Ambrelle Palmer and Brook Hagen
CS340 Final Project

Outdoor Adventures Database

If you've ever planned a multi-day outdoor trip you probably know how complicated it can be with so many possible things to plan for. You need to plan your location, routes, gear, food and the more people going on your trip the more things you have to worry about. Enter the Outdoor Adventures database. The purpose of this database is to allow users to store all information related to their trip in this one place in order to streamline the planning process. This database allows users to add photos, create lists, plan adventures, and join other users in "Tribes" based on their outdoor interests.

DATABASE OUTLINE

The entities in the Outdoor Adventure database are users, photos, adventures, lists, and list items. The User entity's attributes are first and last names, date of birth, home zip code, tribes, and a unique identification number which is also the primary key. The combination of first name and last name are a unique key for a user. Tribes are an attribute of users and are comprised of users. The tribe table includes a user identification number, which is also a foreign key, and a tribe name. The primary key for this table is the combination of the user ID and tribe name. Users can be a part of more than one tribe.

The User entity has a many to many relationship with the Photo entity. Any user can have zero to many photos and photos can be shared amongst many users. The Photo entity has three attributes; a unique identification number, which is it's primary key, a date that the photo was added, and user ID as a foreign key and whose value can be null. This relationship has partial participation from both the User entity and the Photo entity because a user does not need a photo in order to exist and a photo does not need a user in order to exist.

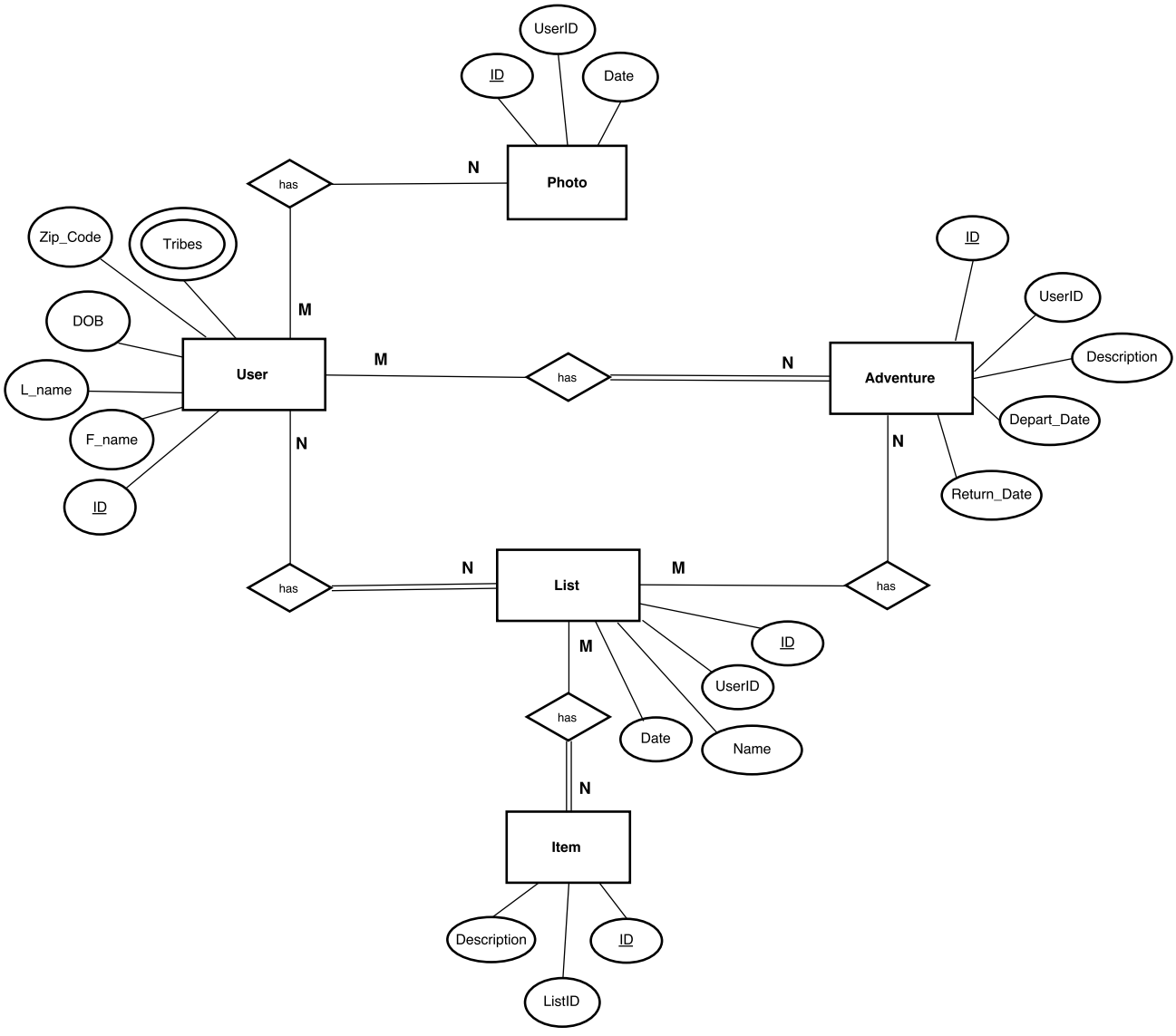
Similar to the relationship with the Photo entity, users can create adventures. This is also a many to many relationship as any user can have zero to many adventures and adventures can be shared amongst many users. The Adventure entity has five attributes; a unique identification number, which is the primary key, a user ID for the user that created the adventure as a foreign key, a description for the adventure, the start date of the adventure, and the return date from the adventure. This is a total participation relationship for the Adventure entity as all Adventure entities must participate in a relationship with a user. The table that represents this relationship references both the user id and adventure id as foreign keys and uses the combination of the two as the primary key.

