P5 - PSM IMPLEMENTATION
Topic : Sustainable Tourism Management System

Submitted By: Group 11

-- Additions to the Tourism Management System (SQL Server)

```
USE STMS
GO
```

-- 1. Stored Procedures

-- Insert new tourist

```
CREATE PROCEDURE AddTourist
    @FirstName NVARCHAR(50),
    @LastName NVARCHAR(50),
    @Nationality NVARCHAR(50),
    @DateOfBirth DATE,
    @Email NVARCHAR(100),
    @PreferredLanguage NVARCHAR(50),
    @SustainabilityPreference NVARCHAR(100),
    @CarbonOffsetParticipation BIT
AS
BEGIN
    INSERT INTO Tourist (FirstName, LastName, Nationality, DateOfBirth, Email,
PreferredLanguage, SustainabilityPreference, CarbonOffsetParticipation)
    VALUES (@FirstName, @LastName, @Nationality, @DateOfBirth, @Email,
@PreferredLanguage, @SustainabilityPreference, @CarbonOffsetParticipation);
END;

EXEC AddTourist
    @FirstName = 'Chris',
    @LastName = 'Paul',
    @Nationality = 'American',
    @DateOfBirth = '1990-05-15',
    @Email = 'chris.paul@example.com',
    @PreferredLanguage = 'English',
    @SustainabilityPreference = 'Eco-friendly accommodations',
    @CarbonOffsetParticipation = 1; -- 1 for TRUE, 0 for FALSE
```

```sql
SELECT *
FROM Tourist
WHERE Email = 'chris.paul@example.com';
```

-- Update accommodation sustainability rating

```sql
CREATE PROCEDURE UpdateAccommodationRating
   @AccommodationID INT,
   @NewRating DECIMAL(5, 2)
AS
BEGIN
   UPDATE Accommodation
   SET SustainabilityRating = @NewRating
   WHERE AccommodationID = @AccommodationID;
END;

EXEC UpdateAccommodationRating
   @AccommodationID = 6, -- Replace with the actual AccommodationID
   @NewRating = 87.50; -- Replace with the desired new sustainability rating

SELECT * FROM Accommodation WHERE AccommodationID = 5;
```

-- Fetch destinations by sustainability score

```sql
CREATE PROCEDURE GetDestinationsBySustainability
   @MinScore DECIMAL(5, 2)
AS
BEGIN
   SELECT *
   FROM Destination
   WHERE SustainabilityScore >= @MinScore;
END;
EXEC GetDestinationsBySustainability
   @MinScore = 80.00; -- Replace with the desired minimum sustainability score

SELECT *
FROM Destination
WHERE SustainabilityScore >= 80.00;
```

-- Procedure to retrieve attractions by type

```sql
CREATE PROCEDURE GetAttractionsByType
```

```
    @AttractionType NVARCHAR(50)
AS
BEGIN
    SELECT AttractionName, DestinationID, Capacity, AccessibilityFeatures
    FROM Attraction
    WHERE AttType = @AttractionType;
END;

EXEC GetAttractionsByType @AttractionType = 'Historical Site';

SELECT AttractionName, DestinationID, Capacity, AccessibilityFeatures
FROM Attraction
WHERE AttType = 'Historical Site';
```

-- Procedure to add a new tour operator

```
CREATE PROCEDURE AddTourOperator
    @OperatorName NVARCHAR(100),
    @Headquarters NVARCHAR(100),
    @SustainabilityCertification NVARCHAR(100),
    @LocalPartnershipPercentage DECIMAL(5, 2),
    @CarbonOffsetProgram NVARCHAR(100),
    @SustainablePackagesOffered INT
AS
BEGIN
    -- Insert a new record into the Tour_Operator table
    INSERT INTO Tour_Operator (
        OperatorName,
        Headquarters,
        SustainabilityCertification,
        LocalPartnershipPercentage,
        CarbonOffsetProgram,
        SustainablePackagesOffered
    )
    VALUES (
        @OperatorName,
        @Headquarters,
        @SustainabilityCertification,
        @LocalPartnershipPercentage,
        @CarbonOffsetProgram,
        @SustainablePackagesOffered
    );
END;
```

```
EXEC AddTourOperator
    @OperatorName = 'Eco Adventures',
    @Headquarters = 'Sydney, Australia',
    @SustainabilityCertification = 'EarthCheck',
    @LocalPartnershipPercentage = 80.5,
    @CarbonOffsetProgram = 'Carbon Fund',
    @SustainablePackagesOffered = 25;

SELECT *
FROM Tour_Operator
WHERE CarbonOffsetProgram = 'Carbon Fund';
```

