Trail & Hiking Database

Joji Araki & William Legault

About

A Trail and Hiking Database that is a comprehensive resource for outdoor enthusiasts, providing detailed information about hiking trails.

Overview

Trail Information:

- **Trail Name**
- **Location**
- **Distance**
- **Difficulty Level** (Easiest, Moderate, Moderately Strenuous, Strenuous, Very Strenuous)
- **Elevation Change**
- **Trail Type (**loop, out-and-back, or point-to-point)

Description and Features:

- **Trail Description**
- **Scenic Views**
- **Flora and Fauna**
- **Geological Features**

Amenities and Facilities:

- **Parking**
- **Restrooms**
- **Water Sources**
- **Picnic Areas**

Trail Conditions and Alerts:

- **Current Conditions**
- **Alerts and Closures**

User Reviews and Ratings:

- **User Ratings**
- **User Reviews**

Map:

- **Trail Map**
- **GPS Coordinates**

Safety Information:

- **Trailhead Signage**
- **Emergency Contacts**
- **Wildlife Safety**

User Contributions:

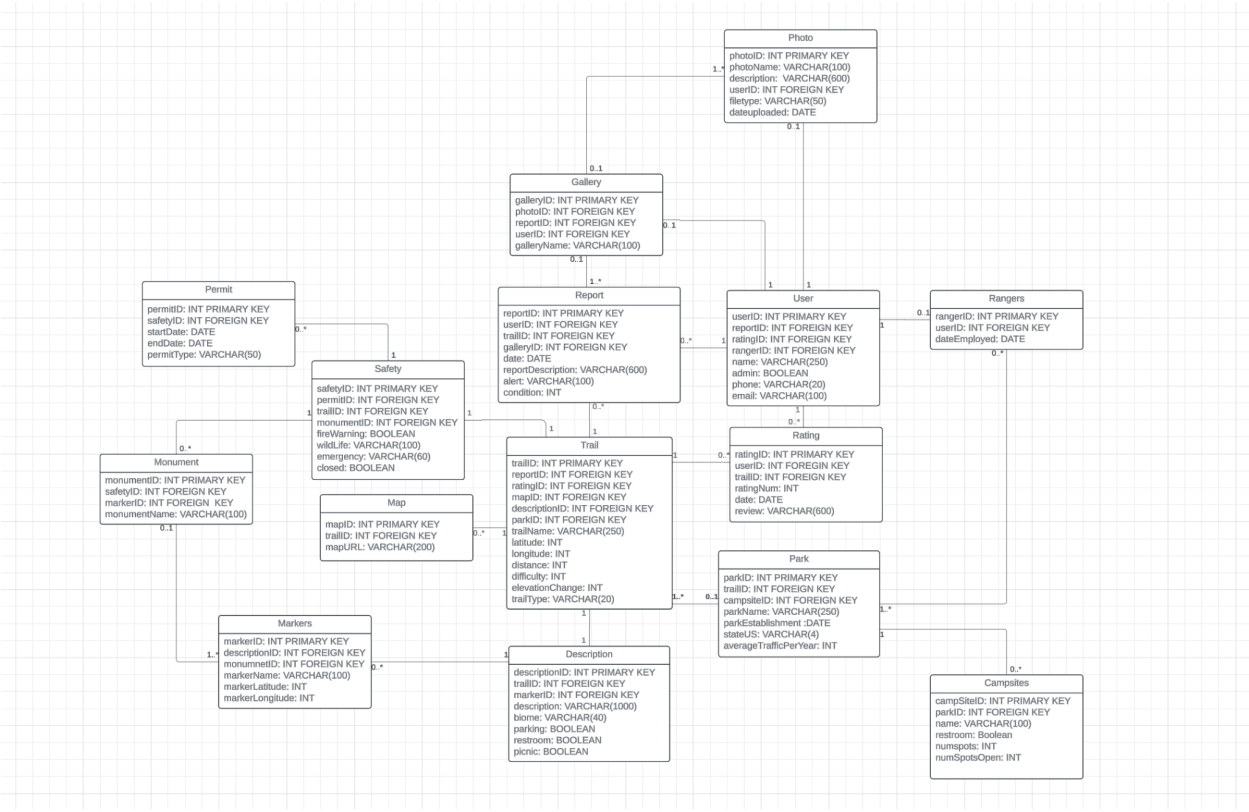
- **Photo Gallery**
- **Trail Logs**

Rules