The last relationship that the User entity is a part of is with the List entity. This is a many to many relationship as a user can have many lists and lists can be shared by many users. The attributes associated with the List entity are a unique identification number as the primary key, the user ID of the user who created the list as a foreign key, the name of the list, and the date the list was created. The List entity's relationship with the User entity is a total participation relationship for List because a list must be related to a user in order to exist. The List entity also has a relationship with the Adventure entity. This is a many to many relationship as an adventure can have many lists and a list can be associated with many adventures. This

relationship is represented in the List table by referencing the adventure ID as a foreign key. This relationship has partial participation on both sides since an adventure does not have to have a list in order to exist nor does a list have to have an adventure in order to exist.

The List entity also has a relationship with the Item entity. This is a many to many relationship because many items can be on a list and any item can be on many lists. The Item entity's attributes are a unique identification number as a primary key, a list ID number for the list it's associated with as a foreign key, and a description of the item. The Item entity's relationship with the List entity is total participation for Item as every member of the Item entity must have a relationship with the List entity in order to exist.

ER DIAGRAM


UserID
ID
Date
Photo
has
N
Zip_Code
Tribes
DOB
L_name
F_name
ID
User
M
M
N
has
N
ID
UserID
Description
Depart_Date
Return_Date
Adventure
N
has
N
List
M
has
has
M
ID
UserID
Date
Name
N
Item
Description
ID
ListID

