Criterion C: Product Development

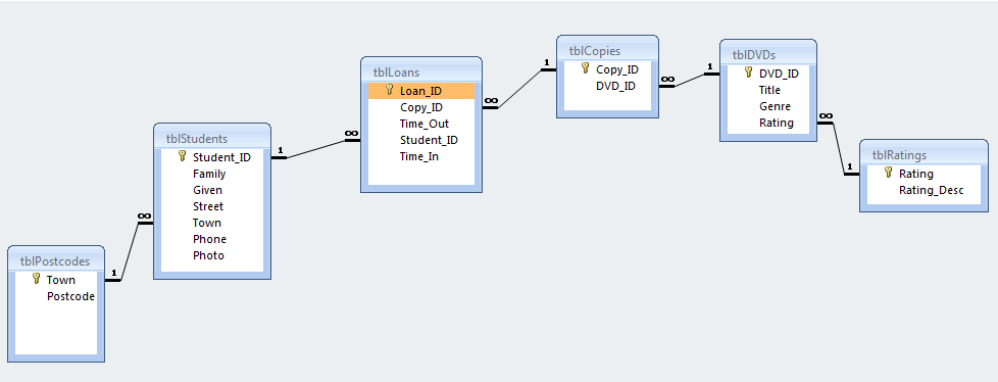
Techniques used to create the database

- Database structure – explanation and justification, 6 related tables including validation techniques normalised to 3NF (pages 1-2)
- Complex queries including calculated fields including concatenation of text, derived fields and the expression builder to generate user friendly output including sub-forms (pages 3-11)
- Macros using the expression builder to allow user input (pages 7-11)
- Other techniques such as Graphics field and use of facilities offered in the software (pages 12-13)

Database structure / algorithmic thinking – explanation and justification

The relational database below consists of 6 linked tables shown below. This has been done to ensure that when data is updated Nicole will not have redundant or inaccurate (where data has been updated in one table, but not in another) data within the database

Table	Keyfield	A record contains...	Additional comments
STUDENTS	Student_ID	Student details	
COPIES	Copy_ID	Copy ID & associated DVD ID	Link table to decompose the many-many relationship between tblLoans and tblDVDs
DVDs	DVD_ID	DVD details	
RATINGS	Rating	Description of rating code	To prevent update anomolies in the tblDVDs
POSTCODES	Town	Town name & its postcode	To prevent update anomolies in the tblStudents
LOANS	Loan_ID	ID of copy & student plus time out and time returned (if applicable)	Provides details of each loan and also acts as a link table to decompose the many-many relationship between tblLoans and tblDVDs



The LOANS table is a link / transaction table linking STUDENTS and COPIES

The COPIES table is necessary as Mme Martin has more than one copy of some DVDs. This table has been created as it is not possible in MS Access to model a many-many table and the relationship has been decomposed into two one-many relationships using a linked table (COPIES).

Litwin, Paul. "FundamentalsOfRelationalDatabaseDesign." *FundamentalsOfRelationalDatabaseDesign*. /www.deeptraining.com, 1994. Web. 13 Apr 2010.

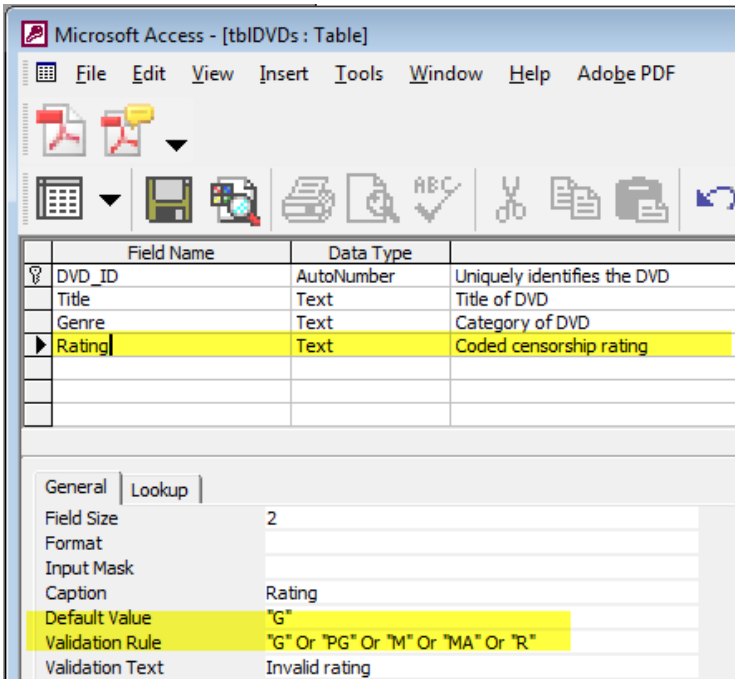
The RATINGS table has been incorporated to eliminate repetition of data which would occur if each DVD record included a rating description.

