvRVG5fZCieOeU51wqjTohg==',
    NEWID(), NEWID(), 0, 0, 0, 0);

    -- Insert user role into AspNetUserRoles
    DECLARE @UserId NVARCHAR(450) = (SELECT Id FROM AspNetUsers WHERE UserName = @UserName);
    DECLARE @RoleId NVARCHAR(450) = (SELECT Id FROM AspNetRoles WHERE Name = @Role);

    IF @RoleId IS NULL
    BEGIN
        -- Insert role into AspNetRoles if it doesn't exist
        SET @RoleId = NEWID();
        INSERT INTO AspNetRoles (Id, Name, NormalizedName, ConcurrencyStamp)
        VALUES (@RoleId, @Role, UPPER(@Role), NEWID());
    END

    INSERT INTO AspNetUserRoles (UserId, RoleId)
    VALUES (@UserId, @RoleId);
END;
go
-- TEAM A
```

```

EXEC InsertEmployeeInfo 1, 'John', 'Doe', 'Tech Lead', '2022-01-15', 'john.doe', 'Manager';
EXEC InsertEmployeeInfo 2, 'Jane', 'Smith', 'Software Developer', '2022-03-22', 'jane.smith', 'Employee';
EXEC InsertEmployeeInfo 3, 'Alice', 'Johnson', 'Software Developer', '2022-06-10', 'alice.johnson', 'Employee';
EXEC InsertEmployeeInfo 4, 'Bob', 'Williams', 'Software Developer', '2022-08-05', 'bob.williams', 'Employee';
EXEC InsertEmployeeInfo 5, 'Charlie', 'Brown', 'Software Developer', '2022-10-18', 'charlie.brown', 'Employee';
EXEC InsertEmployeeInfo 6, 'David', 'Jones', 'Software Developer', '2023-02-14', 'david.jones', 'Employee';
EXEC InsertEmployeeInfo 7, 'Eva', 'Miller', 'Software Developer', '2023-04-30', 'eva.miller', 'Employee';
EXEC InsertEmployeeInfo 8, 'Frank', 'Davis', 'Software Developer', '2023-07-21', 'frank.davis', 'Employee';

-- TEAM B
EXEC InsertEmployeeInfo 9, 'Grace', 'Hall', 'Tech Lead', '2023-09-12', 'grace.hall', 'Manager';
EXEC InsertEmployeeInfo 10, 'Hannah', 'Moore', 'Software Developer', '2023-09-12', 'hannah.moore', 'Employee';
EXEC InsertEmployeeInfo 11, 'Isaac', 'Clark', 'Software Developer', '2024-01-15', 'isaac.clark', 'Employee';
EXEC InsertEmployeeInfo 12, 'Jack', 'Lewis', 'Software Developer', '2024-01-20', 'jack.lewis', 'Employee';

-- HR
EXEC InsertEmployeeInfo 13, 'Kate', 'Martin', 'HR Lead', '2023-05-09', 'kate.martin', 'HR';
EXEC InsertEmployeeInfo 14, 'Liam', 'Walker', 'HR', '2023-07-22', 'liam.walker', 'HR';

-- SUPPORT
EXEC InsertEmployeeInfo 15, 'Mia', 'Harris', 'Support Team Lead', '2023-06-13', 'mia.harris', 'Manager';
EXEC InsertEmployeeInfo 16, 'Noah', 'Clarkson', 'Support Team', '2023-08-25', 'noah.clarkson', 'Employee';
GO

go
CREATE PROCEDURE InsertPerformanceReview
    @EmployeeID INT,
    @ReviewDate DATE,
    @Score INT,
    @Comments VARCHAR(255)
AS
BEGIN
    INSERT INTO Employee.PerformanceReview (employee_id, review_date, score, comments)
    VALUES (@EmployeeID, @ReviewDate, @Score, @Comments);
END;
go