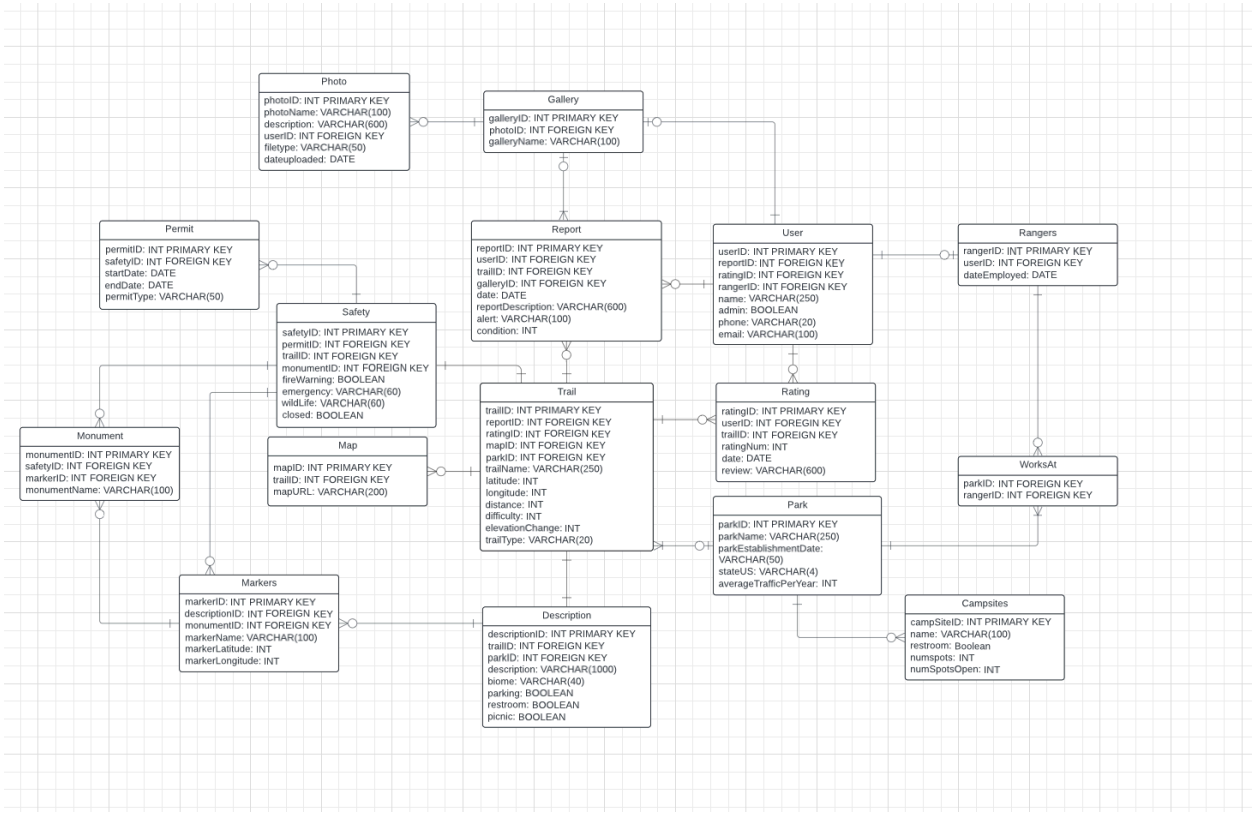
Nouns | Verbs

- Trail information which will have the name of the trail, location, distance, difficulty level based on elevation gain and distance, and a trail type
- Each trail will have an identification id and connected to it will be a description and features of the trail
- Connected to the description and features will be information about amenities and facilities such as bathroom, water, parking and picnic
- There will also be trail conditions and alerts that are updated in real time in order for hikers to stay safe
- User reviews with both user information, review, and rating associated with each
- Users able to write and submit reports of safety or about trail
- Safety information provided for each trail can be linked to condition and alerts
- Add and edit trails
- Administrative accounts
- Map link and relative GPS coordinate area
- Trails must have name, location, distance, difficulty and type
- Users can update trail condition with data associated
- Safety alerts only by admin accounts
- GPS start and end points