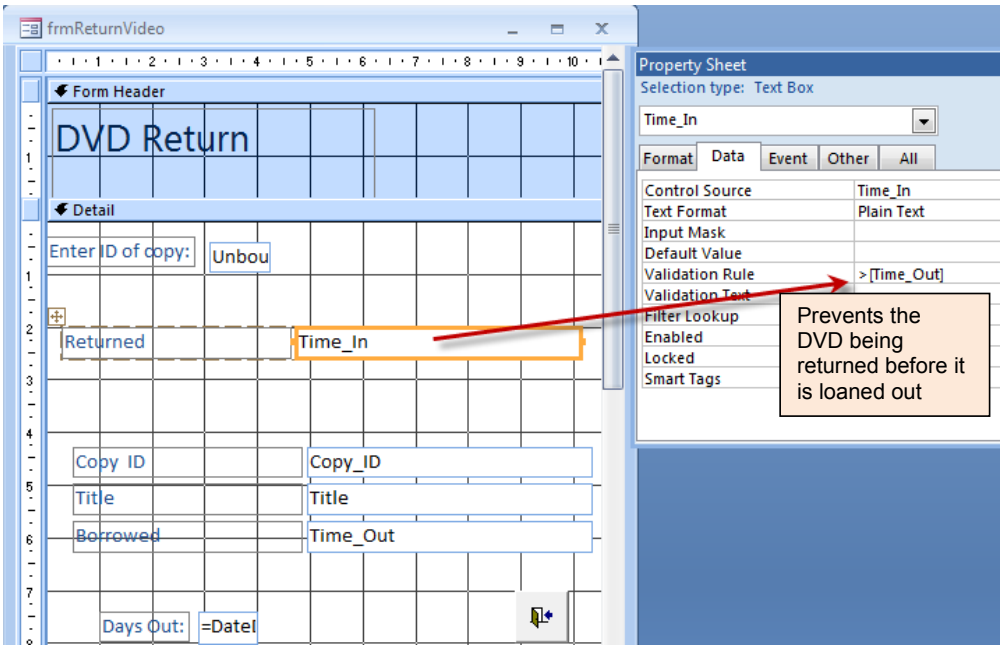
POSTCODES is a look-up table which will save Mme Martin time looking up the postcode each time she enters a new student's address.

Key fields uniquely identify one record in a table and are used for linking tables.

Techniques used to minimise errors during data entry

1. Default values make data entry more efficient and minimise errors eg Time_Out in LOANS defaults to Now() which automatically enters today's date from the computer clock. Rating in DVDs defaults to "G" as most of Mme Martin's DVDs are G rated.
2. Appropriate data types minimise errors eg Time_Out in LOANS is date/time,
3. Input masks limit the field type and number of characters eg Postcode in POSTCODES is 0000 limiting the data entry to 4 numbers.
4. Validation rules limit data entry eg Rating in DVDs (diagram below) is limited to "G" Or "PG" Or "M" Or "MA" Or "R" and if the user enters an unaccepted code the validation text "Invalid rating" provides feedback. Similarly Rating_Desc has a validation rule "General" Or "Parental Guidance" Or "15+over" Or "Mature Audiences" Or "Restricted". The TimeOut cannot be before the TimeIn. This validation rule has been added to the form frmReturnVideo.

