Ecotype Components:
SQL Code

sql

```
-- 1. Core Table for Ecotope
CREATE TABLE Ecotope (
   EcotopeID INT AUTO_INCREMENT PRIMARY KEY,
   Name VARCHAR(255) NOT NULL,
   Location GEOMETRY NOT NULL,  -- Using GEOMETRY for spatial data (point, polygon, etc.)
   TotalArea DECIMAL(10, 2),  -- Area in hectares or acres
   Description TEXT,
   ManagementStatus VARCHAR(100),
   LastUpdated DATETIME DEFAULT CURRENT_TIMESTAMP
);

-- 2. Soils Table
CREATE TABLE Soils (
   SoilID INT AUTO_INCREMENT PRIMARY KEY,
   EcotopeID INT,
   SoilType VARCHAR(100),
   pHLevel DECIMAL(4, 2),
   OrganicContent DECIMAL(5, 2),
   NutrientLevels JSON,  -- Storing nutrients as a JSON object
   ErosionRisk VARCHAR(50),
   Depth DECIMAL(5, 2),  -- Depth in cm
   LastSampleDate DATETIME,
```

```
    FOREIGN KEY (EcotopeID) REFERENCES Ecotope(EcotopeID)
);

-- 3. Water and Hydrology Table
CREATE TABLE WaterHydrology (
    WaterID INT AUTO_INCREMENT PRIMARY KEY,
    EcotopeID INT,
    WaterSource VARCHAR(100),
    WaterQuality JSON,  -- Key water quality indicators stored as JSON
    FlowRate DECIMAL(10, 2),  -- Flow rate in liters/sec or cubic meters/sec
    WaterTableDepth DECIMAL(5, 2),  -- Water table depth in meters
    Precipitation DECIMAL(10, 2),  -- Precipitation in mm
    FloodFrequency VARCHAR(50),
    LastHydrologyCheck DATETIME,
    FOREIGN KEY (EcotopeID) REFERENCES Ecotope(EcotopeID)
);

-- 4. Vegetation Table
CREATE TABLE Vegetation (
    VegetationID INT AUTO_INCREMENT PRIMARY KEY,
    EcotopeID INT,
    DominantSpecies JSON,  -- Storing dominant species as JSON
    InvasiveSpecies JSON,  -- Storing invasive species as JSON
    VegetationCoverage DECIMAL(5, 2),  -- Coverage percentage
    VegetationType VARCHAR(100),  -- E.g., forest, wetland, grassland
    BiodiversityIndex DECIMAL(5, 2),
```

```
    LastSurveyDate DATETIME,
    FOREIGN KEY (EcotopeID) REFERENCES Ecotope(EcotopeID)
);

-- 5. Wildlife and Fauna Table
CREATE TABLE WildlifeFauna (
    WildlifeID INT AUTO_INCREMENT PRIMARY KEY,
    EcotopeID INT,
    SpeciesComposition JSON,  -- List of species
    TrophicLevel VARCHAR(50),
    ConservationStatus VARCHAR(100),
    PopulationTrends JSON,  -- Trends in population sizes
    HabitatUse TEXT,
    LastWildlifeSurvey DATETIME,
    FOREIGN KEY (EcotopeID) REFERENCES Ecotope(EcotopeID)
);

-- 6. Climate and Microclimate Table
CREATE TABLE ClimateMicroclimate (
    ClimateID INT AUTO_INCREMENT PRIMARY KEY,
    EcotopeID INT,
    AverageTemperature DECIMAL(5, 2),  -- In degrees Celsius
    MicroclimateZones JSON,  -- Description of microclimates as JSON
    Humidity DECIMAL(5, 2),  -- Average humidity percentage
    FrostFrequency VARCHAR(50),
    SolarRadiation DECIMAL(10, 2),  -- In MJ/m²
```

```
    LastClimateUpdate DATETIME,
    FOREIGN KEY (EcotopeID) REFERENCES Ecotope(EcotopeID)
);

-- 7. Topography and Geology Table
CREATE TABLE TopographyGeology (
    GeologyID INT AUTO_INCREMENT PRIMARY KEY,
    EcotopeID INT,
    Elevation DECIMAL(10, 2),  -- Elevation in meters
    Slope DECIMAL(5, 2),  -- Slope gradient in degrees
    Aspect VARCHAR(50),  -- Direction of slope (north-facing, etc.)