DATABASE SCHEMA

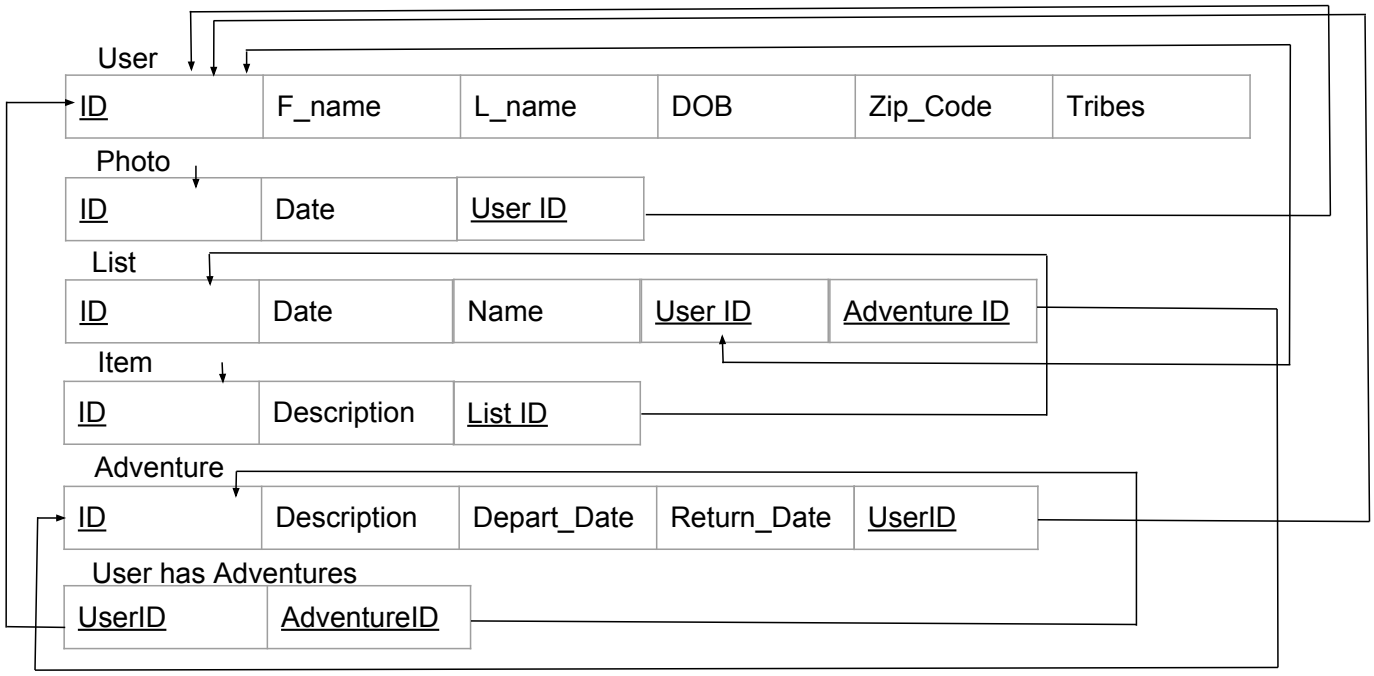


TABLE CREATION QUERIES

User Table

```
CREATE TABLE user(
    id int(10) NOT NULL AUTO_INCREMENT,
    f_name varchar(20) NOT NULL,
    l_name varchar(20) NOT NULL,
    dob date NOT NULL,
    zip_code int(5) NOT NULL,
    UNIQUE KEY (f_name, l_name),
    PRIMARY KEY (id)) ENGINE=InnoDB;
```

Tribes Table

```
CREATE TABLE tribes(
    uid int(10) NOT NULL DEFAULT '0',
    tribe_name varchar(20) NOT NULL,
    PRIMARY KEY (uid, tribe_name),
    FOREIGN KEY (uid) REFERENCES user(id)) ENGINE=InnoDB;
```

Photo Table

```
CREATE TABLE photo(
    id int(10) NOT NULL AUTO_INCREMENT,
    uid int(10) DEFAULT NULL,
    date_created TIMESTAMP NULL DEFAULT NULL,
    PRIMARY KEY (id),
    FOREIGN KEY (uid) REFERENCES user (id) ON DELETE CASCADE ON UPDATE
    CASCADE) ENGINE=InnoDB;
```

List Table

```
CREATE TABLE list(
    id int(10) NOT NULL AUTO_INCREMENT,
    uid int(10) NOT NULL,
    aid int(10) DEFAULT NULL,
    name varchar(255) NOT NULL,
    date_created TIMESTAMP DEFAULT NOW(),
    PRIMARY KEY (id),
    FOREIGN KEY (uid) REFERENCES user (id),
    FOREIGN KEY (aid) REFERENCES adventures(id) ON DELETE CASCADE ON
    UPDATE CASCADE) ENGINE=InnoDB;
```

Item Table

```
CREATE TABLE item(
    id int(10) NOT NULL AUTO_INCREMENT,
    lid int(10) NOT NULL,
    description varchar(100) NOT NULL,
    PRIMARY KEY (id),
    FOREIGN KEY (lid) REFERENCES list (id)) ENGINE=InnoDB;
```