-- TEAM A
-- TEAM A
-- John Doe (steady decline)
EXEC InsertPerformanceReview 1, '2023-01-10', 90, 'Strong start to the year, excellent leadership';
EXEC InsertPerformanceReview 1, '2023-03-10', 88, 'Maintaining performance, good team collaboration';
EXEC InsertPerformanceReview 1, '2023-05-10', 85, 'Slight decline in output, needs to focus on delegation';
EXEC InsertPerformanceReview 1, '2023-07-10', 82, 'Decreased productivity, improvement needed in time management';
EXEC InsertPerformanceReview 1, '2023-09-10', 78, 'Needs improvement, focus on team engagement';
EXEC InsertPerformanceReview 1, '2023-11-10', 75, 'Further decline, address performance issues';
EXEC InsertPerformanceReview 1, '2024-01-10', 70, 'Needs significant improvement, focus on strategic planning';
EXEC InsertPerformanceReview 1, '2024-03-10', 68, 'Performance issues, requires attention to detail';

-- Jane Smith (gradual improvement)
EXEC InsertPerformanceReview 2, '2023-01-15', 70, 'Needs improvement, work on technical skills';
EXEC InsertPerformanceReview 2, '2023-03-15', 72, 'Slight improvement, good progress in coding practices';
EXEC InsertPerformanceReview 2, '2023-05-15', 75, 'Continuing to improve, better communication with team';
EXEC InsertPerformanceReview 2, '2023-07-15', 78, 'Good progress, strong technical abilities';
EXEC InsertPerformanceReview 2, '2023-09-15', 80, 'Solid performance, shows potential in problem-solving';
EXEC InsertPerformanceReview 2, '2023-11-15', 82, 'Consistently improving, great at handling tasks independently';
EXEC InsertPerformanceReview 2, '2024-01-15', 85, 'Strong performance, excellent team player';
EXEC InsertPerformanceReview 2, '2024-03-15', 88, 'Exceeds expectations, innovative solutions';

-- Alice Johnson (steady performance)
EXEC InsertPerformanceReview 3, '2023-01-20', 78, 'Consistent performance, reliable in project delivery';
EXEC InsertPerformanceReview 3, '2023-03-20', 80, 'Maintaining standards, good project management';
EXEC InsertPerformanceReview 3, '2023-05-20', 82, 'Good work, excellent in meeting deadlines';
EXEC InsertPerformanceReview 3, '2023-07-20', 81, 'Stable performance, shows leadership potential';
EXEC InsertPerformanceReview 3, '2023-09-20', 83, 'Slight improvement, effective communicator';
EXEC InsertPerformanceReview 3, '2023-11-20', 82, 'Maintaining consistency, reliable team member';
EXEC InsertPerformanceReview 3, '2024-01-20', 84, 'Solid performance, good technical knowledge';
EXEC InsertPerformanceReview 3, '2024-03-20', 85, 'Good progress, proactive problem-solving';

-- Bob Williams (gradual improvement)
EXEC InsertPerformanceReview 4, '2023-01-25', 65, 'Needs improvement, focus on code quality';
EXEC InsertPerformanceReview 4, '2023-03-25', 68, 'Slight improvement, better communication skills';
EXEC InsertPerformanceReview 4, '2023-05-25', 70, 'Improving performance, showing potential';
EXEC InsertPerformanceReview 4, '2023-07-25', 73, 'Continued improvement, strong technical abilities';
EXEC InsertPerformanceReview 4, '2023-09-25', 75, 'Good progress, excellent problem-solving skills';
EXEC InsertPerformanceReview 4, '2023-11-25', 78, 'Solid performance, reliable team member';
EXEC InsertPerformanceReview 4, '2024-01-25', 80, 'Strong performance, great collaboration with peers';
EXEC InsertPerformanceReview 4, '2024-03-25', 82, 'Exceeds expectations, innovative thinker';

```



```

-- David Jones (gradual improvement)
EXEC InsertPerformanceReview 6, '2023-03-01', 55, 'Poor start, needs to focus on basics';
EXEC InsertPerformanceReview 6, '2023-05-01', 60, 'Slight improvement, better understanding of tasks';
EXEC InsertPerformanceReview 6, '2023-07-01', 65, 'Continued improvement, showing potential';
EXEC InsertPerformanceReview 6, '2023-09-01', 68, 'Better performance, good technical skills';
EXEC InsertPerformanceReview 6, '2023-11-01', 70, 'Good progress, reliable team member';
EXEC InsertPerformanceReview 6, '2024-01-01', 73, 'Solid performance, excellent collaboration';
EXEC InsertPerformanceReview 6, '2024-03-01', 75, 'Strong improvement, great problem-solving skills';
EXEC InsertPerformanceReview 6, '2024-05-01', 78, 'Exceeds expectations, innovative thinker';

