CCT395, Fall 2010, Topics in CCIT: Databases

Assignment 2: Database Design

Due on October 20, 2010, at the beginning of class

This assignment is worth 150 points, 15% of your course grade. The assignment should be done *individually*. You can *discuss* the problems with your fellow students. If you do so, please specify on your assignment with whom you discussed it. (E.g., "Discussed with Obiwan Kenobi and Darth Vader.") Discussing problems, however, should not be interpreted as doing them together. The final diagrams, answers and tables must be your own.

1. Database Design

You've been invited to join a small but well-funded startup company that is going to take on Facebook and other social networking sites. To succeed, you will of course need to offer features that Facebook and Twitter do not offer. As the first step, however, you need to figure out how to build a simple system that would offer some of Facebook's core features. Of course, this system will include a database.

You decided that the first iteration of the system will need to support the following features:

1. People should be able to create accounts on the site. They will need to provide an email address while registering.
2. Registered users should be able to provide information about themselves in their accounts, including their name, age, and gender.
3. Registered users should be able to find other members (e.g., by name or email address) and to designate some of them as their "friends." Users designated as "friends" would need to confirm that they are in fact friends.
4. Registered users will be able to post short "status" messages on the website.
5. Registered users will be able to post photos.
6. When posting status messages and photos, users should be able to specify whether those should be visible to all users, just their friends, or friends and friends-of-friends. They should be able to later change who gets to see their status messages and photos. People who become "friends" or a particular user should be able to see all messages and photos currently set to be shared with "friends," including those that have been shared before they became friends.
7. Users should be able to see status messages and photos that they are allowed to see, browsing them either by time or by user.
8. Users should be able to post comments on status messages and photos that they are allowed to see.
9. Users should be able to *delete* their photos, status messages and comments.

1.1. A Basic ER Diagram and a Data Dictionary (40 points)

Draw an ER diagram for this database. At this point, use many-to-many relationships where appropriate and do *not* use associative entities. Use the simplified diagram style that we used in class, with either crow's foot or UML notation for relationships. Label all relationships. Attach a concise explanation for any design choices that may require an explanation. If some choices depend on information not provided in the outline above, then you

should explain your assumptions. (You assumptions should not directly contradict the stated requirements.)

In addition to the diagram, describe each entity in the form of a "data dictionary": for each entity, provide a short (50-100 words) description of what it represents and fill in a table like this one (one per entity) that lists the attributes, their domains, and their meaning. For example:

Course

Describes a course that has been approved and may be offered in different sessions. The course represents the information that is associated with all instances of this course (in different sessions), rather than with a specific offering of the course.

Attribute	**Domain**	**Meaning**
course_id	integer	an arbitrary ID to be used as an identifier
code	text string, up to 10 characters	the official course code
description	a long text string, up to 300 characters	the official catalog description of the course
type	either "Y" or "H"	"H" means the course is a half-year course, "Y" means it is a full year course

1.2. An ER Diagram with Associative Entities (10 points)

Modify your ER diagram to get rid of all many-to-many relationships, replacing them with associative entities. Explain any decision that may require an explanation.

2. Madame Z's Fortune-Telling Center

Madame Z runs a fortune-telling business, employing a number of fortune tellers. The business has many customers, some of whom come back on a regular basis. Before a visit customers can browser the list of fortune-tellers by name or method used (tarot, astrology, etc.) and can schedule a session with a fortune-teller. Repeated customers are usually assigned to one person as their "primary" fortune-teller. When customers come in for a session, a number of predictions can be made, which are saved for future reference. The customer is billed for the services provided during the session, which can include multiple billable items. They can either receive an invoice after the completion of the session to be paid immediately, or have an invoice sent to them by mail at the end of the month, in which case the invoice would include all services for that month. Customers who receive an invoice by mail can pay it by mailing a check. The payments are recorded when they are received.

Madame Z wants to start using a database for tracking the relevant information, so she hired a consultant who produced the ER shown on the next page.