    RockType VARCHAR(100),
    SoilParentMaterial VARCHAR(255),
    GeologicalFeatures TEXT,  -- Notable features like cliffs or caves
    LastTopoUpdate DATETIME,
    FOREIGN KEY (EcotopeID) REFERENCES Ecotope(EcotopeID)
);

-- 8. Ecosystem Processes Table
CREATE TABLE EcosystemProcesses (
    ProcessID INT AUTO_INCREMENT PRIMARY KEY,
    EcotopeID INT,
    NutrientCyclingRate DECIMAL(10, 2),  -- Rate of nutrient cycling
    PollinationRate DECIMAL(5, 2),  -- Pollination success rate as percentage
    EnergyFlow TEXT,  -- Description of energy flow
    DecompositionRate DECIMAL(10, 2),  -- Decomposition rate in g/m^2/day
```

```
    LastProcessUpdate DATETIME,
    FOREIGN KEY (EcotopeID) REFERENCES Ecotope(EcotopeID)
);

-- 9. Human Use and Impacts Table
CREATE TABLE HumanUseImpacts (
    HumanImpactID INT AUTO_INCREMENT PRIMARY KEY,
    EcotopeID INT,
    LandUseType VARCHAR(100),  -- Type of land use, e.g., agriculture, urban
    PollutionLevels JSON,  -- Pollutant concentrations in air, water, soil
    RecreationActivity VARCHAR(100),
    ResourceExtraction TEXT,  -- Description of resource extraction (mining, logging)
    ImpactAssessment TEXT,  -- Impact description (positive/negative)
    LastImpactUpdate DATETIME,
    FOREIGN KEY (EcotopeID) REFERENCES Ecotope(EcotopeID)
);

-- 10. Ecological Services Table
CREATE TABLE EcologicalServices (
    ServiceID INT AUTO_INCREMENT PRIMARY KEY,
    EcotopeID INT,
    CarbonSequestration DECIMAL(10, 2),  -- Carbon sequestered in tons per hectare/year
    WaterPurification DECIMAL(5, 2),  -- Efficiency of water purification (% reduction)
    HabitatCreation TEXT,  -- Description of habitats created or restored
    FloodMitigation DECIMAL(5, 2),  -- Percentage of water retained for flood mitigation
    ErosionControl DECIMAL(5, 2),  -- Erosion control efficiency (% reduction)
```

```
    LastServiceUpdate DATETIME,
    FOREIGN KEY (EcotopeID) REFERENCES Ecotope(EcotopeID)
);
```

Key Features in the SQL Schema:

- **Primary Keys and Foreign Keys**: Each table has a primary key, typically an auto-incrementing integer (EcotopeID, SoilID, etc.), and foreign keys link the tables to the central Ecotope table.
- **JSON Fields**: For flexible data structures (e.g., species composition, nutrient levels, pollution levels), JSON is used to store lists and nested data.
- **Spatial Data**: The GEOMETRY data type is used in the Location field of the Ecotope table to store geographical information, enabling spatial queries.
- **Temporal Tracking**: Fields like LastUpdated, LastSampleDate, and LastSurveyDate track when the data was last updated.

Additional Considerations:

- **Indexes**: You may want to add indexes to frequently queried fields (e.g., EcotopeID in component tables) for performance.
- **Foreign Key Constraints**: These ensure referential integrity, meaning records in the component tables must correspond to valid entries in the Ecotope table.
- **GIS Extensions**: MySQL supports spatial extensions like ST_Distance() and ST_Within() that you can use with the GEOMETRY field in the Ecotope table for spatial analysis.

4o