Jamie Hlusko, 996289449

I worked alone and I used 2 grace days on this assignment, so I have 1 left.

**CSC343 Assignment 3**

**Part A - Assumptions:**

Users can edit their profile, so if they change their city of residence, there might be a city in the database with no users residing there.

Each event is organized by a single member.

Each post is by a single author.

Posts can have multiple replies.

Replies can have replies.

People can make each of their couches available independently.

A single couch request can cover a range of start/end times.

A single event has a single duration (so you can’t have one event like “movie screenings”, you would need separate events for each screening).

Also durations include time, so they can be used for events.

If the same content by the same author is posted to multiple groups it still constitutes multiple postings.

Each time someone stays with someone they can make a new evaluation.

The credibility score doesn’t need to be stored, the application can construct it as needed.

**Part B - ER Diagram**

## 

## Part C

User (UserID, Real Name, Interests, Music, Movies, Books, Email, Gender, Birth Date, Has Pets, Allows Pets)

City (CityID, Name, Country)

Search (SearchID, UserID, Host Gender, Country, City, Start Date, End Date, NotifierID)

Couch (CouchID, Capacity, Owner)

Duration (DurationID, Start Date, End Date)

Request (RequestID, Description, Number of Surfers, Status, UserID, CouchID)

Event (EventID, Description, Location, DurationID)

Group (GroupID, Description, Type, Category, Organizer)

Post (PostID, Content, Author, GroupID)

Residence (UserID, CityID)

Matches (SearchID, CouchID)

Couch Availability (CouchID, DurationID)

Request Duration (RequestID, DurationID)

Evaluation (EvaluationID, Evaluator, Evaluatee, Testimonial, Rating)

Friendship (Sender, Receiver, Date met, Type, Experience Shared, Visible on Sender’s Friends List)

Invited (UserID, EventID)

Attending (UserID, EventID)

Reply (Parent, Child)

## Part D

The User table contains the users of the couchsurfing website.

CREATE TABLE User (

PRIMARY KEY UserID [INTEGER],

Real Name [VARCHAR(50)],

Interests [VARCHAR(120)],

Music [VARCHAR(120)],

Movies [VARCHAR(120)],

Books [VARCHAR(120)],

Email [VARCHAR(50)],

Gender [ENUM(“F”, “M”)] NOT NULL,

Birth Date [DATE],

Has Pets [BOOLEAN],

Allows Pets [BOOLEAN]);

The City table contains the cities where users have lived.

CREATE TABLE City (

PRIMARY KEY CityID [INTEGER],

Name [VARCHAR(50)],

Country [VARCHAR(50)]);

The Search table contains all the searches users perform.

CREATE TABLE Search (

SearchID [INTEGER],

UserID [INTEGER] REFERENCES User CASCADE,

Host Gender [ENUM(“F”, “M”)] NOT NULL,

Country [VARCHAR(50)]

City [VARCHAR(50)],

Start Date [TIMESTAMP],

End Date [TIMESTAMP],

NotifierID [INTEGER],

PRIMARY KEY (SearchID, UserID);

The Couch table contains the capacities of all couches users own.

CREATE TABLE Couch (

PRIMARY KEY CouchID [INTEGER],

Capacity [INTEGER],

Owner [INTEGER] REFERENCES User);

The Duration table contains the start/end dates of visits and events.

CREATE TABLE Duration (

PRIMARY KEY DurationID [INTEGER],

Start Date [TIMESTAMP],

End Date [TIMESTAMP]);

The Request table contains the information a guest submits to a potential host.

CREATE TABLE Request (

PRIMARY KEY RequestID [INTEGER],

Description [VARCHAR(120)],

Number of Surfers [INTEGER],

Status [ENUM(“accepted”, “pending”, “declined”, “maybe”)] NOT NULL,

UserID [INTEGER] REFERENCES User,

CouchID [INTEGER] REFERENCES Couch);

The Event table contains events created by users.

CREATE TABLE Event (

PRIMARY KEY EventID [INTEGER],

Description [VARCHAR(120)],

Location [VARCHAR(50)],

DurationID [INTEGER] REFERENCES Duration);

The Group table contains information about couchsurfer groups.

CREATE TABLE Group (

PRIMARY KEY GroupID [INTEGER],

Description [VARCHAR(120)],

Type [ENUM(“Public”, “Private”)] NOT NULL,

Category [VARCHAR(20)],

Organizer [INTEGER] REFERENCES User);

The Post table contains information about user’s posts in groups.

CREATE TABLE Post (

PRIMARY KEY PostID [INTEGER],

Content [VARCHAR(120)],

Author [INTEGER] REFERENCES User,

GroupID [INTEGER] REFERENCES Group);

The Residence table shows which cities users live in.

CREATE TABLE Residence (

UserID [INTEGER] REFERENCES User,

CityID [INTEGER] REFERENCES City,

PRIMARY KEY (UserID, CityID);

The Matches table shows the search hits.

CREATE TABLE Matches (

SearchID [INTEGER] REFERENCES Search(SearchID),

CouchID [INTEGER] REFERENCES Couch)

PRIMARY KEY (SearchID, CouchID);

The Couch Availability table shows when couches are available for surfer visits.

CREATE TABLE Couch Availability (

CouchID [INTEGER] REFERENCES Couch,

DurationID [INTEGER] REFERENCES Duration,

PRIMARY KEY (CouchID, DurationID);

The Request Duration table shows when a guest wants to couchsurf.

CREATE TABLE Request Duration (

RequestID [INTEGER] REFERENCES Request,

DurationID [INTEGER] REFERENCES Duration,

PRIMARY KEY (RequestID, DurationID);

The Evaluation table contains the evaluation information a guest can leave about a host.

CREATE TABLE Evaluation (

PRIMARY KEY EvaluationID [INTEGER],

Evaluator [INTEGER] REFERENCES User,

Evaluatee [INTEGER] REFERENCES User,

Testimonial [VARCHAR(120)],

Rating [ENUM(“1”, “2”, “3”, “4”, “5”)] NOT NULL);

The Friendship table contains information about user relationships.

CREATE TABLE Friendship (

Sender [INTEGER] REFERENCES User,

Receiver [INTEGER] REFERENCES User,

Date met [DATE],

Type [VARCHAR(20)],

Experience Shared [VARCHAR(120)],

Visible on Sender’s Friends List [BOOLEAN])

PRIMARY KEY (Sender, Receiver);

The Invited table contains information on who was invited to which event.

CREATE TABLE Invited (

UserID [INTEGER] REFERNCES User,

EventID [INTEGER] REFERENCES Event,

PRIMARY KEY (UserID, EventID);

The Attending table contains information on who attends which events.

CREATE TABLE Attending (

UserID [INTEGER] REFERNCES User,

EventID [INTEGER] REFERENCES Event,

PRIMARY KEY (UserID, EventID);

The Reply table contains information about which posts are replies to which other posts.

CREATE TABLE Reply (

Parent [INTEGER] REFERENCES Post,

Child [INTEGER] REFERENCES Post,

PRIMARY KEY (Parent, Child);