2.1. Analyzing the Diagram (20 points)

Using the diagram, please answer the following questions (5 points per question):

a) Looking at a particular prediction, is it possible to know who made it, what client it was for, and what method of fortune-telling was used for it? Explain.

b) Is it possible to identify to which session a particular payment applies? Explain.

c) Invoices are issued at the end of each month. When preparing an invoice for a particular customer, how would the system determine the amount to bill? Would it be possible to send customer an invoice specifying to what session each item corresponds?

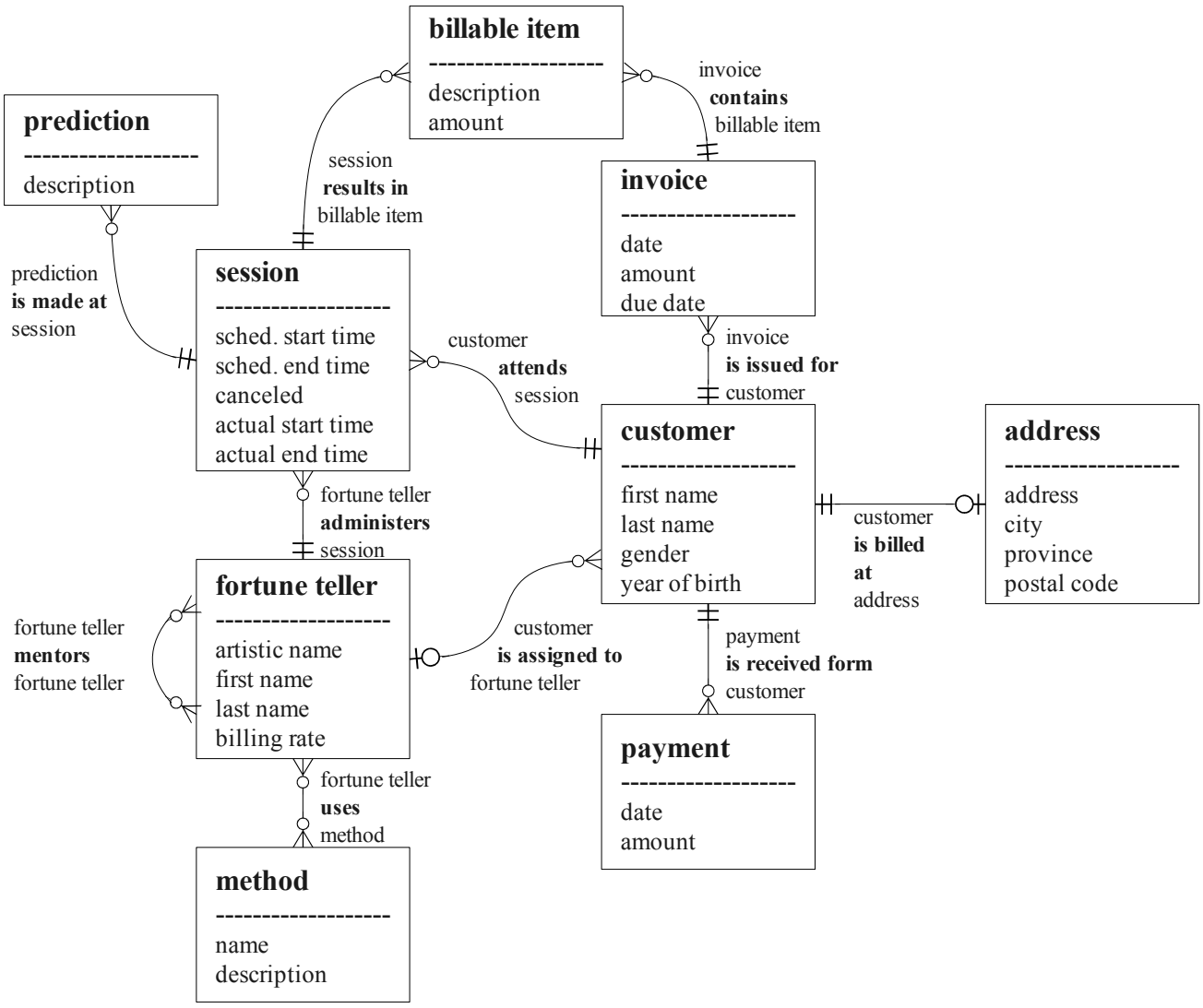

Figure 1. An ER diagram for Madame Z's Fortune-Telling Center.

d) Does the diagram contain any relations that would need to be broken-up with associative entities? If so, which ones? Explain.