-- 2. Views

-- High sustainability destinations

This view analyzes `HighSustainabilityDestinations`, to list destinations with a sustainability score greater than 80, along with their country, region, and score.

```
CREATE VIEW HighSustainabilityDestinations AS
SELECT DestinationName, Country, Region, SustainabilityScore
FROM Destination
WHERE SustainabilityScore > 80;

SELECT *
FROM HighSustainabilityDestinations;
```

– Visitor Capacity Analysis

This view gives `VisitorCapacityAnalysis`, to evaluate whether destinations are within or exceeding their visitor capacity limits, providing a capacity status for each destination.

```
CREATE VIEW v_VisitorCapacityAnalysis AS
SELECT
    DestinationName,
    Country,
    AverageAnnualVisitors,
    CarryingCapacity,
    CASE
        WHEN AverageAnnualVisitors <= CarryingCapacity THEN 'Within Capacity'
        ELSE 'Over Capacity'
    END AS CapacityStatus
FROM Destination;

SELECT *
FROM v_VisitorCapacityAnalysis
ORDER BY CapacityStatus, DestinationName;
```

-- Revenue analytics

This view analyzes total tourism revenue for each destination by aggregating data from the `Destination` and `Economic_Impact` tables, ordered by revenue in descending order.

```
CREATE VIEW RevenueAnalytics AS
SELECT DestinationName, SUM(TourismRevenue) AS TotalRevenue
FROM Destination
JOIN Economic_Impact ON Destination.DestinationID = Economic_Impact.DestinationID
GROUP BY DestinationName;

SELECT *
FROM RevenueAnalytics
ORDER BY TotalRevenue DESC;
```

-- Accommodation Efficiency Analysis

This view evaluates accommodations based on their energy, water, and waste management scores, providing insights for operational improvements.

```
CREATE VIEW v_AccommodationEfficiencyAnalysis AS
SELECT
    AccommodationName,
    DestinationName,
    SustainabilityRating,
    EnergyEfficiencyScore,
    WaterConservationScore,
    WasteManagementScore,
    CAST(ROUND((EnergyEfficiencyScore + WaterConservationScore +
WasteManagementScore) / 3.0, 2) AS DECIMAL(10, 2)) AS AverageEfficiencyScore
FROM
```

```
    Accommodation A
JOIN
    Destination D ON A.DestinationID = D.DestinationID
WHERE
    SustainabilityRating IS NOT NULL;


SELECT *
FROM v_AccommodationEfficiencyAnalysis
WHERE AverageEfficiencyScore > 80;
```

-- Regional Visitor Stats

This view provides a summary of visitor statistics by region and climate type, helping identify tourism trends and patterns.

```
CREATE VIEW v_RegionalVisitorStats AS
SELECT
    Region,
    Climate,
    COUNT(DestinationID) AS TotalDestinations,
    SUM(AverageAnnualVisitors) AS TotalVisitors,
    ROUND(CAST(AVG(CAST(SustainabilityScore AS FLOAT)) AS DECIMAL(10, 2)), 2) AS
AverageSustainabilityScore
FROM
    Destination
GROUP BY
    Region, Climate;


SELECT *
FROM v_RegionalVisitorStats;
```

-- 3. User-Defined Functions

-- Calculate average visitor capacity

```
CREATE FUNCTION AvgVisitorCapacity()
RETURNS DECIMAL(10, 2)
AS
BEGIN
   RETURN (
      SELECT AVG(CarryingCapacity)
      FROM Destination
   );
END;

SELECT dbo.AvgVisitorCapacity() AS AverageCarryingCapacity;
```