Adventures Table

```
CREATE TABLE adventures(
    id int(10) NOT NULL AUTO_INCREMENT,
    description varchar(255) NOT NULL,
    uid int(10) NOT NULL,
    depart_date date DEFAULT NULL,
    return_date date DEFAULT NULL,
    PRIMARY KEY (id),
    FOREIGN KEY (uid) REFERENCES user(id)) ENGINE=InnoDB;
```

User-Adventures Table

```
CREATE TABLE user_adventure(
    uid int(10) NOT NULL,
    aid int(10) NOT NULL,
    PRIMARY KEY (uid, aid),
    FOREIGN KEY (uid) REFERENCES user(id),
    FOREIGN KEY (aid) REFERENCES adventures(id) ON DELETE CASCADE ON
    UPDATE CASCADE) ENGINE=InnoDB;
```

GENERAL USE QUERIES

INSERT QUERIES

Add a new User

```
INSERT INTO user (f_name, l_name, dob, zip_code)
    VALUES ([f_name], [l_name], [dob], [zip_code]);
```

Add a Tribe to a User

```
    INSERT INTO tribes (uid, tribe_name)
    VALUES ((SELECT id FROM user WHERE id = [uid]), [tribe_name]);
```

Add a New Photo

```
INSERT INTO photo (uid, date_created)
    VALUES ((SELECT id FROM user WHERE id =[uid]), CURRENT TIMESTAMP);
```

Add a New List

```
INSERT INTO list (uid, aid, name, date_created)
    VALUES ((SELECT id FROM user WHERE id = [uid]),
    (SELECT id FROM adventures WHERE id = [aid]), [name], CURRENT TIMESTAMP);
```

Add an Item to a List

```
INSERT INTO item (list_id, description)
    VALUES((SELECT id FROM list WHERE id = [lid]), [description]);
```

Add a New Adventure

```
INSERT INTO adventures (uid, description, depart_date, return_date)
    VALUES([description], (SELECT id FROM user WHERE id = [uid]), [depart_date],
    [return_date]);
```

Add an Additional User to an Adventure

```
INSERT INTO user_adventure(uid, aid)
    VALUES((SELECT id FROM user WHERE id = [uid]),
    (SELECT id FROM adventures WHERE id = [adventureID]));
```

SELECT (All) QUERIES

Get All Users

```
SELECT * FROM user;
```

Get All Tribes

```
SELECT * FROM tribes;
```

Get All Photos

```
SELECT * FROM photo;
```

Get All Lists

```
SELECT * FROM list;
```

Get All Items on Every List

```
SELECT * FROM item;
```

Get All Adventures

```
SELECT * FROM adventures;
```

FILTER QUERIES

Get All Users of a Particular Tribe

```
SELECT f_name, l_name FROM user u
        INNER JOIN tribes t ON t.uid = u.id
        WHERE t.tribe_name = '[some_activity]';
```

Get User's Photos by User's Id

```
SELECT id, date_created FROM photo p
        INNER JOIN user ON photo.uid = user.id
        WHERE user.id = '[user.id]';
```

Get All Items on a List By List Name

```
SELECT description FROM item i
        INNER JOIN list l ON i.lid = l.id
        WHERE l.name = '[listName]';
```

Get All Users on a Particular Adventure

```
SELECT user.f_name, user.l_name FROM user
        INNER JOIN user_adventure ON user.id = user_adventure.uid
        INNER JOIN adventures ON user_adventure.aid = adventures.id
        WHERE adventures.description =  '[description]';
```

UPDATE QUERIES

Update User's Location by User Id

```
UPDATE user
    SET zip_code = [new_zip_code]
    WHERE id = [userID];
```

Update User's Last Name by User Id

```
UPDATE user
    SET l_name = [new_last_name]
    WHERE id = [userID];
```

DELETE QUERIES

Delete Tribe from User by Tribe Name and User Id

```
DELETE FROM Tribes
    WHERE tribe.uid = [userID]
    AND tribes, tribe_name = [tribe_name];
```

Delete Item from List by List Id and Item Description

```
DELETE FROM item
    WHERE item.description = [itemDescription] AND item.lid = [list_ID];
```