**Database Design: Using Postgresql and Web Tools**

**to Implement the Online Workout Management System**

In this case, you’ll design a relational database for a business that offers personal training over the Internet. After your design is completed and correct, you will create database tables and populate them with data. Then you will produce one form with a subform, four queries, and one report. The form will include customer information, including whether a personal trainer has been assigned and which level of training the customer wants. The queries will address the following questions and tasks: Which customers are over 40 years old? Which customers are very fit or moderately fit and enrolled in the Gold fitness package? How many clients does each trainer have? All members must be over 18 years old, so you will delete records for anyone younger than 18. Your report will summarize the monthly fees brought in by each trainer for each client. You will create a customized navigation pane to manage the database easily.

**BACKGROUND**

Your best friend, Sally, is a health and exercise science major at your university. The two of you have decided to start a business that offers personal training to customers via the Internet. Here’s how it works: Customers register for monthly exercise regimes at three different levels: Gold, Silver, or Bronze. The customers pay a monthly fee for licensed personal trainers to send them a series of exercise workouts, along with links to videos and nutrition information. Sally is in charge of hiring the personal trainers and assembling the various videos and other equipment and material. Because you are proficient in database, you will be in charge of creating the database, which is a prototype system. The final version will be migrated to the Web.

Sally’s idea of the business includes recording specific information about the customers. You will not only have to record their information, such as name, address, phone, and e-mail address, you will also have to ask some probing questions to collect data such as birth date, exercise history, fitness level, and medical history. Once the information is recorded, the customer can be assigned a personal trainer and register for a level of training. The Gold level is the most expensive, at $30 per month. That price includes the exercise regime, unlimited e-mails, and customized streaming video from the customer’s personal trainer. The next level, Silver at $20, also includes the exercise regime and e-mailing, but it only provides

to prerecorded videos on the Web. Finally, the third level, Bronze at $18.50, includes only the exercise regime and 10 e-mail consultations. Along with this information, the database also needs to keep track of information about personal trainers hired by Online Workouts.

Because this database will eventually be moved to the Web, Sally thinks it’s a good idea to create a form with a subform as a prototype of how customers might register for this service. The main form would include all customer information and the subform would include the customer’s personal trainer and desired package level (Gold, Silver, or Bronze).

You and Sally have put your heads together and come up with a number of useful queries that should be added to the database. First, it would be good for the personal trainers to be alerted to customers who are over 40 years old. A query could list those clients easily. Another possible question is which customers are very fit or moderately fit in the Gold program. This list could be useful for marketing further products such as a personal nutritionist or exercise clothing. You would also like to keep tabs on how many clients each personal trainer handles. In addition, although you have stated on your Web site that customers must be at least 18 to enroll, some underage people might have signed up. By local law, no one under 18 can be directed remotely by a personal trainer. You can use a query to delete underage customers.

Finally, you and Sally would like to see how much money you are making each month per trainer, along with customer details. This would fit nicely into a monthly report. In addition, you propose creating a custom navigation pane so that users who are unfamiliar with database system can easily use the database.

1. **Building the Database (15 points)**

Use the SQL DDL statements to create the tables required for this project. Please also note that the tables are created in certain order such that by the time when a foreign key needs to be created, the corresponding primary key also need to be created.

* First, determine the tables you need by listing the name of each table and the fields it should contain. Avoid data redundancy. Do not create a field if it could be created by a “calculated field” in a query.
* You’ll need a transaction table for recording which trainer is with which customer and the level of program. Avoid duplicating data.
* Keep in mind that some tables need a compound primary key to uniquely identify a record within a table.
* Insert at least 5 records for personal trainers with names, address, etc.
* Insert at least 20 records for customer with names, address, etc.
* Create at least 10 records for trainers assigned for a certain customer.

**2. Database Implementation (55 points)**

You need to write SQL queries, stored procedures/functions, and triggers to implement this project. The following requirements and functionalities need to be implemented.

1. (10 points)Create a procedure called Over an Age. The output of the query should list customers who are over an certain age input by user, with only the Customer ID, Last Name, First Name, Email Address, and Birth Date. Your output should resemble that shown in Figure 1, although your data will be different.