-- Calculate sustainability score ratio

```
CREATE FUNCTION SustainabilityScoreRatio(@Score DECIMAL(5, 2))
RETURNS NVARCHAR(50)
AS
BEGIN
   RETURN (
      CASE
         WHEN @Score >= 90 THEN 'Excellent'
         WHEN @Score >= 70 THEN 'Good'
         ELSE 'Average'
      END
   );
END;


SELECT
   DestinationName,
   SustainabilityScore,
   dbo.SustainabilityScoreRatio(SustainabilityScore) AS SustainabilityRating
FROM
   Destination;
```

-- Get total tourists for a destination

```
CREATE FUNCTION TotalTourists(@DestinationID INT)
RETURNS INT
AS
BEGIN
   RETURN (
      SELECT COUNT(*)
      FROM Visits
      WHERE DestinationID = @DestinationID
   );
END;


SELECT
   D.DestinationName,
   dbo.TotalTourists(D.DestinationID) AS TotalTourists
FROM
   Destination D;
```

-- Get Destination Sustainability Category

This scalar function categorizes destinations into sustainability levels (e.g., Low, Medium, High) based on their `SustainabilityScore`.

```
CREATE FUNCTION fn_GetDestinationSustainabilityCategory (@SustainabilityScore
DECIMAL(5, 2))
RETURNS NVARCHAR(50)
AS
BEGIN
   DECLARE @Category NVARCHAR(50);

   IF @SustainabilityScore >= 80
      SET @Category = 'High';
   ELSE
      IF @SustainabilityScore >= 50
         SET @Category = 'Medium';
      ELSE
         SET @Category = 'Low';

   RETURN @Category;
END;
```

```
SELECT
    DestinationName,
    Country,
    dbo.fn_GetDestinationSustainabilityCategory(SustainabilityScore) AS SustainabilityCategory
FROM
    Destination;
```

--Calculate Average Efficiency Score

This table-valued function calculates the average efficiency score (energy, water, waste) for all accommodations.

```
CREATE FUNCTION fn_CalculateAverageEfficiencyScore ()
RETURNS TABLE
AS
RETURN
(
    SELECT
        AccommodationID,
        AccommodationName,
        DestinationID,
        CAST((EnergyEfficiencyScore + WaterConservationScore + WasteManagementScore) /
3.0 AS DECIMAL(10, 2)) AS AverageEfficiencyScore
    FROM
        Accommodation
    WHERE
        EnergyEfficiencyScore IS NOT NULL AND
        WaterConservationScore IS NOT NULL AND
        WasteManagementScore IS NOT NULL
);


SELECT *
FROM dbo.fn_CalculateAverageEfficiencyScore();
```

-- 4. DML Trigger

--Prevent Overcapacity Visitors

- **Enforce Data Integrity:** Prevents the insertion or update of rows where the number of average annual visitors exceeds the carrying capacity of the destination.

- **Automate Error Handling:** Automatically raises an error and stops the operation if the rule is violated.

```
CREATE TRIGGER trg_PreventOverCapacity
ON Destination
AFTER INSERT, UPDATE
AS
BEGIN
   IF EXISTS (
      SELECT 1
      FROM inserted
      WHERE AverageAnnualVisitors > CarryingCapacity
   )
   BEGIN
      ROLLBACK TRANSACTION;
      THROW 50001, 'Visitor count exceeds carrying capacity!', 1;
   END
END;
```

– Trigger Validation for Insert query

```
INSERT INTO Destination (DestinationName, Country, Region, AverageAnnualVisitors,
CarryingCapacity, SustainabilityScore)
VALUES ('Test Destination', 'Test Country', 'Test Region', 1000001, 1000000, 85.50);
```

– Trigger Validation for Update query

```
UPDATE Destination
SET AverageAnnualVisitors = 5000000
WHERE DestinationID = 1; -- Assuming DestinationID = 1 is a valid record and its
CarryingCapacity is less than 5000000.
```

