Introduction for Databases – Project1 Proposal

Qianyuan Chen (qc2200) Keyu Lai (kl2844)

General:
Our application, Eventorer, is about event exploring, where users can search for recent events happening in New York. Similar with Stubhub or SeatGeek, there are tons of events from different categories in our website and user can search, explore, compare and find tickets for the event. Besides, user can also review the place, performer and other related information about this event, and help them to make right choices.

Examples:
Entities: user (uid, password, email, birthday), event (name, category, date, description…), ticket (tid, price, seat, seller), venue (name, location, coordinate, …), performer (pid, name, age, specialty)
Relationship&constraints: event has several tickets and each ticket has exactly one event, event happens at a location (exactly one), event has at least one performer, user favorites several tickets/events, etc.

Data source:
We will use public APIs related to tickets information like stubhub or SeatGeek to populate our event and ticket database. Multiple APIs will use for similar events so that we can compare different tickets information between them. Then data from Yelp or Wiki will be used as source for location and performer information. So basically all data in our website will be true and come from different popular related websites.

User story:
1. Search: search in our database about events using name, keywords, location, performer, etc. So the searching will be done from multiple directions in different databases
2. Browse&Compare: tickets for same event from different source, or events happening in same location, etc. Also user can browse events aggregated by location, time, categories and so on
3. Favorite: User can save tickets in their account for later use, and show intent for event to see who will go to this event together
4. Review: more details and reviews for venue, and surrounding information about each venue like restaurant

E-R diagram:

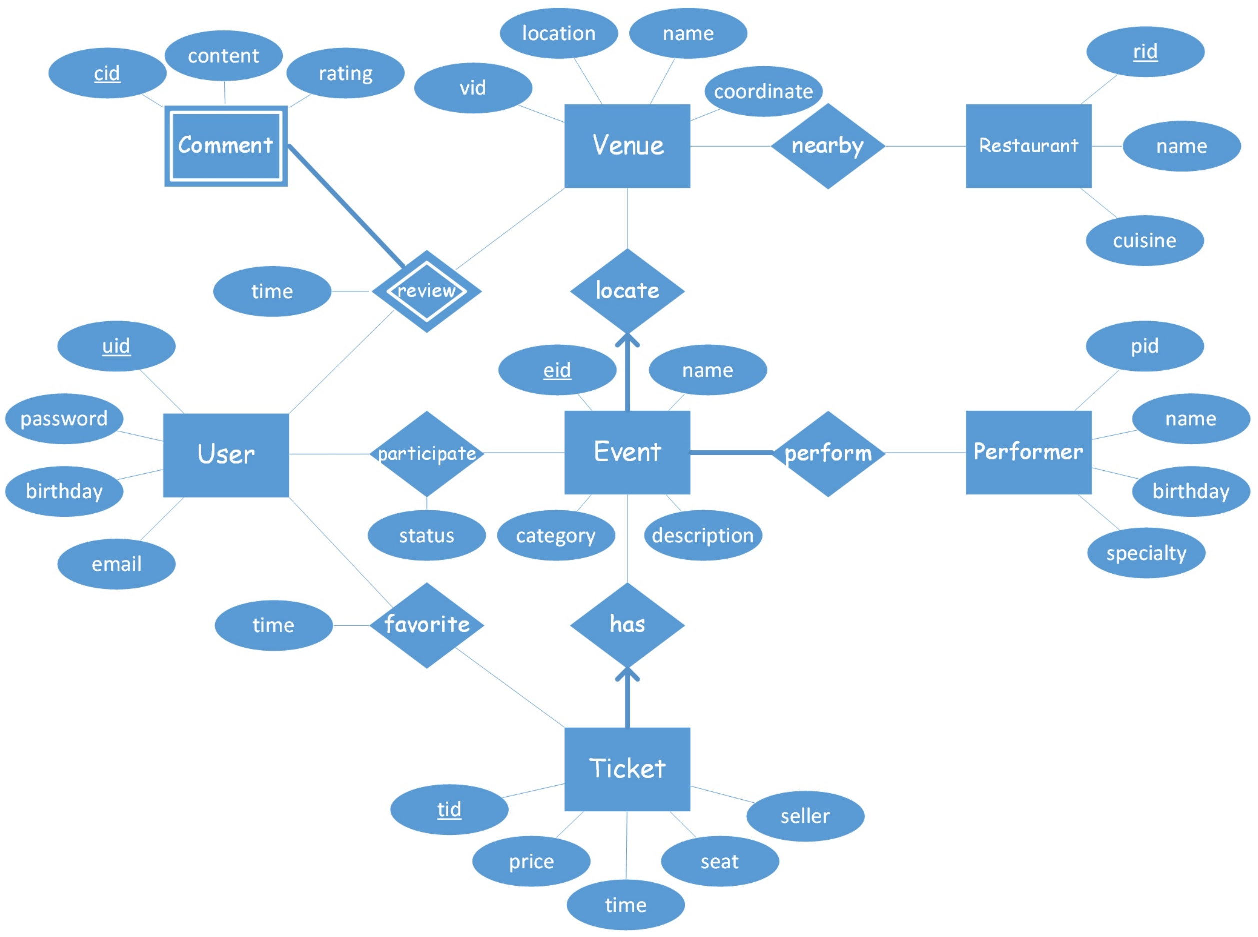

Fig. 1 E-R diagram of Eventorer database

Schema:

```
/* entity */

CREATE TABLE User (
        uid int,
        password varchar(20),
        birthday date,
        email varchar(20),
        PRIMARY KEY (uid)
)

CREATE TABLE Event (
        eid int,
        name varchar(30),
        description text,
        category varchar(20),
        PRIMARY KEY (eid)
)

CREATE TABLE Ticket (
        tid int,
        price real,
        happen_date date,
        seat varchar(20),
        seller varchar(20),
        PRIMARY KEY (tid)
)

CREATE TABLE Performer (
        pid int,
        name varchar(20),
        birthday date,
        specialty varchar(20),
        PRIMARY KEY (pid)
)

CREATE TABLE Venue (
        vid int,
        location varchar(60),
        name varchar(30),
        coordinate point,
        PRIMARY KEY (vid)
)
```

```
CREATE TABLE Restaurant (
        rid int,
        name varchar(30),
        cuisine varchar(30),
        PRIMARY KEY (rid)
)

/* relationship */

CREATE TABLE Participates (
        uid int,
        eid int,
        status int,
        PRIMARY KEY (uid, eid),
        FOREIGN KEY (uid) REFERENCES User,
        FOREIGN KEY (eid) REFERENCES Event
)

CREATE TABLE Favors (
        uid int,
        tid int,
        time timestamp,
        PRIMARY KEY (uid, tid),
        FOREIGN KEY (uid) REFERENCES User,
        FOREIGN KEY (tid) REFERENCES Ticket
)

CREATE TABLE Has (
        eid int NOT NULL,
        tid int,
        PRIMARY KEY (tid),
        FOREIGN KEY (eid) REFERENCES Event
                ON DELETE CASCADE
        FOREIGN KEY (tid) REFERENCES Ticket
)

CREATE TABLE Performs (
        pid int,
        eid int,
        PRIMARY KEY (pid, eid),
        FOREIGN KEY (pid) REFERENCES Performer,
        FOREIGN KEY (eid) REFERENCES Event
)
```

```
CREATE TABLE Reviews (
        cid int,
        vid int,
        uid int,
        content text,
        rating int,
        PRIMARY KEY (cid, vid, uid),
        FOREIGN KEY (vid) REFERENCES Venue
                ON DELETE CASCADE,
        FOREIGN KEY (uid) REFERENCES User
                ON DELETE CASCADE
)

CREATE TABLE Locates (
        vid int NOT NULL,
        eid int,
        PRIMARY KEY (eid),
        FOREIGN KEY (vid) REFERENCES Venue
                ON DELETE CASCADE
        FOREIGN KEY (eid) REFERENCES Event
)

CREATE TABLE Nearby (
        vid int,
        rid int,
        distance real,
        PRIMARY KEY (vid, rid),
        FOREIGN KEY (vid) REFERENCES Venue,
        FOREIGN KEY (rid) REFERENCES Restaurant
)
```