-- Eva Miller (steady performance)
EXEC InsertPerformanceReview 7, '2023-05-05', 90, 'Excellent start, outstanding work quality';
EXEC InsertPerformanceReview 7, '2023-07-05', 92, 'Continued excellence, great problem-solving skills';
EXEC InsertPerformanceReview 7, '2023-09-05', 93, 'Outstanding work, excellent in meeting deadlines';
EXEC InsertPerformanceReview 7, '2023-11-05', 94, 'Maintaining high standards, reliable team member';
EXEC InsertPerformanceReview 7, '2024-01-05', 92, 'Slight decline, still excellent performance';
EXEC InsertPerformanceReview 7, '2024-03-05', 93, 'Excellent performance, strong technical skills';
EXEC InsertPerformanceReview 7, '2024-05-05', 94, 'Outstanding work, great team collaboration';
EXEC InsertPerformanceReview 7, '2024-07-05', 95, 'Exceeds expectations, innovative solutions';

-- Frank Davis (steady performance)
EXEC InsertPerformanceReview 8, '2023-09-01', 82, 'Good start, strong technical skills';
EXEC InsertPerformanceReview 8, '2023-11-01', 83, 'Maintaining performance, reliable team member';
EXEC InsertPerformanceReview 8, '2024-01-01', 84, 'Consistent performance, good problem-solving skills';
EXEC InsertPerformanceReview 8, '2024-03-01', 85, 'Solid performance, excellent collaboration';
EXEC InsertPerformanceReview 8, '2024-05-01', 86, 'Great progress, reliable and dependable';
EXEC InsertPerformanceReview 8, '2024-07-01', 87, 'Maintaining high standards, strong technical knowledge';
EXEC InsertPerformanceReview 8, '2024-09-01', 88, 'Good work, excellent in meeting deadlines';
EXEC InsertPerformanceReview 8, '2024-11-01', 89, 'Outstanding performance, innovative thinker';

-- TEAM B
-- Grace Hall (steady performance)
EXEC InsertPerformanceReview 9, '2023-10-10', 90, 'Strong leadership, excellent team management';
EXEC InsertPerformanceReview 9, '2023-12-10', 92, 'Great strategic planning, effective collaboration';
EXEC InsertPerformanceReview 9, '2024-02-10', 94, 'Outstanding management, reliable and dependable';
EXEC InsertPerformanceReview 9, '2024-04-10', 93, 'Consistent performance, excellent problem-solving skills';
EXEC InsertPerformanceReview 9, '2024-06-10', 92, 'Maintaining standards, good team engagement';
EXEC InsertPerformanceReview 9, '2024-08-10', 91, 'Slight decline, focus on strategic goals';
EXEC InsertPerformanceReview 9, '2024-10-10', 90, 'Strong performance, effective leadership';
EXEC InsertPerformanceReview 9, '2024-12-10', 89, 'Maintaining high standards, needs focus on delegation';

-- Hannah Moore (gradual improvement)
EXEC InsertPerformanceReview 10, '2023-09-15', 65, 'Needs improvement, focus on technical skills';
EXEC InsertPerformanceReview 10, '2023-11-15', 70, 'Slight improvement, better understanding of tasks';
EXEC InsertPerformanceReview 10, '2024-01-15', 75, 'Continued improvement, strong coding practices';
EXEC InsertPerformanceReview 10, '2024-03-15', 78, 'Good progress, reliable team member';
EXEC InsertPerformanceReview 10, '2024-05-15', 80, 'Solid performance, good problem-solving skills';
EXEC InsertPerformanceReview 10, '2024-07-15', 82, 'Strong improvement, effective communicator';
EXEC InsertPerformanceReview 10, '2024-09-15', 85, 'Excellent performance, innovative thinker';
EXEC InsertPerformanceReview 10, '2024-11-15', 88, 'Exceeds expectations, reliable and dependable';