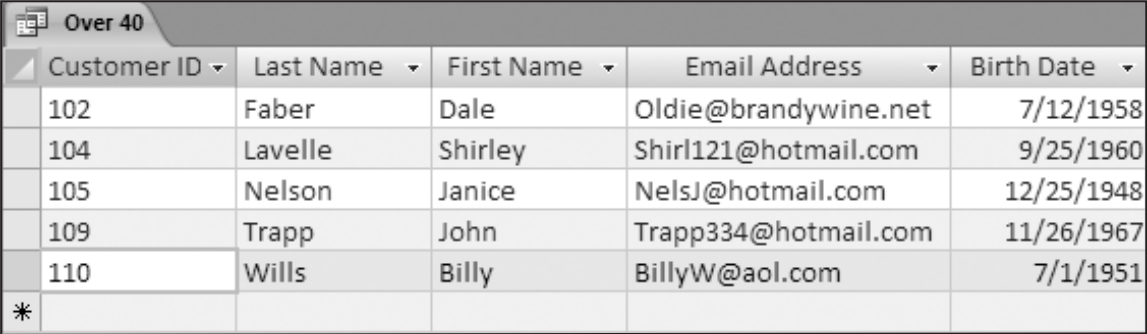


Figure 1: Over an Age

1. (5 points)Create a procedure called Fit and Gold. Show Customer ID, Last Name, First Name, and Email Address. List only customers who are very fit or moderately fit and who have joined the Gold program. Your output should look like that in Figure 2, although your data will be different.

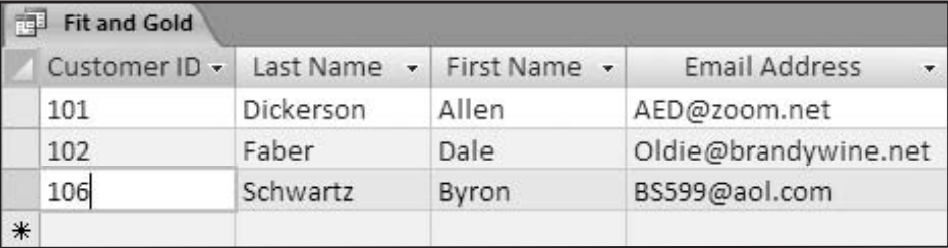
****

Figure 2: Fit and Gold

1. (5 points)Create a procedure called Clients per Trainer. Use the query to add up all the customers for each personal trainer. Display the personal trainer’s (PT) Last Name and First Name along with the Number of Clients. Note the column heading change. Your data will differ but your output should resemble that in Figure 3.

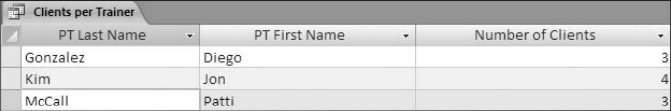


Figure 3: Clients per Trainer

1. (10 points)Create a procedure that deletes any record for any customer under the age of 18. Save the query as Delete Young Members. Use triggers to process the trainers who are related to the customers deleted.
2. (5 points) You need to make your code user friendly by designing and displaying appropriate messages for all exceptions. For example, if someone wants to make an appointment with a trainer, but entered a non-existent name, your program should report the problem clearly.
3. (20 points)Create a procedure called Monthly Money Brought In by Trainer. Your report’s output should show headers for Personal Trainer Last Name and First Name, the customer’s Last Name and First Name, the Level, and Price Per Month. Use the following procedure: 1. First create a query for input to the report. 2. Group the report on the Personal Trainer Last Name. 3. Sum the Price field. 4. Name the report Monthly Money Brought In by Trainer. 5. Adjust the design view to resemble that in Figure 6.

****

Figure 6: Monthly Money Brought In by Trainer.

**3. Interface (20 points)**

Implement a Web interactive interface using any program language. Your interface program should utilize as many of your database stored procedures/functions as possible.

**4. Documentation (10 points)**

Documentation consists of the following aspects:

1. Each procedure and function and every other object you create for your project needs to be explained clearly regarding its objective and usage.
2. Your code needs to be well documented with in-line comments.

**5. Hand-ins, Demo and Grading**

1. You will also need to submit your source code along with your documentation to the Blackboard.
2. It is required to demonstrate your project to the instructor using tuples created by the instructor.
3. The grading will be based on the quality of your code, the documentation and on how successful of your demo is.