Graduate Student: Database Integration

ISDS 558, Prof. Robin Burke
Spring 2002
Due: 3/5/2002

Objective:
Build an application for database management using bound controls and SQL queries.

What to do:
Develop a Visual Basic application that manages a database of loans. The Access 2000 database
“loans.mdb” can be found in the course folder on Doctor (\Doctor\Faculty Files\burke\isds558\hwk2). It
contains three tables: Customers, Loans and Types. Examine this database and note the foreign key
relationship between Customers and Loans: a customer can have any number of loans.
Develop an application that manages this database. It should have five forms:
1. Customer list form. This form has a List control that shows all the customers in the database by name
   and allows adding, deleting or editing a customer entry. If a customer is deleted who has outstanding
   loans (with current balance > 0), the system should pop-up a confirmation dialog to verify that the user
   wants to do this.
2. Customer edit form: This form is displayed when a new user is being added or an existing one
   modified. The “Save” button should update the database with the new data and close the form.
3. Loan list form. The customer form should have a button to bring up the list of the loans owed by a
   particular individual. Generate this list using a parameterized SQL query (pg. 184-186). The results
   should be displayed in a DataGrid control and show all of the fields. None of the information should be
   editable. A delete function should be available for loans with a balance of 0.
4. Loan add form. The loan information form should have an “Add” button to bring up the loan add form.
   This form will look like the form for Homework #1 with fields for principal, rate and term. Use a
   DataCombo control in the form to list the names of the possible loan types. (See the example on pp.
   106-109 of the book.) Saving this information should create a new record in the Loan database for the
   appropriate user.
5. About box. As in the last assignment, be sure to include an about box with your name, the assignment
   number and the due date.

Sketch the interface for each form BEFORE you create it in VB. Use the examples in the book as models.
Freehand is fine (although a ruler may help and graph paper is even better); computer-generated is also OK.

What to turn in:
Before class time, place your project and all associated files (including the database) in a folder titled as in
the following example “RBurke_Hwk2”. (Use your first initial and last name, of course.) Put this folder in
the course Drop Box on Doctor (\Doctor\Assignments\Burke\isds558\HWK2). If you make an error and
want to resubmit, that is OK, but you won’t be able to delete, modify or overwrite any folders you have
already submitted. Submit a second time, adding a letter to the end of the folder name (e.g.
“RBurke_Hwk2A”). I will grade the most recent version submitted (up to class time).
At class time, turn in your user interface sketches with a cover page that includes your name and the
assignment number.

Assessment:
This assignment will be assessed on the completeness of the solution to the problem, on the clarity of the
user interface and on the quality of the coding style. Partial solutions will be given partial credit.

Hints and notes:
• The interface should be consistent: the “Loans” button on the customer list form should be disabled if
  no customer is selected, the “Save” button should be disabled if nothing in the customer record has
  changed, etc.
• The interface should be informative: the loan information form should display the name of the customer for whom it was generated and the customer ID.
• The interface should be visually appealing. Label and fields should be aligned. Controls should be grouped into conventional regions.
• Remember to supply appropriate names to controls, variables and other objects. Use the “Hungarian notation” prefixes suggested in the book: “dbl” for a DataList control, “fra” for a Frame, “cmd” for a Command object.
• The project should contain a DataEnvironment bound to the loans.mdb database. Be careful to make the data environment portable as shown on pages 124-125. Copy your project to different locations to make sure. You will lose points if I can’t copy your project from the Drop Box and run it.
• This is a pretty big assignment and I’m giving you three weeks to do it. Don’t leave everything until the last minute (or even last weekend)!

Extra Credit:
• Use a SQL JOIN in the query that generates the loan list so that loan types are displayed rather than their numeric codes.