COSC5360 Database Design

Project Description

Trip Share Database Application

Spring 2019

Project Description

In this project, you will develop ***a trip share database management system*** where authorized members can explore the world by exchanging and sharing travel information with each other. Members can post comments, rate attractions, write reviews, build itinerary, etc. The system also stores business partners' information, such as hotel booking, ticket purchasing, etc. The project is organized in four phase: conceptual design and requirements analysis (Phase I), database design requirements (Phase II), normalization (Phase III), and final report &demo (Phase IV).

The database should keep track of the following information:

- Destinations in the system are stored in a geographical hierarchy of Countries, States, Cities, and interesting places, including tourist attractions, restaurants, and shopping malls, in the cities.

- Each destination has a name, description, rating (1.0 ~10.0), and location(**s**), where the names of the different attractions may be the same and some locations may have more than one geographical positions. For example, within 100-mile of Tyler you may see a city of *Paris* along with a Texas's Eiffel Tower in Paris, Texas.

- In the database, the names for all Countries are unique, however the names may be the same for some States or Cities in different Countries. (see the above example)

- Members in the system can be divided into two classes: regular members and preferred members. The database will store each member's email, username, password, ranking, address, wish-destination(s), visited-destination(s), number of followers, and number of followed people.

- Any member can visit any destination in the database. For each visit, the start date and end date will be stored, and the start date cannot be later in time than end date.

- Any member can write comments about any destination in the database. At the same moment, only one comment is allowed for the same member. Hence, the system will store the posting time, date, rating and the content of each comment. (No id will be created.) Make sure that the system always keeps track of whom posted the comments and when they were posted. Other members in the system can like/dislike or reply to the comments.

- For regular member, once the number of followers reaches 1,000,000, the member will be promoted to the preferred member who has more accesses and permissions in the system. Preferred members can edit the destination description in order to provide more accurate information and to correct some errors for the community. The system will keep track of modified date and all the members who modified the destination. Non-preferred members cannot modify anything.

- Members may upload photos about the visited destinations. For each photo, the database will automatically generate a unique id (alphabet A−Z and digit 0−9) and save the link of the picture.

- Any member can create trip plans which are associated with one or several attractions in the database. The system will store a unique id, name, start date/time, end date/time, duration, the associated members and attraction(s) for each trip. Meantime, the arrival/departure time/date for each attraction will be tracked.

- Any member can create itinerary for incoming trip by adding one of more destinations into it. The system will store itinerary id, itinerary name, purpose (food, shopping, business, family …), budget, rating, start/end dates, public/private, member, all associated destination(s), and start/end dates for each destination. Other members may evaluate the posted itinerary.

- The database also stores a cost price for attraction and restaurants only, not for shopping malls.

- Business Partners are companies that conduct commercial advertisings via the system. For each partner, the database keeps track of name of the business, business type, phone, contact person, rating (by members).

Project Questions

a) Can you think 5 more rules (other than the one explicitly described above) that are likely to be used in the system?
b) Is the ability to model super-class/subclass relationships likely to be important in such environment? Why or why not?
c) Justify using a Relational DBMS like Oracle for this project.

Project Phases

I. Draw an EER to accurately represent this set of requirement. This will be your Conceptual Design. Clearly specify any assumption that you are making. You can use any tools (software) to draw the EER. You don't need describe the value constraints of the attributions in the EER diagram. (20%)

II. Use a relational DBMS to implement the database. Perform the following steps. (20%)

a) Convert your Conceptual model to a Logical model that can be implemented in a relational DBMS like Oracle. During this process you replace M-N relationships and multi-valued attributes with constructs that can be implemented in the relational DBMS. Draw EER for the logical model after your modifications. Feel free to change your conceptual model (first delivery) if needed.
b) Convert the EER to a database design. Document your design in Database Schema format like the one we discussed in the class.

III. Use appropriate naming conventions for all of your tables and attributes. (45%)

a) Normalize all of your tables to third normal form. Make any necessary changes to the EER. Explain why these changes needed to be made.
b) Draw a dependency diagram for each table.
c) Write SQL statements to create database, tables and all other structures. Primary keys and foreign keys must be defined appropriately. The quantity

constraints of the relation between the entities, which should be described in EER diagram, are not required.

d) Use the Create View statement to create the following views: (TBD)

e) Answer the following Queries. Feel free to use any of the views that you created in part (d). (TBD)

I. Document the final term project report and demo. (15%)

a) Problem description (Copy it from Web Site).

b) Project questions (Answer questions listed in this project).

c) EER diagram with all assumptions.

d) Relation schema after normalization. All relations must be in 3NF. The relation schema should include primary keys as well as foreign keys (if any) for all relations.

e) All requested SQL statements.

f) Dependency diagram.

g) Demo.