Quiz 6. Database: from start to finish.
30 points. The grading will be much stricter. Your database and its forms and reports should be well-designed.

This quiz was posted no later than 12pm on Tue. Aug. 10, and is due no later than 12pm on Thu. Aug. 12. No extension will be allowed. So, you could consider the quiz as a take-home. Nonetheless, I will be in the lab during regular hours (2pm – 3:50pm) on Tue. and Wed., and from 8:30-10:20am (the University’s scheduled time for final exam) on Thu. If no one shows up after 10mns, I will leave the lab.

Note: It would be nice, if you could come to the lab, and on time, on Tue. Thus, you could fill out the Evaluation.

Now, the quiz. You are the owner of a store that sells unique and hard-to-find items. You have kept an inventory of items in MS Excel in one giant worksheet. You have designed this worksheet in such a way that there is one row (i.e., record) for each item, in which all data related to this item is recorded: the class of the item, its sub-class, its title, its description, and the information about the vendor you have purchased it from.

Your business has grown, so you want to migrate to MS Access because it has proven too painful to maintain flat Excel data. Complete the following steps to create the database:

1. Download the file Flat-Database.xls. It is a MS Excel file that stores a tiny portion of your inventory data. We only need a tiny portion for this exercise. Take a look at it.
2. Using pencil and paper, normalize it (i.e., split it into small tables).
   a. First, eliminate redundant data. For example, the following table has multiple records (i.e., redundant data) for each student.

<table>
<thead>
<tr>
<th>Student#</th>
<th>StName</th>
<th>AdvName</th>
<th>AdvRoom</th>
<th>Class#</th>
</tr>
</thead>
<tbody>
<tr>
<td>1022</td>
<td>Sue Brown</td>
<td>Jones</td>
<td>412</td>
<td>101-07</td>
</tr>
<tr>
<td>1022</td>
<td>Sue Brown</td>
<td>Jones</td>
<td>412</td>
<td>143-01</td>
</tr>
<tr>
<td>1022</td>
<td>Sue Brown</td>
<td>Jones</td>
<td>412</td>
<td>159-02</td>
</tr>
<tr>
<td>4123</td>
<td>Jim White</td>
<td>Smith</td>
<td>216</td>
<td>210-01</td>
</tr>
<tr>
<td>4123</td>
<td>Jim White</td>
<td>Smith</td>
<td>216</td>
<td>211-02</td>
</tr>
</tbody>
</table>

So, the following two separate tables (one for students, the other for classes) are created for values that apply to multiple records.

<table>
<thead>
<tr>
<th>Student#</th>
<th>StName</th>
<th>AdvName</th>
<th>AdvRoom</th>
</tr>
</thead>
<tbody>
<tr>
<td>1022</td>
<td>Sue Brown</td>
<td>Jones</td>
<td>412</td>
</tr>
<tr>
<td>4123</td>
<td>Jim White</td>
<td>Smith</td>
<td>216</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Student#</th>
<th>Class#</th>
</tr>
</thead>
<tbody>
<tr>
<td>1022</td>
<td>101-07</td>
</tr>
<tr>
<td>1022</td>
<td>143-01</td>
</tr>
<tr>
<td>1022</td>
<td>159-02</td>
</tr>
<tr>
<td>4123</td>
<td>210-01</td>
</tr>
<tr>
<td>4123</td>
<td>211-02</td>
</tr>
<tr>
<td>4123</td>
<td>214-01</td>
</tr>
</tbody>
</table>
b. Second, eliminate fields that do not relate to the main entity of the table and create for these fields another table. For example, the main entity of the student table above is about students, and her/his adviser’s name and room number have nothing to do with it. So, we get the following two separate tables, one for students, the other for advisors. Notice the creation of the AdvID field. Convince yourself that the creation of this field is necessary.

<table>
<thead>
<tr>
<th>Student#</th>
<th>StName</th>
<th>AdvID</th>
</tr>
</thead>
<tbody>
<tr>
<td>1022</td>
<td>Sue Brown</td>
<td>1</td>
</tr>
<tr>
<td>4123</td>
<td>Jim White</td>
<td>2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>AdvID</th>
<th>AdvName</th>
<th>AdvRoom</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Jones</td>
<td>412</td>
</tr>
<tr>
<td>2</td>
<td>Smith</td>
<td>216</td>
</tr>
</tbody>
</table>

c. Hints: You should come up with exactly four (4) small tables.

3. Based on Step 2 above, draw an entity-relationship diagram showing the entities (i.e., relations, i.e., tables) and the relationships between the entities. Help: See Appendix pages 9 through 11 of the Tutorial book for some help.