-- Isaac Clark (steady performance)
EXEC InsertPerformanceReview 11, '2024-01-20', 78, 'Good start, strong technical skills';
EXEC InsertPerformanceReview 11, '2024-03-20', 80, 'Maintaining standards, reliable team member';
EXEC InsertPerformanceReview 11, '2024-05-20', 82, 'Good work, excellent problem-solving skills';
EXEC InsertPerformanceReview 11, '2024-07-20', 81, 'Stable performance, effective communicator';
EXEC InsertPerformanceReview 11, '2024-09-20', 83, 'Slight improvement, reliable and dependable';
EXEC InsertPerformanceReview 11, '2024-11-20', 82, 'Consistent performance, strong technical abilities';
EXEC InsertPerformanceReview 11, '2025-01-20', 84, 'Solid performance, great collaboration';
EXEC InsertPerformanceReview 11, '2025-03-20', 85, 'Good progress, innovative solutions';

-- Jack Lewis (gradual improvement)
EXEC InsertPerformanceReview 12, '2024-01-25', 60, 'Needs improvement, focus on basics';
EXEC InsertPerformanceReview 12, '2024-03-25', 65, 'Slight improvement, better understanding of tasks';
EXEC InsertPerformanceReview 12, '2024-05-25', 70, 'Continued improvement, strong technical skills';
EXEC InsertPerformanceReview 12, '2024-07-25', 73, 'Good progress, reliable team member';
EXEC InsertPerformanceReview 12, '2024-09-25', 75, 'Solid performance, good problem-solving skills';
EXEC InsertPerformanceReview 12, '2024-11-25', 78, 'Strong improvement, effective communicator';
EXEC InsertPerformanceReview 12, '2025-01-25', 80, 'Excellent performance, innovative thinker';
EXEC InsertPerformanceReview 12, '2025-03-25', 82, 'Exceeds expectations, reliable and dependable';

-- HR
-- Kate Martin (steady performance)
EXEC InsertPerformanceReview 13, '2023-07-25', 88, 'Strong HR management, great employee relations';
EXEC InsertPerformanceReview 13, '2023-09-25', 89, 'Maintaining standards, effective in conflict resolution';
EXEC InsertPerformanceReview 13, '2023-11-25', 90, 'Excellent performance, strong communication skills';
EXEC InsertPerformanceReview 13, '2024-01-25', 91, 'Consistent performance, good team collaboration';
EXEC InsertPerformanceReview 13, '2024-03-25', 92, 'Outstanding work, reliable and dependable';
EXEC InsertPerformanceReview 13, '2024-05-25', 91, 'Maintaining high standards, effective problem-solving';
EXEC InsertPerformanceReview 13, '2024-07-25', 90, 'Strong performance, excellent employee engagement';
EXEC InsertPerformanceReview 13, '2024-09-25', 89, 'Consistent performance, reliable and dependable';

```

```

-- Liam Walker (steady decline)
EXEC InsertPerformanceReview 14, '2023-08-20', 78, 'Dependable performance, good HR support';
EXEC InsertPerformanceReview 14, '2023-10-20', 76, 'Slight decline, needs to focus on communication';
EXEC InsertPerformanceReview 14, '2023-12-20', 74, 'Continued decline, work on problem-solving';
EXEC InsertPerformanceReview 14, '2024-02-20', 72, 'Needs improvement, focus on team collaboration';
EXEC InsertPerformanceReview 14, '2024-04-20', 70, 'Performance issues, address team engagement';
EXEC InsertPerformanceReview 14, '2024-06-20', 68, 'Further decline, requires attention to detail';
EXEC InsertPerformanceReview 14, '2024-08-20', 66, 'Needs significant improvement, focus on task completion';
EXEC InsertPerformanceReview 14, '2024-10-20', 65, 'Performance concerns, work on time management';

)-- SUPPORT
-- Mia Harris (steady performance)
EXEC InsertPerformanceReview 15, '2023-09-15', 87, 'Great customer service, strong problem resolution';
EXEC InsertPerformanceReview 15, '2023-11-15', 88, 'Maintaining standards, effective support skills';
EXEC InsertPerformanceReview 15, '2024-01-15', 89, 'Excellent performance, reliable and dependable';
EXEC InsertPerformanceReview 15, '2024-03-15', 90, 'Consistent performance, good team collaboration';
EXEC InsertPerformanceReview 15, '2024-05-15', 91, 'Outstanding work, effective in resolving issues';
EXEC InsertPerformanceReview 15, '2024-07-15', 90, 'Maintaining high standards, great customer interaction';
EXEC InsertPerformanceReview 15, '2024-09-15', 89, 'Strong performance, reliable support';
EXEC InsertPerformanceReview 15, '2024-11-15', 88, 'Consistent performance, good technical skills';