UML Diagram



Logical Model

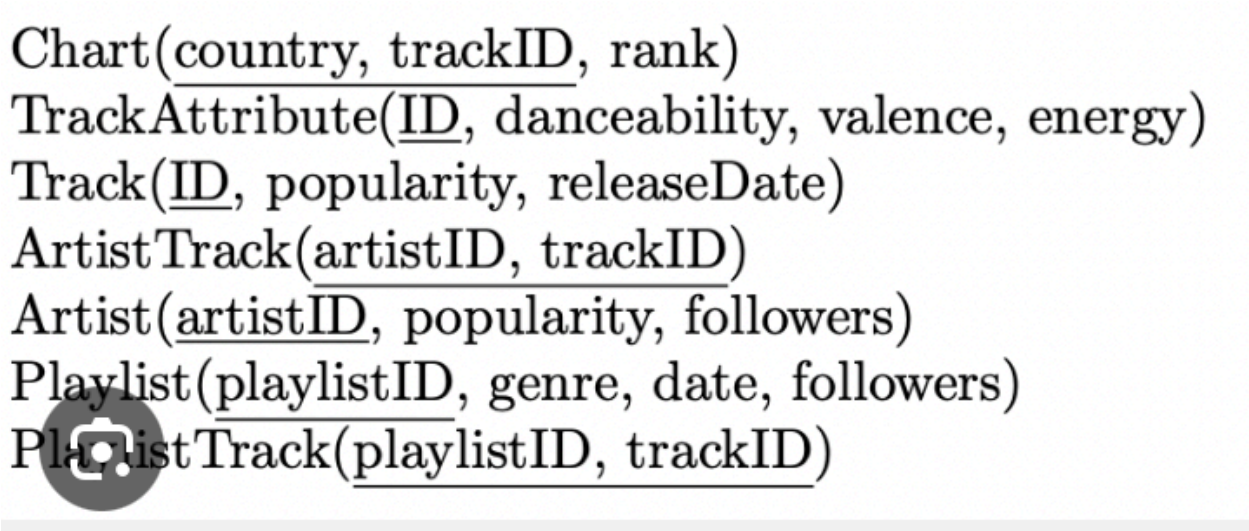


Relational Schema

User(userID, *reportID, ratingID, rangerID,* name, admin, phone, email)
Report(reportID, *userID, trailID, galleryID,* date, reportDescription, alert, condition)
Gallery(galleryID, *photoID,* galleryName)
Photo(photoID, photoName, description, fileType, dateUploaded)
Trail(trailID, *reportID, ratingID, mapID, parkID,* trailName, latitude, longitude, distance, difficulty, elevationChange, trailType)
Rating(ratingID, *userID, trailID,* ratingNum, date, review)
Map(mapID, mapURL)
Safety(safetyID, *permitID, trailID, monumentID,* fireWarning, emergency, wildLife, closed)
Permit(permitID, *safetyID,* startDate, endDate, permitType)
Monument(monumentID, *safetyID, markerID,* monumentName)
Markers(markerID, *descriptionID, markerID,* markerName, markerLatitude, markerLongitude)
Description(descriptionID, *trailID, parkID,* description, biome, parking, restroom, picnic)
Park(parkID, parkName, parkEstablishmentDate, stateUS, averageTrafficPerYear)
Campsites(campsiteID, name, restroom, numSpots, numSpotsOpen)
Rangers(rangerID, *userID,* dateEmployed)

WorksAt(*rangerID*, *parkID*)

Ex.



