```sql
CREATE DATABASE PracticeDB;

USE PracticeDB;
GO

-- I. SQL Server Setup & Basic Queries

CREATE TABLE Employees (
    id INT NOT NULL PRIMARY KEY IDENTITY(1,1),
    name NVARCHAR(255) NOT NULL,
    dob DATE NOT NULL,
    salary DECIMAL(10,2) NOT NULL,
    position VARCHAR(100) NOT NULL
);

INSERT INTO Employees (name, dob, salary, position)
VALUES
    ('John Doe', '1985-06-15', 55000.00, 'Software Engineer'),
    ('Jane Smith', '1990-09-22', 62000.00, 'Data Analyst'),
    ('Michael Brown', '1982-12-05', 75000.00, 'Project Manager'),
    ('Emily Davis', '1995-03-18', 50000.00, 'HR Specialist'),
    ('David Wilson', '1988-07-10', 68000.00, 'Network Administrator');

SELECT * FROM Employees;

-- 2.   Using Variables and Expressions
-- DECLARE Variable

DECLARE @name1 NVARCHAR(255), @dob1 DATE, @salary1 FLOAT, @position1 NVARCHAR(255);
DECLARE @name2 NVARCHAR(255), @dob2 DATE, @salary2 FLOAT, @position2 NVARCHAR(255);

SET @name1 = 'Alice Johnson';
SET @dob1 = '1992-04-25';
SET @salary1 = 58000.00;
SET @position1 = 'Marketing Specialist';

SET @name2 = 'Robert Miller';
SET @dob2 = '1987-11-15';
SET @salary2 = 72000.00;
SET @position2 = 'IT Manager';

-- Insert records using variables
INSERT INTO Employees (name, dob, salary, position)
VALUES (@name1, @dob1, @salary1, @position1),
       (@name2, @dob2, @salary2, @position2);

-- 2.b Retrieve data with condition base on variable
DECLARE @SearchPosition VARCHAR(100);
SET @SearchPosition = 'Marketing Specialist';

SELECT * FROM Employees
WHERE position = @SearchPosition;
```

```sql
-- 2.c Update and delete records base on your variable
-- Update
DECLARE @UpdateName VARCHAR(255), @NewSalary DECIMAL(10,2);
SET @UpdateName = 'Alice Johnson';
SET @NewSalary = 60000.00;

UPDATE Employees
SET salary = @NewSalary
WHERE name = @UpdateName;

-- Delete
DECLARE @DeleteName VARCHAR(255);
SET @DeleteName = 'Robert Miller';

DELETE FROM Employees
WHERE name = @DeleteName;

-- 3. Conditional Statements (IF, CASE)

DECLARE @EmpSalary FLOAT;

-- Set the salary to test
SET @EmpSalary = 55000;

IF @EmpSalary > 50000
    PRINT 'High Salary';
ELSE
    PRINT 'Low Salary';

SELECT
    name,
    dob,
    salary,
    position,

    -- Classify Salary using CASE
    CASE
        WHEN salary > 50000 THEN 'High Salary'
        ELSE 'Low Salary'
    END AS Salary_Category,

    -- Classify Employee as Junior or Senior based on DOB
    CASE
        WHEN YEAR(dob) < 2000 THEN 'Junior'
        ELSE 'Senior'
    END AS Employee_Category

FROM Employees;

--3. WHILE Loop with Condition
```

```sql
DECLARE @Counter INT = 1;

WHILE @Counter <= 10
BEGIN
    PRINT 'Iteration: ' + CAST(@Counter AS NVARCHAR);

    -- Exit loop when counter reaches 5
    IF @Counter = 5
        BREAK;

    SET @Counter = @Counter + 1;
END;

--II. Break, continue, and return
--1.  Display data 2, 4, 6, … 100
--Exit the loop when the counter reaches 50

DECLARE @Counter INT = 2;

WHILE @Counter <= 100
BEGIN
    -- Print the current value
    PRINT CAST(@Counter AS NVARCHAR);

    -- Exit the loop when the counter reaches 50
    IF @Counter = 50
        BREAK;

    -- Increment by 2 (Even numbers only)
    SET @Counter = @Counter + 2;
END;

--2.Display value from 1 to 100
--Skip even numbers and print only odd numbers

DECLARE @Counter INT = 1;




WHILE @Counter <= 100
BEGIN
    -- Skip even numbers using CONTINUE
    IF @Counter % 2 = 0
    BEGIN
        SET @Counter = @Counter + 1;
        CONTINUE; -- Skip printing
    END;

    -- Print odd numbers
    PRINT CAST(@Counter AS NVARCHAR);
    -- Increment by 1
    SET @Counter = @Counter + 1;
END;

--III. Create and run simple stored procedure
--1. Create a Stored Procedure to Select Data from the Employees Table
CREATE PROCEDURE GetAllEmployees
AS
BEGIN
    SELECT * FROM Employees;
END;

EXEC GetAllEmployees;

--2. Create a Stored Procedure to Insert Data with Parameters
CREATE PROCEDURE InsertEmployee
    @Name NVARCHAR(100),
    @Dob DATE,
    @Salary FLOAT,
    @Position NVARCHAR(100)
AS
BEGIN
    INSERT INTO Employees (name, dob, salary, position)
    VALUES (@Name, @Dob, @Salary, @Position);
END;

EXEC InsertEmployee 'CHHORN Vannak', '1995-04-30', 750, 'System Engineer';

--3. Select Salary and Handle NULL Values
CREATE PROCEDURE CheckEmployeeSalaries
AS
BEGIN
    DECLARE @Salary FLOAT, @EmployeeID INT;

    DECLARE SalaryCursor CURSOR FOR
    SELECT id, salary FROM Employees;

    OPEN SalaryCursor;

    FETCH NEXT FROM SalaryCursor INTO @EmployeeID, @Salary;

    WHILE @@FETCH_STATUS = 0
    BEGIN
        -- If salary is NULL, skip to next record
        IF @Salary IS NULL
        BEGIN
            FETCH NEXT FROM SalaryCursor INTO @EmployeeID, @Salary;
            CONTINUE;
        END;

        -- Print salary
        PRINT 'Employee ID: ' + CAST(@EmployeeID AS NVARCHAR) +
                ' - Salary: ' + CAST(@Salary AS NVARCHAR);

        FETCH NEXT FROM SalaryCursor INTO @EmployeeID, @Salary;
    END;

    CLOSE SalaryCursor;
    DEALLOCATE SalaryCursor;
END;

EXEC CheckEmployeeSalaries;
```