-- Noah Clarkson (gradual improvement)
EXEC InsertPerformanceReview 16, '2023-08-10', 65, 'Needs improvement, focus on technical support';
EXEC InsertPerformanceReview 16, '2023-10-10', 68, 'Slight improvement, better problem-solving skills';
EXEC InsertPerformanceReview 16, '2023-12-10', 70, 'Continued improvement, reliable team member';
EXEC InsertPerformanceReview 16, '2024-02-10', 73, 'Good progress, effective communicator';
EXEC InsertPerformanceReview 16, '2024-04-10', 75, 'Solid performance, good troubleshooting skills';
EXEC InsertPerformanceReview 16, '2024-06-10', 78, 'Strong improvement, great customer service';
EXEC InsertPerformanceReview 16, '2024-08-10', 80, 'Excellent performance, innovative thinker';
EXEC InsertPerformanceReview 16, '2024-10-10', 82, 'Exceeds expectations, reliable and dependable';

go

SET IDENTITY_INSERT Department.DepartmentInfo ON;
go

)CREATE PROCEDURE InsertDepartmentInfo
    @DepartmentID INT,
    @DepartmentName VARCHAR(255)
AS
)BEGIN
)    INSERT INTO Department.DepartmentInfo (department_id, department_name)
)        VALUES (@DepartmentID, @DepartmentName);
)END;
go

EXEC InsertDepartmentInfo 1, 'HR';
EXEC InsertDepartmentInfo 2, 'Main Team A';
EXEC InsertDepartmentInfo 3, 'Secondary Team B';
EXEC InsertDepartmentInfo 4, 'Support Team';
go

SET IDENTITY_INSERT Department.DepartmentInfo OFF;
go

)CREATE PROCEDURE InsertEmployeeDepartmentInfo
    @EmployeeID INT,
    @DepartmentID INT
AS
)BEGIN
)    INSERT INTO Department.EmployeeDepartment (employee_id, department_id)
)        VALUES (@EmployeeID, @DepartmentID);
)END;
go

-- Team A in Software Development
EXEC InsertEmployeeDepartmentInfo 1, 2;
EXEC InsertEmployeeDepartmentInfo 2, 2;
EXEC InsertEmployeeDepartmentInfo 3, 2;
EXEC InsertEmployeeDepartmentInfo 4, 2;
EXEC InsertEmployeeDepartmentInfo 5, 2;
EXEC InsertEmployeeDepartmentInfo 6, 2;
EXEC InsertEmployeeDepartmentInfo 7, 2;
EXEC InsertEmployeeDepartmentInfo 8, 2;

-- Team B in Software Development
EXEC InsertEmployeeDepartmentInfo 9, 3;
EXEC InsertEmployeeDepartmentInfo 10, 3;
EXEC InsertEmployeeDepartmentInfo 11, 3;
EXEC InsertEmployeeDepartmentInfo 12, 3;

-- HR Department
EXEC InsertEmployeeDepartmentInfo 13, 1;
EXEC InsertEmployeeDepartmentInfo 14, 1;

-- Support Team
EXEC InsertEmployeeDepartmentInfo 15, 4;
EXEC InsertEmployeeDepartmentInfo 16, 4;
go

```

## Stored Procedures SQL

### Insert Employee Info

```
CREATE PROCEDURE InsertEmployeeInfo
    @EmployeeID INT,
    @FirstName VARCHAR(255),
    @LastName VARCHAR(255),
    @Position VARCHAR(255),
    @HireDate DATE,
    @UserName VARCHAR(255),
    @Role VARCHAR(255)
AS
BEGIN
    INSERT INTO Employee.EmployeeInfo (employee_id, first_name, last_name, position, role_position, hire_date)
    VALUES (@EmployeeID, @FirstName, @LastName, @Position, @Role, @HireDate);

