

Ideas:

The relationship between user and parking table is many-to-many. (i.e. At different times, a user may park its car at many parking lots, and a parking lot may provide service to many users.) It is difficult to directly store and retrieve data in this case.

However, we can decouple this many-to-many relationship into two one-two-many relationships by using the history table. The role of the history table is to handle the interactions between users and parking lots by logging the parking activities. In this way, each piece of data in the history table can match exactly one user and one parking lot, which makes it more convenient to store and retrieve data.

To implement this, the main ideas are: (1) Let user table only store the information describing users. (2) Let parking table only store the information describing parking lots. (3) The history table is used to log parking activities and serves as a bridge between user and parking table.

Updates:

Remove user\_name, start\_time, end\_time, stay\_period, number\_plate from parking table.

Add u\_id to user table. (u\_id means user id and serves as a primary key uniquely identifying each user.)

Add p\_id to parking table. (Similar to u\_id)

Add u\_id, p\_id to history table. (Identify each parking activity is performed by which user in which parking lot.)

Add stay\_period to history table.

Current functions supported:

get\_user\_details(user\_name, password): Get details of a user given user\_name and password

get\_parking\_history\_of\_user(u\_id, time\_scope): Get parking history of a user in a specified time\_scope

Create\_user(first\_name, last\_name, user\_name, password, date\_of\_birth, email, address, billing\_address): Create a new user

Create\_parking\_lot(postcode, street\_name): create a new parking lot

Post\_parking(u\_id, p\_id, start\_time, end\_time, number\_plate): post a new parking activity to the history table

Delete\_history\_of\_a\_user(u\_id, time\_scope): delete parking history of a user in a specified time scope

Packages required:

mysql-connector-python

Set up:

1. Download and install mysql8.1
2. Set the username and password in the mysql8.1 configurator
3. Enter the password in the mysql command line client
4. Run following command: create database ELEC0138;
5. Run following command: use ELEC0138;
6. Run following command: source path\to\the\file\ELEC0138.sql
7. Firstly run init\_db\_connection(host, user, password, database) in db\_operations.py. host is ‘localhost’ by default. User and password correspond to your username and password of mysql. Database means the name of the database(‘ELEC0138’ )
8. All the other functions could be called after running init\_db\_connection()