Department of Electrical and Computer Engineering
University of Puerto Rico
Mayagüez Campus

CIIC 4060/ICOM 5016 - Backend System for Disaster Site Resources Locator Phase III: Entity Relationship Diagram / Schema report

Héctor Montes
Samuel Diaz

The main goal of the Disaster site resource locator is to connect people that are in need of resources with people that can supply those resources. Here we describe the entities and relations of our entity relation diagram (ERD).

Users:

Each person that registers to the web site will be represented as a user. The user entity is used to store attributes that are common between requesters, suppliers and administrators. The user entity will use the username attribute as its primary key. Therefore, each user must have a unique username. A user can be an administrator, a requester or a supplier but it cannot be both of these at the same time. A user must supply its address information since it will be used to show the location of the supplied items (if the user is a supplier) or show the location of a request (if the user is a requester).

```
users (
        user_name VARCHAR(15) NOT NULL ,
        email VARCHAR(30) NOT NULL,
        password CHAR(40) NOT NULL,
        first_name VARCHAR(10) NOT NULL,
        last_name VARCHAR(10) NOT NULL,
```

```
        dob DATE,
        phone_number VARCHAR(15),
        address INT NOT NULL,
        FOREIGN KEY (address_id) REFERENCES address (address_id)
        ON DELETE CASCADE,
        PRIMARY KEY (user_name)
)
```

Address:

An address entity will specify the location of a user. We are assuming that the location of the requests is the same as the user that is a requester . We are also assuming that the location to the resources is the same as the user that is a supplier The address will contain the country name, city name, street name and district name. All attributes must be specified when creating an Address entity.

```
address (
        address_id INT NOT NULL AUTO_INCREMENT,
        country VARCHAR(25) NOT NULL,
        city VARCHAR(25) NOT NULL,
        street VARCHAR(40) NOT NULL,
        district VARCHAR(20),
        zipcode VARCHAR(9),
        longitude FLOAT(7,4)
        latitude FLOAT(7,4),
        PRIMARY KEY (address_id)
)
```

Administrator:

The administrator entity is used to store the permission level of the administrator. Depending on the administrator level, an administrator can remove unallowed requests or announcements and ban/unban other users from posting. A user and administrator are associated by the relation Administrators. This relation is one to one, a user entity is coupled to only one administrator entity.

```
administrators (
        administrator_id INT NOT NULL AUTO_INCREMENT,
        permission_level INT NOT NULL,
        user_name VARCHAR(15) NOT NULL,
        FOREIGN KEY (user_name) REFERENCES users (user_name)
```

```
        PRIMARY KEY (administrator_id)
)
```

Requester:

The requester entity represents users that are in need of resources. A requester can create a request entity to ask for specific items. A requester can create many requests where each creation stores the time and date the request was made. The requester entity can reserve supplies that have been announced by suppliers. Each request entity has an attribute to store the resquesters balance (buying power). If the balance is zero the requester can only reserve supplies that are up to donation; Otherwise, the requester can reserve any supply if it has sufficient funds. A requester can also make payments for the resources that were reserved.

```
requesters (
        requester_id INT NOT NULL AUTO_INCREMENT,
        balance DECIMAL(15,2),
        user_name VARCHAR(15) NOT NULL,
        FOREIGN KEY (user_name) REFERENCES users (user_name)
        PRIMARY KEY (requester_id)
)
```

Request:

A request represents a resource that is needed. The request entity has attributes that describe the quantity of the item. A requester can make many requests and each one will store the date of creation. Once a requester reserves a supply that matches the type of the request the request quantity will be updated accordingly until it reaches zero, when it will be said that the request is filled.

```
requests (
        request_id INT NOT NULL AUTO_INCREMENT,
        request_quantity INT NOT NULL,
        request_date DATE,
        resource_id INT NOT NULL,
        requester_id INT NOT NULL,
        fOREIGN KEY (resource_id) REFERENCES resources (resource_id)
        ON DELETE CASCADE,
        FOREIGN KEY (requester_id) REFERENCES requesters (requester_id),
        PRIMARY KEY (request_id)
)
```

Payment:

A payment entity represents a transaction made by a requester entity to purchase a reserved resource. A requester can make many payments. A new payment entity has to be created for each reserved resource that the requester intends to purchase. Many payments are processed at the same time if the user buys assorted items. A payment entity is also used to represent donated items.

```
payments (
        payment _id INT NOT NULL AUTO_INCREMENT,
        total_payment DECIMAL(15,2),
        payment_date DATE,
        reservation_id INT NOT NULL,
        requester_id INT NOT NULL,
        FOREIGN KEY (reservation_id) REFERENCES reservations (reservation_id),
        FOREIGN KEY (requester_id) REFERENCES requesters (requester_id),
        PRIMARY KEY (payment_id)
)
```

Supplier:

A supplier entity represents people that are selling or donating resources to those in need. Since a supplier can be a company, the company's name (if supplied) can appear in the information on the resource supplied instead of the user's name. Suppliers can announce the resources that they are willing to sell or donate. A supplier can announce many supplies, and each announcement will have its date recorded.

```
suppliers (
        supplier_id INT NOT NULL AUTO_INCREMENT,
        company_name VARCHAR(25),
        user_name VARCHAR(15) NOT NULL,
        FOREIGN KEY (user_name) REFERENCES users (user_name),
        PRIMARY KEY (supplier_id)
)
```

Supply:

A supply represents resources that are being sold or donated. The supply entity has attributes such as quantity and price that give information on the properties of the resource being sold or donated. Many supplies can be reserved by a requester.

```
supplies (
```

```
        supply_id INT NOT NULL AUTO_INCREMENT,
        supply_quantity INT,
        supply_date DATE,
        price DECIMAL(15,2),
        resource _id INT NOT NULL,
        supplier_id INT NOT NULL,
        FOREIGN KEY (resource_id) REFERENCES resources (resource_id)
        ON DELETE CASCADE,
        FOREIGN KEY (supplier_id) REFERENCES suppliers (supplier_id)
        PRIMARY KEY (supply_id)
)
```

Resource:

A resource represents any item that is being requested or supplied in the website. The resource entity has attributes that describe the item being described. When making a request of a supply the user must provide the resources name, description and type. The attribute type will help to categorize the item, since one can name and describe the same item in different ways.

```
resources (
        resource_id INT NOT NULL AUTO_INCREMENT,
        resource_type VARCHAR(20) NOT NULL,
        resource_name VARCHAR(20) NOT NULL,
        resource_description VARCHAR(30),
        sold BOOLEAN,
        PRIMARY KEY (resource_id)
)
```

Payment Information:

The way a requester can add funds to their balance is by using their credit cards. Payment Information is a table that represents the information of a user's credit card. It has attributes that hold the cards information such as card number, expiration date and CVV. A requester can have various cards that could be used to add funds to their balance. Each payment information is linked to a requester using a requester_id as foreign key

.

```
payment_information (
        payment_information_id INT NOT NULL AUTO_INCREMENT,
        owner_name VARCHAR(50) NOT NULL,
        card_number VARCHAR(19) NOT NULL,
        expiration_date DATE NOT NULL,
```

```
        cvv VARCHAR(4) NOT NULL,
        requester_id INT NOT NULL,
        FOREIGN KEY (requester_id) REFERENCES requesters (requester_id)
        PRIMARY KEY (payment_information_id)
    )
```

Purchases:

The purchases table will store information related to the purchase made by a requester. It has attributes that store the date, and it links a supplied resource and a requester by using supply_id and requester_id as foreign keys.

```
    purchases (
        purchase_id INT NOT NULL AUTO_INCREMENT,
        purchase_date DATE NOT NULL,
        supply_id INT NOT NULL,
        requester_id INT NOT NULL,
        FOREIGN KEY (supply_id) REFERENCES supplies (supply_id),
        FOREIGN KEY (requester_id) REFERENCES requesters (requester_d)
        PRIMARY KEY (purchase_id)
    )
```

Reservations:

The Reservations table will store information related to the reservation of a resource made by a requester. It has attributes that store the date when the reservation was made. It links a supplied resource and a requester by using supply_id and requester_id as foreign keys.

```
    reservations (
        reservation_id INT NOT NULL AUTO_INCREMENT,
        reservation_date DATE NOT NULL,
        supply_id INT NOT NULL,
        requester_id INT NOT NULL,
        FOREIGN KEY (supply_id) REFERENCES supplies (supply_id),
        FOREIGN KEY (requester_id) REFERENCES requesters (requester_id),
        PRIMARY KEY (reservation_id)
    )
```