    -- Insert user into AspNetUsers
    -- Default Password (YourSecurePassword123!)
    INSERT INTO AspNetUsers (Id, UserName, NormalizedUserName, Email, NormalizedEmail, EmailConfirmed, PasswordHash,
    SecurityStamp, ConcurrencyStamp, PhoneNumberConfirmed, TwoFactorEnabled, LockoutEnabled, AccessFailedCount)
    VALUES (@EmployeeID, @UserName, UPPER(@UserName), @UserName + '@example.com', UPPER(@UserName + '@example.com'), 1,
    'AQAAAAIAAYagAAAAEC9LifLJYADjCODAqX2wGnmCnA1cACbMkhsEArIPy4ZXvRVG5fZCieOeU51wqjTohg==',
    NEWID(), NEWID(), 0, 0, 0, 0);

    -- Insert user role into AspNetUserRoles
    DECLARE @UserId NVARCHAR(450) = (SELECT Id FROM AspNetUsers WHERE UserName = @UserName);
    DECLARE @RoleId NVARCHAR(450) = (SELECT Id FROM AspNetRoles WHERE Name = @Role);

    IF @RoleId IS NULL
    BEGIN
        -- Insert role into AspNetRoles if it doesn't exist
        SET @RoleId = NEWID();
        INSERT INTO AspNetRoles (Id, Name, NormalizedName, ConcurrencyStamp)
        VALUES (@RoleId, @Role, UPPER(@Role), NEWID());
    END

    INSERT INTO AspNetUserRoles (UserId, RoleId)
    VALUES (@UserId, @RoleId);
END;
```

### Insert Performance Review

```
CREATE PROCEDURE InsertPerformanceReview
    @EmployeeID INT,
    @ReviewDate DATE,
    @Score INT,
    @Comments VARCHAR(255)
AS
BEGIN
    INSERT INTO Employee.PerformanceReview (employee_id, review_date, score, comments)
    VALUES (@EmployeeID, @ReviewDate, @Score, @Comments);
END;
go
```

## Insert Department Info

```
CREATE PROCEDURE InsertDepartmentInfo
    @DepartmentID INT,
    @DepartmentName VARCHAR(255)
AS
BEGIN
    INSERT INTO Department.DepartmentInfo (department_id, department_name)
    VALUES (@DepartmentID, @DepartmentName);
END;
go
```

## Insert Employee Department Info

```
CREATE PROCEDURE InsertEmployeeDepartmentInfo
    @EmployeeID INT,
    @DepartmentID INT
AS
BEGIN
    INSERT INTO Department.EmployeeDepartment (employee_id, department_id)
    VALUES (@EmployeeID, @DepartmentID);
END;
go
```

## Add New Employee

When adding a new employee I need to make it add entries to both the EmployeeInfo and the EmployeeDepartment. To create an new employee entry and connect it to the department.

```

CREATE PROCEDURE AddNewEmployee
    @FirstName VARCHAR(255),
    @LastName VARCHAR(255),
    @Position VARCHAR(255),
    @HireDate DATE,
    @DepartmentID INT,
    @LoginName VARCHAR(255),
    @RoleName VARCHAR(255),
    @RolePosition VARCHAR(255)
AS
BEGIN
    SET NOCOUNT ON;

    -- Insert the new employee and return the new employee ID
    INSERT INTO Employee.EmployeeInfo (first_name, last_name, position, hire_date, login_name, role_position)
    OUTPUT INSERTED.employee_id
    VALUES (@FirstName, @LastName, @Position, @HireDate, @LoginName, @RolePosition);

    -- Store the new employee ID
    DECLARE @NewEmployeeID INT;
    SET @NewEmployeeID = SCOPE_IDENTITY();

    -- Insert into the EmployeeDepartment table
    INSERT INTO Department.EmployeeDepartment (employee_id, department_id)
    VALUES (@NewEmployeeID, @DepartmentID);

    -- Add the login to the role
    EXEC sp_addrolemember @RoleName, @LoginName;

    -- Return the new employee ID
    SELECT @NewEmployeeID AS NewEmployeeID;
END;

```

## Update Employee Details

