AirBnB - Robot

From app concept to open source program

Further training project of

INTERNATIONALE HOCHSCHULE AKADEMIE

https://www.iu-akademie.de

Project by

Björn Leue, *25.11.1985

iu akademie e-mail: bjoern.leue@iu-academy.org

personal e-mail: webmaster@wildsite.de

Last editing: 27.03.23

Introduction – Preview – App Concept – DBMS UML – Explanation – HTML user interface

Contentlist

Introduction

For the further training of the IU-Academy ("Build a datamart in sql") I received the order in the fourth module to create a datamart for the AirBnb example.

Below you will find all the thoughts I've had about it.

Preview

I looked at the task and analyzed the necessary todo.
I decided to include an extended variant with user interface.

Tasks:
Create a DBMS UML with user roles and actions that could be performed by user role
Fill the datatable with dummydata
Create functions for work with that data
Create a web gui

Acceptance criterias:

- Python 3.7
- Installations and execution instructions(ReadMe.md, docstrings,..)
- at least 2-3 join over three tables
- at least 20 entities
- at least 2-3 join over three tables
- recursive relations
- specify all cardinality specifications in for example Chen notation
- short description of your current attributes in a data dictionary

App concept

For the implementation I thought of a Python backend according to the specifications.
This can be controlled by accepting parameters in Key=Value format.

How to call the functions:

Windowskey, then type „cmd“, press Enter and type the following:

for example is „PathToFile“: C:\xampp\htdocs\AirbnbDatamart

```
cd PathToFile
python dateiname.py action=TestEverything
python dateiname.py action=SignupUser user_name=YOURNAME user_password=YOURPASS
python dateiname.py action=StartServer
python dateiname.py action=ShowAll
```

For that i programming a main function, which get the parameter take it to a settings.json file.

From this settings.json will all settted parameter taken for the computing.

Next, the „action“ parameter set the function that have to handled, and data will received.

These are visually evaluated when using the web version and evaluable data via the HTML front end.

Structure

AirBnBDatamart.py	Main program, sets Attributes to json and start an action
actions.py	collection of functions for running the program
sqlite.py	collection of functions for database connection
test_project.py	collection of functions for testing the whole project

DBMS UML

Users

id (Pimary Key)	int autoincrement
name	string
email	string
phone	string
usertype_id (Foreign Key)	int
password	string

UserType

id (PK)	int autoincrement
name	string

Comments

id (PK)	int autoincrement
user_id (FK)	int
place_id (FK)	int
comment	string
created	datetime

RentablePlaces

id (PK)	int autoincrement
name	string
description	string
address	string
city	string
state	string
zip_code	int
price	float
user_id (FK)	int

Rentals

id	int autoincrement
place_id	int
start_date	datetime
end_date	datetime
rental_price	float
created	datetime

UserEvaluation

id (PK)	int autoincrement
user_id (FK)	int
place_id (FK)	int
rating	int
comment	string
created	datetime

PlaceEvaluation

id (PK)	int autoincrement
user_id (FK)	int
place_id (FK)	int
rating	int
comment	string
created	datetime

Photos

id (PK)	int autoincrement
user_id	int
path	string
created	datetime

Explanation:

- Users table: This table stores the basic information about the users of the platform. It has a foreign key usertype_id that links to the UserType table to indicate the type of user (e.g., guest or host).
- UserType table: This table stores the possible types of users, such as guest or host.
- Comments table: This table stores the comments that users leave about the rentable places they have stayed in. It has foreign keys user_id and place_id that link to the Users and RentablePlaces tables, respectively.
- RentablePlaces table: This table stores the information about the rentable places listed on the platform. It has a foreign key user_id that links to the Users table to indicate the owner of the place.
- Rentals table: This table stores the timespan that a place is rented.
- UserEvaluation table: This table stores the evaluations that users leave about other users (e.g., hosts or guests). It has foreign keys user_id and place_id that link to the Users and RentablePlaces tables, respectively.
- PlaceEvaluation table: This table stores the evaluations that users leave about the rentable places they have stayed in. It has foreign keys user_id and place_id that link to the Users and RentablePlaces tables, respectively.
- Photos table: This table stores the path of uploaded images.

Two joins over three tables:

- To get all the rentable places owned by a user along with their comments, we can join the RentablePlaces table with the Comments table on the id and place_id columns, respectively, and then filter by the user_id of the owner.
- To get all the evaluations that a user has received from other users, we can join the Users table with the UserEvaluation table on the id and user_id columns, respectively, and then filter by the user_id of the evaluated user.

HTML user-interface

The variable names must all be entered correctly for the Python script.
Because I find this a bit uncomfortable, I will create a web frontend where the data can be entered in input fields.
The fields are recorded via Javascript and sent to a Python script that calls the Python backend script with parameters.
From the HTML user interface (for usability of the transfer to the command line) the comma's are replaced with --comma-- and spaces with the math cubic (^{3}) character.
This change in the given text is reset in the Python backend.
Html / Javascript ↔ PHP Script ↔ Python Programm ↔ DBMS (MariaDB or SQLite3)

HTML Pages

index.html	Entry point, redirect to login.html
login.html	Login / signup form, data evaluation by Python
start.html	Basic HTML structure for templates, loaded by selecting the function given in the frontend
Templates	HTML snippets with required input fields to control the Python functions