Note: Application is running on port 8501.

Andrew Bell, Max Christ
Professor Stoyanovich

December 6, 2020
Principles of Database Systems

Hiking Application: Entity Sets, Relationship Sets, Business Rules, and Schema

The following project is to create a relational database for recording data about hiking, and information related to hiking. The main focus of this project is on the hiker, and the places, activities, and objects which a hiker will encounter. This database can be useful for hikers who want to plan their trips. It can also be useful for park personnel who can keep track of important information, like the frequency of injuries in specific trails, parks, or mountains. The database is described in full detail below.

Entity Sets (Attributes are listed for each, and the primary keys are bolded):

- Hikers
  - **Name**
  - **DOB**
  - Years_experience
- Supplies (weak entity set for Hiker)
  - **Name**
  - Quantity
  - Weight
- Injuries
  - **Name**
  - **Body_part**
  - Severity
- Trails
  - Trail_length
  - **Trailhead**
  - Elevation_change
  - **Name**
  - Difficulty
- Mountains
  - Elevation
  - Name
  - **Coordinates**
- Mountain_ranges
  - **Name**
  - Tallest_peak
  - Country
- National_parks
  - **Name**
  - **Year_founded**

    - Country
    - Park_size
- Rangers
    - **Name**
    - **DOB**
    - Address
    - Phone
- Campgrounds
    - **Address**
    - Capacity
    - Has_tent_sites
    - **Name**
    - Price

Relationships:

- A hiker **hikes** trails
- Trails **exist on** mountains
- Rangers **manage** National Parks
- Mountains are **part of** mountain ranges
- Mountains **are in** National Parks
- Hikers **own** supplies
- Hikers **camp** in campgrounds
- Campgrounds are **located in** National Parks
- Hikers can get **injured** on trails

Business Rules:

A hiker is uniquely identified by a name, and date of birth. A hiker can hike trails, and must have hiked at least one trail. Trails are uniquely identified by a name, and the location of the trailhead. Trails have to have been hiked by at least one hiker. A trail must exist on exactly one mountain. A mountain is uniquely identified by its coordinates. A mountain has to be traversed by at least 1 trail. A mountain range contains at least one mountain, and is uniquely identified by its name. A mountain can be a part of at most one mountain range. A mountain can be in at most one National Park. Hikers can own supplies, and the system only tracks supplies which are owned by a specific hiker. A supply is uniquely identified by its owner's name and date of birth, and its own name. Campgrounds are located in exactly 1 National Park. Campgrounds are uniquely identified by a name and address. Campgrounds either have sites for tents, or they do not (meaning they only have sites for RVs). A National Park is uniquely identified by a name, and the year it was founded. National Parks can have multiple campgrounds. Rangers manage exactly 1 National Park, and are uniquely identified by a name and date of birth. National Park must have at least 1 ranger. A hiker can be injured on a trail. An Injury is uniquely identified by its name and the affected body part, and injuries have severity levels corresponding to a number from 1 to 10. Only injuries made on trails and made by hikers in the system are tracked by the system. Hikers camp in exactly one campground at a time. The system only keeps track of where a hiker is currently camping.