```
CREATE PROCEDURE UpdateEmployeeDetails
    @EmployeeID INT,
    @FirstName VARCHAR(255),
    @LastName VARCHAR(255),
    @Position VARCHAR(255),
    @HireDate DATE,
    @DepartmentID INT
AS
BEGIN
    SET NOCOUNT ON;

    BEGIN TRY
        -- Update employee details in Employee.EmployeeInfo table
        UPDATE Employee.EmployeeInfo
        SET first_name = @FirstName,
            last_name = @LastName,
            position = @Position,
            hire_date = @HireDate
        WHERE employee_id = @EmployeeID;

        -- Update department ID in Department.EmployeeDepartment table
        UPDATE Department.EmployeeDepartment
        SET department_id = @DepartmentID
        WHERE employee_id = @EmployeeID;

        -- Check if any rows were affected
        IF @@ROWCOUNT = 0
        BEGIN
            RAISERROR ('Employee not found or details unchanged.', 16, 1);
        END
    END TRY
    BEGIN CATCH
        DECLARE @ErrorMessage NVARCHAR(4000) = ERROR_MESSAGE();
        DECLARE @ErrorSeverity INT = ERROR_SEVERITY();
        DECLARE @ErrorState INT = ERROR_STATE();

        -- Rollback the transaction if one is active
        IF @@TRANCOUNT > 0
            ROLLBACK;

        -- Raise the error
        RAISERROR (@ErrorMessage, @ErrorSeverity, @ErrorState);
    END CATCH;
END;
```

## Get Employee Performance Reviews

```
CREATE PROCEDURE GetEmployeePerformanceReview
    @EmployeeID INT
AS
BEGIN
    SET NOCOUNT ON;

    SELECT * from Employee.PerformanceReview
    WHERE employee_id = @EmployeeID;
END;
```

## Get Top 5 Employees

```
CREATE PROCEDURE GetTop5EmployeesAverageScore
AS
BEGIN
    SET NOCOUNT ON;

    SELECT TOP 5
        employee.employee_id,
        employee.first_name,
        employee.last_name,
        AVG(pr.score) AS average_score
    FROM
        Employee.EmployeeInfo employee
    INNER JOIN
        Employee.PerformanceReview pr ON employee.employee_id = pr.employee_id
    GROUP BY
        employee.employee_id, employee.first_name, employee.last_name
    ORDER BY
        average_score DESC;
END;
```

## Get Employees That Need a Performance Review

```
CREATE PROCEDURE GetEmployeeDueForReview
    @MonthCutOff INT
AS
BEGIN
    SET NOCOUNT ON;

    DECLARE @CutOff DATE;
    SET @CutOff = DATEADD(MONTH, -@MonthCutOff, GETDATE());

    SELECT
        employee.employee_id,
        employee.first_name,
        employee.last_name,
        employee.position,
        employee.hire_date,
        MAX(pr.review_date) AS last_review_date
    FROM
        Employee.EmployeeInfo employee
    LEFT JOIN
        Employee.PerformanceReview pr
    ON
        employee.employee_id = pr.employee_id
    GROUP BY
        employee.employee_id, employee.first_name, employee.last_name, employee.position, employee.hire_date
    HAVING
        MAX(pr.review_date) IS NULL OR MAX(pr.review_date) <= @CutOff
    ORDER BY
        last_review_date ASC;
END;
```



## Triggers SQL

### Trigger After Insert Performance Review

```
CREATE TRIGGER Trigger_AfterInsert_PerformanceReview
ON Employee.PerformanceReview
AFTER INSERT
AS
BEGIN
    INSERT INTO Employee.PerformanceReviewAudit (review_id, employee_id, review_date, score, comments, operation_type)
    SELECT review_id, employee_id, review_date, score, comments, 'INSERT'
    FROM inserted;
END;
```

### Trigger After Update Performance Review

```
CREATE TRIGGER Trigger_AfterUpdate_PerformanceReview
ON Employee.PerformanceReview
AFTER UPDATE
AS
BEGIN
    INSERT INTO Employee.PerformanceReviewAudit (review_id, employee_id, review_date, score, comments, operation_type)
    SELECT i.review_id, i.employee_id, i.review_date, i.score, i.comments, 'UPDATE'
    FROM inserted i
    INNER JOIN deleted d ON i.review_id = d.review_id;
END;
```