SQL Data Definition Statements

```
CREATE TABLE Trail

(trailID INT AUTO_INCREMENT PRIMARY KEY,

reportID INT,

ratingID INT,

mapID INT,

parkID INT,

descriptionID INT NOT NULL,

trailName VARCHAR(250) NOT NULL,
```

```
latitude INT NOT NULL,

longitude INT NOT NULL,

distance INT NOT NULL,

difficulty INT NOT NULL,

elevationChange INT NOT NULL,

trailType VARCHAR(20) NOT NULL,

FOREIGN KEY (reportID) REFERENCES Report(reportID),

FOREIGN KEY (ratingID) REFERENCES Rating(ratingID),

FOREIGN KEY (mapID) REFERENCES Map(mapID),

FOREIGN KEY (parkID) REFERENCES Park(parkID),

FOREIGN KEY (descriptionID) REFERENCES Description(descriptionID)

);




CREATE TABLE Description

(descriptionID INT,

trailID INT NOT NULL,

markerID INT NOT NULL,

descript VARCHAR(600),

biome VARCHAR(40),

parking BOOLEAN,

restroom BOOLEAN,
```

```
picnic BOOLEAN,

FOREIGN KEY (trailID) REFERENCES Trail(trailID),

FOREIGN KEY (markerID) REFERENCES Marker(markerID)

);




CREATE TABLE Map

(mapID INT AUTO_INCREMENT PRIMARY KEY,

trailID INT NOT NULL,

mapURL VARCHAR(200) NOT NULL,

FOREIGN KEY (trailID) REFERENCES Trail(trailID)

);




CREATE TABLE Rating

(ratingID INT AUTO_INCREMENT PRIMARY KEY,

userID INT NOT NULL,

trailID INT NOT NULL,

ratingNum INT NOT NULL,

date DATE NOT NULL,

review VARCHAR(600),

FOREIGN KEY (trailID) REFERENCES Trail(trailID),
```

```
FOREIGN KEY (userID) REFERENCES User(userID)

);

--



CREATE TABLE User

(userID INT AUTO_INCREMENT PRIMARY KEY,

reportID INT,

galleryID INT,

photoID INT,

rangerID INT,

name VARCHAR(250) NOT NULL,

ratingID INT,

admin BOOLEAN NOT NULL,

phone VARCHAR(20) NOT NULL,

email VARCHAR(100) NOT NULL,

FOREIGN KEY (reportID) REFERENCES Report(reportID),

FOREIGN KEY (rangerID) REFERENCES Rangers(rangerID),

FOREIGN KEY (photoID) REFERENCES Photo(photoID),

FOREIGN KEY (galleryID) REFERENCES Gallery(galleryID),

FOREIGN KEY (ratingID) REFERENCES Rating(ratingID)

);
```

```
CREATE TABLE Report

(reportID INT AUTO_INCREMENT PRIMARY KEY,

userID INT NOT NULL,

trailID INT NOT NULL,

galleryID INT,

date DATE NOT NULL,

reportDescription VARCHAR(600) NOT NULL,

alert VARCHAR(100),

condition INT,

FOREIGN KEY (trailID) REFERENCES Trail(trailID),

FOREIGN KEY (userID) REFERENCES User(userID),

FOREIGN KEY (galleryID) REFERENCES Gallery(galleryID)

);




CREATE TABLE Gallery

(galleryID INT AUTO_INCREMENT PRIMARY KEY,

photoID INT NOT NULL,

reportID INT,

userID INT NOT NULL,

galleryName VARCHAR(100),
```

```
FOREIGN KEY (photoID) REFERENCES Photo(photoID),

FOREIGN KEY (reportID) REFERENCES Report(reportID),

FOREIGN KEY (userID) REFERENCES User(userID)

);




CREATE TABLE Photo

(photoID INT AUTO_INCREMENT PRIMARY KEY,

photoName VARCHAR(100) NOT NULL,

Descript VARCHAR(600),

userID INT NOT NULL,

fileType VARCHAR NOT NULL,

dateUploaded DATE NOT NULL,

FOREIGN KEY (userID) REFERENCES User(userID)

);




CREATE TABLE Rangers

(rangerID INT AUTO_INCREMENT PRIMARY KEY,

dateEmployed DATE NOT NULL,

userID INT NOT NULL,

FOREIGN KEY (userID) REFERENCES User(userID)
```

```
);



CREATE TABLE Park

(parkID INT AUTO_INCREMENT PRIMARY KEY,

trailID INT NOT NULL,

campsiteID INT,

parkName VARCHAR(250) NOT NULL,

parkEstablishment DATE NOT NULL,

stateUS VARCHAR(4) NOT NULL,

averageTrafficperYear INT,

FOREIGN KEY (trailID) REFERENCES Trail(trailID),

FOREIGN KEY (campsiteID) REFERENCES Campsites(campsiteID)

);



CREATE TABLE Campsites

(campsiteID INT AUTO_INCREMENT PRIMARY KEY,

name VARCHAR(100) NOT NULL,

parkID INT NOT NULL,

restroom BOOLEAN,

numspots INT,
```

```
numSpotsOpen INT,

FOREIGN KEY (parkID) REFERENCES Park(parkID)

);









CREATE TABLE Safety

(safetyID INT AUTO_INCREMENT PRIMARY KEY,

trailID INT NOT NULL,

permitID INT,

monumentID INT,

fireWarning BOOLEAN NOT NULL,

emergency VARCHAR(60),

wildLife VARCHAR(60),

closed BOOLEAN NOT NULL,

FOREIGN KEY (trailID) REFERENCES Trail(trailID),

FOREIGN KEY (permitID) REFERENCES Permit(permitID),

FOREIGN KEY (monumentID) REFERENCES Monument(monumentID),

);
```

```
CREATE TABLE Permit

(permitID INT AUTO_INCREMENT PRIMARY KEY,

startDate DATE NOT NULL,

safetyID INT,

endDate DATE NOT NULL,

permitType VARCHAR(50) NOT NULL,

FOREIGN KEY (safetyID) REFERENCES Safety(safetyID)

);




CREATE TABLE Monument

(monumentID INT AUTO_INCREMENT PRIMARY KEY,

safetyID INT NOT NULL,

monumentName VARCHAR(100) NOT NULL,

markerID INT NOT NULL,

FOREIGN KEY (safetyID) REFERENCES Safety(safetyID),

FOREIGN KEY (markerID) REFERENCES Marker(markerID)

);




CREATE TABLE Markers

(markerID INT AUTO_INCREMENT PRIMARY KEY,
```

```
descriptionID INT NOT NULL,

monumentID INT,

markerName VARCHAR(100),

markerLatitude INT,

markerLongitude INT,

FOREIGN KEY (descriptionID) REFERENCES Description(descriptionID),

FOREIGN KEY (monumentID) REFERENCES Monument(MonumentID)

);



CREATE TABLE Stores

(galleryID INT,

photoID INT,

FOREIGN KEY (galleryID) REFERENCES Gallery(galleryID),

FOREIGN KEY (photoID) REFERENCES Photo(photoID)

);



CREATE TABLE WorksAt

(parkID INT,

rangerID INT,

FOREIGN KEY (parkID) REFERENCES Park(parkID),

FOREIGN KEY (rangerID) REFERENCES Rangers(rangerID)

);
```