-- Auto-Update Accommodation Efficiency Ratings

Ensures that whenever data in the Accommodation table is updated, the corresponding SustainabilityRating is automatically recalculated and updated using the latest efficiency score.

```
CREATE TRIGGER trg_UpdateSustainabilityRating
ON Accommodation
AFTER UPDATE
AS
BEGIN
   UPDATE A
   SET SustainabilityRating = E.AverageEfficiencyScore
   FROM Accommodation A
   CROSS APPLY dbo.fn_CalculateAverageEfficiencyScore() E
   WHERE A.AccommodationID = E.AccommodationID;
END;
```

– Trigger Validation for Update query

```
UPDATE Accommodation
SET
   EnergyEfficiencyScore = 85.00,
   WaterConservationScore = 90.00,
   WasteManagementScore = 80.00
WHERE AccommodationID = 1; -- Assuming AccommodationID = 1 exists.
```

– Query to verify if table is updated

```
SELECT
   AccommodationID,
   AccommodationName,
   EnergyEfficiencyScore,
   WaterConservationScore,
   WasteManagementScore,
   SustainabilityRating
FROM Accommodation
WHERE AccommodationID = 1;
```

-- 5. Column Data Encryption

-- Encrypting sensitive columns

```
ALTER TABLE Tourist
ADD EncryptedEmail VARBINARY(MAX);

-- Example: Encrypt existing emails
UPDATE Tourist
SET EncryptedEmail = ENCRYPTBYKEY(KEY_GUID('TouristKey'), Email);
```

– Encrypting Data During Insert

-- Step 1: Create a Master Key

```
CREATE MASTER KEY ENCRYPTION BY PASSWORD = 'StmsMaster@6789';
```

-- Step 2: Create a Certificate

```
CREATE CERTIFICATE CertificateName
WITH SUBJECT = 'STMS Certificate for Symmetric Key Encryption';
```

-- Step 3: Create a Symmetric Key

```
CREATE SYMMETRIC KEY SymmetricKeyName
WITH ALGORITHM = AES_256
ENCRYPTION BY CERTIFICATE CertificateName;
```

-- Step 4 : Alter table to add password column in tourist table

```
ALTER TABLE Tourist
ADD EncryptedPassword VARBINARY(MAX);
```

-- Step 5: Open the Symmetric Key, Encrypt, and Insert Data

```
OPEN SYMMETRIC KEY SymmetricKeyName
DECRYPTION BY CERTIFICATE CertificateName;
```

```
INSERT INTO Tourist (FirstName, LastName, Email, EncryptedPassword)
VALUES (
    'Richard',
    'Hall',
    'richard.hall@example.com',
    EncryptByKey(Key_GUID('SymmetricKeyName'), CONVERT(NVARCHAR(MAX),
'Stms@richard5678'))
);



-- Step 6: Close the Symmetric Key
CLOSE SYMMETRIC KEY SymmetricKeyName;
```

--To decrypt

```
OPEN SYMMETRIC KEY SymmetricKeyName
DECRYPTION BY CERTIFICATE CertificateName;

SELECT
    FirstName,
    LastName,
    Email,
    CONVERT(NVARCHAR(MAX), DecryptByKey(EncryptedPassword)) AS DecryptedPassword
FROM
    Tourist;

CLOSE SYMMETRIC KEY SymmetricKeyName;
```

-- 6. Non-Clustered Indexes

```
-- Index on DestinationName
CREATE NONCLUSTERED INDEX idx_DestinationName
ON Destination (DestinationName);

-- Index on Accommodation
CREATE NONCLUSTERED INDEX IX_Accommodation_Type
ON Accommodation (AccType);

-- Index on SustainabilityScore
CREATE NONCLUSTERED INDEX idx_SustainabilityScore
```

```
ON Destination (SustainabilityScore);

-- Index on Accommodation
CREATE NONCLUSTERED INDEX IX_Accommodation_DestinationID
ON Accommodation (DestinationID);


-- Index on Transportation Provider
CREATE NONCLUSTERED INDEX IX_TransportProvider_ProviderName
ON Transportation_Provider (ProviderName);
```