4. Based on Steps 2 and 3, complete on paper the initial database design as follows. For each relation (i.e., table), identify all primary and foreign keys. For each field, determine its attributes, such as data type, field size, and validation rules.

5. Now, open MS Access, and create the data structure in Step 4. That is, create the tables with their fields. Be sure to set each field’s properties correctly, select a primary key for each table, and then define the relationships between appropriate tables. Also be sure to resize every column. Use a database name of U-H2F.

Hint: you do not have to create the tables from scratch and enter the data yourself by hand (that would be tedious to do). You already have all the data in Excel, and you already know from Step 4 how each table should look like. So, for each table, create an Excel file that reflects the structure of the table. This should be easy as I assume you know how to copy and paste in Excel. Then, import each Excel file to Access. Use the help wizard to find out how to import. When you are done importing, make sure to set each field’s properties correctly, and the relationships between tables.

6. For each table, create a form that you’ll use to view, add, edit, and delete records in that table. Give the form an appropriate title. This title should go in the Form Header. Now, there should be four forms (assuming that you have come up with exactly four small tables as hinted earlier). Name those forms in a meaningful way. For example, if you have a table named University, then name its form as University Form. Caution: make sure the layout of each form is well-designed and user-friendly. Any form that is hard to view, read, and edit is unacceptable, and points will be taken off.

7. For tables that have a one-to-many relationship, create a form with the main form for that primary table and a subform for the related table. Again, name those forms in a meaningful way. For example, if you have two related tables named University and Students, then name its relationship form as University and Student Form. Caution: same as that of Step 6.

8. Do not forget to use the Input Mask for any field that should be masked. That is, data should be easy to read, and readily identifiable. For example, if the data is a financial amount, place a dollar ($) sign in front of it.
9. Create a select query based on all four tables. (I am going to borrow the
terminology from Flat-Database.xls file to explain to you what you should do
next). Display the Sub-Class, the Class, the Title, the Last Name and the First
Name, and the Price, in that order. Create a calculated field named
ProjectedProfit that displays the results of adding $5 to 9% of the price. Format
the calculated field as standard with two decimal places. Sort the query in
descending order by ProjectedProfit. Use an appropriate Input Mask. Save the
query as Projected Profit Query.

10. Design a custom report that includes at least the following features:
    a. Use the Projected Profit Query query as the basis for the report.
    b. Include in the Page Header section the report title, the current date, the
       page number, and the column headings.
    c. Include in a Group Header section, the sub-class name and the class name,
       using an ascending order sort for both fields.
    d. Include in a Detail section, the title, the first name and last name of the
       person from which the item was purchased, the price, and the
       ProjectedProfit field. Hide duplicate values for the title.
    e. Include in a Group Footer section, an appropriate label and totals by sub-
       class for the price and ProjectedProfit fields.
    f. Include in a Report Footer section, an appropriate label and totals for the
       price and ProjectedProfit fields.
    g. Again, those labels should be meaningful.
    h. Name the report as Projected Profit Report.

11. Create a form in which you add command buttons to open the following objects
    of the database: the tables, the other forms, the queries and the report. Help: use
    the Help Wizard to find out how to create a command button with a wizard. Give
    the button meaningful names. For example, if it opens the ProjectedProfit report,
    the text on the button should read something like Project Profit. Lay out the
    buttons nicely. Resize the buttons as necessary. Add a meaningful title to the
    form. Use the above image as a guide.
    a. Create the form, report and query buttons first before creating the table
       buttons.
b. Once you have created those buttons, now you can create the table buttons. But, create them as if they were “Find Next” of “Record Navigation.” Then, for each button, after you have exited the Command Button Wizard window, right-click it, choose Built Event … It will open the Visual Basic Editor and take you directly to the subroutine associated with the button. Delete the following lines of this subroutine:
   
   Screen.PreviousControl.SetFocus
   DoCmd.FindNext

   Then, insert the following statements instead:
   
   Dim tblName As String
   tblName = "xxxxx"
   DoCmd.OpenTable tblName

   Next, replace xxxx with the actual name of the table you want to open.

c. Save the form as Menu.

12. Once you have created the buttons, play with them first to see if they work. Then, modify the event handling codes associated with them to display messages as follows. Make sure that the form Menu is in design view. Right-click one of the button, then choose Build Event … This will open the Visual Basic Editor. Now, for each subroutine, insert the following statement right after the “On Error GoTo” line:
   
   MsgBox("xxxxx")

   Replace xxxxx with a short description of the form, report, query, or table that is about to show. Make sure you do this for each button.

13. Now play with the buttons again.

14. When you are satisfied, save U-H2F.mdb and upload it to your Q6 directory by noon on Thu. Aug. 12. No extension will be allowed.