Do not use more than 100 words per question.

2.2. Converting the ER Diagram to a Relational Representation (20 points)

Convert the diagram into relational form. Use the notation from Harrington's RDD, chapter 6. E.g.:

Customer (customer_number, customer_first_name, customer_last_name, customer_gender, customer_year_of_birth, customer_primary_teller)

Identify primary keys. Briefly summarize how you arrived at your representation and explain any non-obvious choices.

Does your schema satisfy the requirements for the 1NF? 2NF? 3NF? Explain each answer. If your schema fails one of the requirements, change it so that it does satisfy them.

2.3. Building the Database (15 points)

Write SQL to create the tables for Madame Z's database. Test your code using the database server and then include it in your assignment. Explain any non-obvious choices.

2.4. Populating the Database (15 points)

Write SQL to populate the database with the following information:

> Jason Smith, male, born in 1972, is signed up for a session with "Dr Rain", from 2-3 pm on Oct. 29, 2010.
>
> Mary Smith, female, born in 1969, was signed up for a session with Dr Rain from 4-5 pm on Oct. 15, but this session was canceled.
>
> Pete Brown, male, has Dr Rain as the primary teller but signed up for a session with Madame Z herself. The session was scheduled to take place from noon to 2 pm on Oct. 6, but Jason showed up late. The session began at 12:30 and went until 2 pm. Pete was charged $500 for the session. He received his invoice immediately upon the completion of the session and paid it in full right away.
>
> Dr Rain's real name is Robert Green. He uses two methods: Tarot cards and astrology. No description is available for Tarot cards, but astrology is described as "Determining the future by the position of celestial bodies." Dr Rain is mentored by Madame Z.

Populate the database you created on the server and *test* it. Include your SQL in the assignment.

3. Normalization

For each of the following tables you should first identify all functional dependencies and candidate keys, then determine whether the requirements for the 1NF, 2NF, and 3NF are satisfied. If the table violates some of the requirements, explain how it does so and show how to fix it. Explain your changes. If a table satisfies the requirements for 1NF, 2NF and 3NF, check if it has any *other* irregularities involving normalization. If so, explain them and show how they can be fixed.

Please note that we are not interested at this point in whether the tables are complete or otherwise appropriate. Our concern is solely with the normalization rules. For functional dependencies use the format from Harrington's RDD (p. 112).

3.1. Student (10 points)

Table "student" - represents a student in a course enrollment database.

Column	Possible Values	Meaning
name	last name, comma, then given names	the student's full name
student_no*	up to 8 digits	the official student number, assigned by the university
utorid	8 characters	the student's UTorID
sessions	a list of session codes	the sessions during which the student was registered

3.2. Pet (10 points)

Table "pet" - represents a pet in a vet's database.

Column	Possible Values	Meaning
pet_no*	an integer	an arbitrary ID of the pet
name	100 characters	pet's name
owner_no	an integer	the ID of the pet's owner
owner	100 characters	owner's name
telephone	10 digits	owner's telephone number

3.3. Sale (10 points)

Table "sale" - represents a sale of property in a realty database. Please note that a single property may be bought and sold several times. Assume that no two properties can share an address.

Column	Possible Values	Meaning
property_id*	an integer	the ID of the property
seller_id*	an integer	the ID of the seller
buyer_id*	an integer	the ID of the buyer
buyer_name	100 characters	buyer's name
seller_name	100 characters	seller's name
price	integer	final price (in dollars) after all negotiations
address	200 characters	the address of the property