Data Values

6

Database Queries:

```
-- 1. Retrieve a list of trail names, descriptions, and URL to online map

SELECT Trail.trailName, Description.description, Map.mapURL

FROM Trail

JOIN Description ON Trail.trailID = Description.trailID

JOIN Map ON Trail.trailID = Map.trailID;

-- 2. Retrieves all safety info on trails with a difficulty rating higher than 3

SELECT *

FROM Safety

WHERE trailID IN (

SELECT trailID

FROM Trail

WHERE difficulty > 3

);



-- 3. Retrieve list of trails
```

```sql
SELECT parkID, COUNT(trailID) AS trailCount

FROM Trail

GROUP BY parkID

HAVING trailCount > 5;

-- 4. Retrieve trails with a difficulty level above 2 and a distance greater than 5 or greater elevation change than 500

SELECT *

FROM Trail

WHERE (distance > 5 OR elevationChange > 500) AND difficulty > 2;



-- 5. Retrieve names of administrators with reports dating back to beginning of 2023 and the descriptions as well as trail names

SELECT User.name, Report.reportDescription, Trail.trailName

FROM User

JOIN Report ON User.userID = Report.userID

JOIN Trail ON Report.trailID = Trail.trailID

WHERE Report.date > '01-01-2023' AND User.admin = true;
```