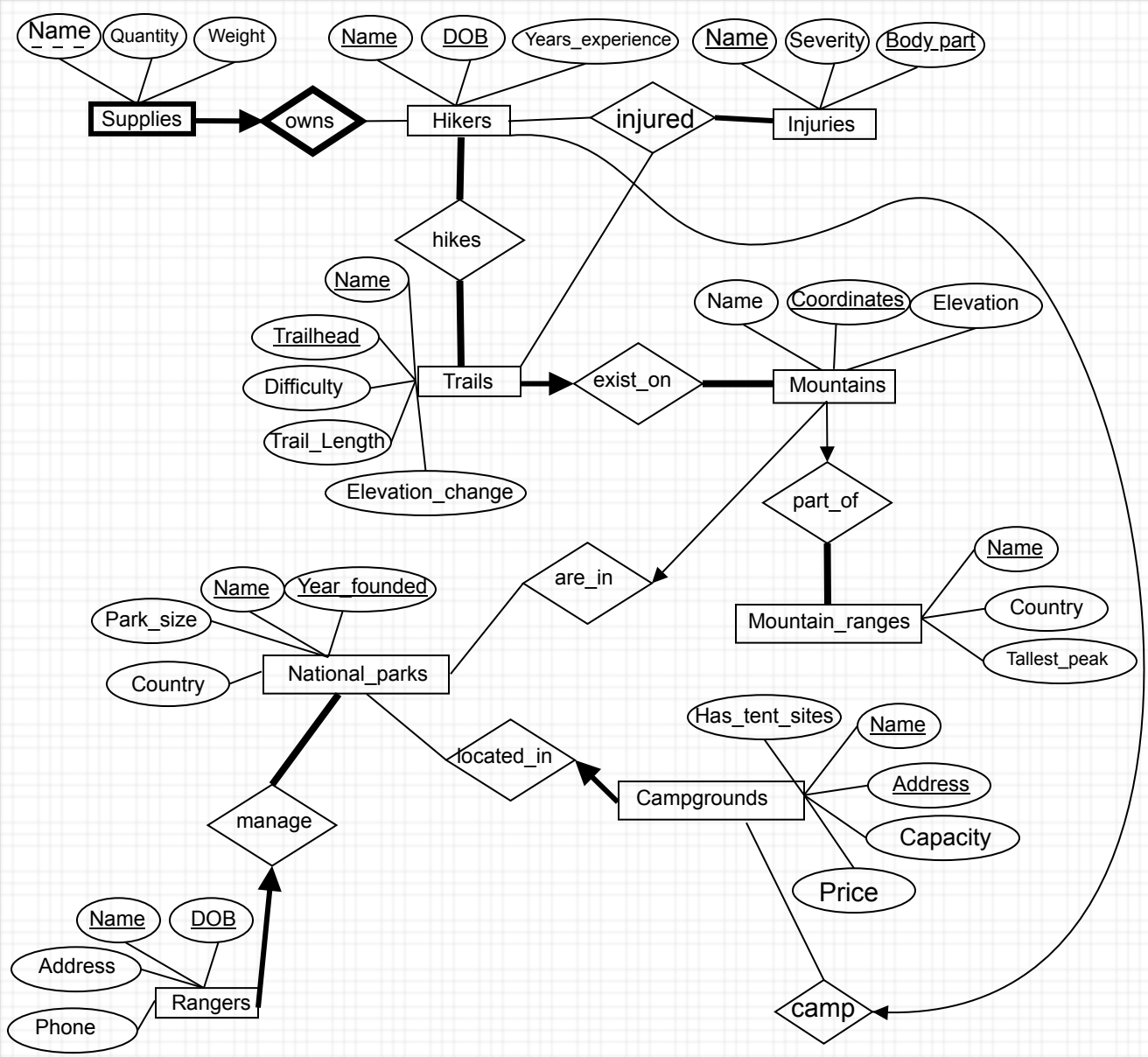
Data Acquisition:

The database has been populated manually with real and realistic data. In other words, there weren't any existing databases that were used to populate the tables in this project. For example, park, mountain, campground, and trail related data was generated mostly by using basic internet searches, and data related to hikers and rangers was fabricated in order to resemble lifelike information.

Application User Interactions:

The application will consist of a webpage with various input formats for users to view the desired data, including the following widgets.

- The user will input text to display all of the trails at a given national park.
- The user can query for the number of injuries that occurred on a given trail, by using a multi-select drop down menu.
- The user can use a drop down menu to query the number of total supplies across all hikers by supply name.
- The user can use a drop down of hikers to query for the hiker that has the total highest elevation gain / most miles hiked.
- Display the mountains in national parks (with checkboxes for each park)
- Give national parks with the cheapest campgrounds (under a certain threshold) (slider)


Name
Quantity
Weight
Supplies
owns
Name
DOB
Years_experience
Hikers
Name
Severity
Body_part
injured
Injuries
hikes
Name
Trailhead
Difficulty
Trail_Length
Elevation_change
Trails
exist_on
Name
Coordinates
Elevation
Mountains
part_of
Name
Country
Tallest_peak
Mountain_ranges
are_in
Name
Year_founded
Park_size
Country
National_parks
Has_tent_sites
Name
Address
Capacity
Price
Campgrounds
located_in
manage
Name
DOB
Address
Phone
Rangers
camp

```
drop table if exists Hikers, Supplies, Injuries, hikes, Trails, Mountains, National_parks,
Mountain_ranges, Campgrounds, Rangers CASCADE;

/*The following table contains the Mountain_ranges entity*/
create table Mountain_ranges (
        name char(32) primary key,
        country char(32),
        Tallest_peak char(32)
);

/*The following table contains the National_parks entity set*/
create table National_parks (
        name char(32),
        year_founded integer,
        park_size float,
        country char(32),
        primary key(name,year_founded)
);

/*The following table contains the Mountains entity set and the part_of and are_in relations*/
create table Mountains (
        name char(32),
        elevation float,
        coordinates char(50) primary key,
        part_of_mountain_range char(32),
        in_national_park_name char(32),
        in_national_park_year_founded integer,
        foreign key (part_of_mountain_range) references Mountain_ranges(name),
        foreign key (in_national_park_name, in_national_park_year_founded) references
National_parks(name, year_founded)
);

/*The following table contains the campground entity set and the located_in relation*/
create table Campgrounds (
        name char(32),
        address char(250),
        has_tent_sites boolean,
        capacity integer,
        price float,
        located_in_park_name char(32) not null,
        located_in_park_year_founded integer not null,
        primary key(name,address),
        foreign key (located_in_park_name, located_in_park_year_founded) references
National_parks(name, year_founded)
);

/*The following table contains the Hikers entity set and the camp relation*/
create table Hikers (
        name char(32),
        dob date,
        years_experience integer,
        camp_name char(32),
        camp_address char(250),
        primary key(name, dob),
```

```
        foreign key (camp_name, camp_address) references Campgrounds(name,address)
);

/*The following table contains the supplies weak entity set and its relation to Hikers*/
create table Supplies (
        name char(32),
        quantity integer,
        weight float,
        owned_by_name char(32) not null,
        owned_by_dob date not null,
        primary key(name, owned_by_name, owned_by_dob),
        foreign key (owned_by_name, owned_by_dob) references Hikers(name, dob) on delete
cascade
);

/*The following table contains the trails entity set and the exist_on relation*/
create table Trails (
        name char(32),
        trailhead char(32),
        difficulty char(12),
        trail_length float,
        elevation_change float,
        mountain_coordinates char(50) not null,
        primary key(name, trailhead),
        foreign key (mountain_coordinates) references Mountains(coordinates)
);

/*The following table contains the injuries entity set and the injured ternary relationship with
hikers and trails */
create table Injuries (
        name char(32),
        body_part char(32),
        severity integer,
        hiker_name char(32),
        hiker_dob date,
        on_trail_name char(32),
        on_trail_trailhead char(32),
        primary key(name, body_part, hiker_name, hiker_dob, on_trail_name, on_trail_trailhead),
        foreign key (hiker_name, hiker_dob) references Hikers(name, dob),
        foreign key (on_trail_name, on_trail_trailhead) references Trails(name, trailhead)
);

/*The following table contains the hikes relation*/
create table hikes (
        hiker_name char(32),
        hiker_dob date,
        trail_name char(32),
        trail_trailhead char(32),
        primary key(hiker_name,hiker_dob,trail_name,trail_trailhead),
        foreign key (hiker_name, hiker_dob) references Hikers(name, dob),
        foreign key (trail_name, trail_trailhead) references Trails(name, trailhead)
);

/*The following table contains the Rangers entity set and manage relation*/
```

```
create table Rangers (
	name char(32),
	dob date,
	address char(250),
	phone char(32),
	manages_park_name char(32) not null,
	manages_park_year_founded integer not null,
	primary key(name, dob),
	foreign key (manages_park_name, manages_park_year_founded) references
National_parks(name, year_founded)
);
```