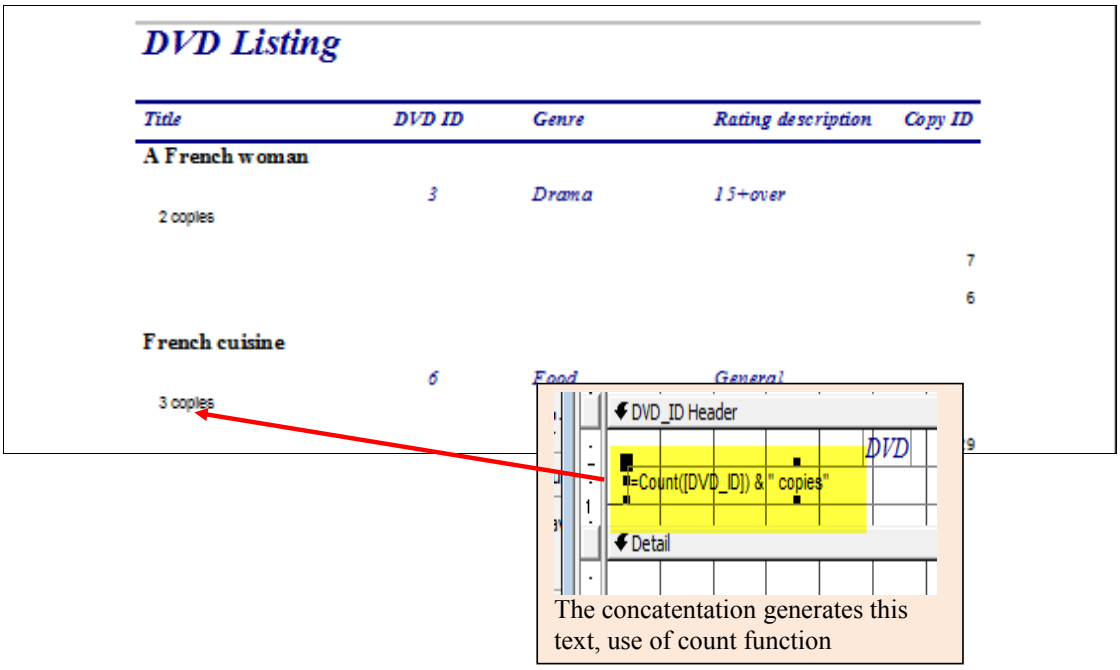



frmReturnVideo
Form Header
DVD Return
Detail
Enter ID of copy:
Unbou
Returned
Time_In
Copy ID
Copy_ID
Title
Title
Borrowed
Time_Out
Days Out:
=Datel
Property Sheet
Selection type: Text Box
Time_In
Format
Data
Event
Other
All
Control Source
Time_In
Text Format
Plain Text
Input Mask
Default Value
Validation Rule
>[Time_Out]
Validation Text
Filter Lookup
Enabled
Locked
Smart Tags
Prevents the DVD being returned before it is loaned out

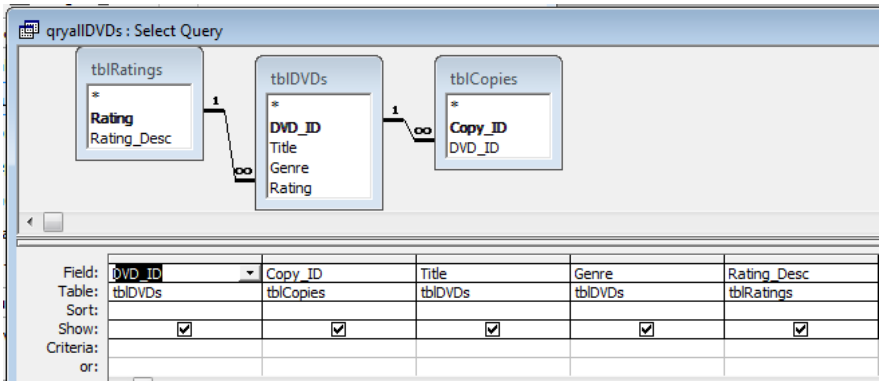
Complex queries / calculated fields including concatenation of text, derived fields and the expression builder to generate user friendly output

1. List of all DVDs using complex queries, derived fields and concatenation

Mme Martin requires a list of all her DVDs. A report has been generated which includes the Copy_ID and totals the number of copies of each video.



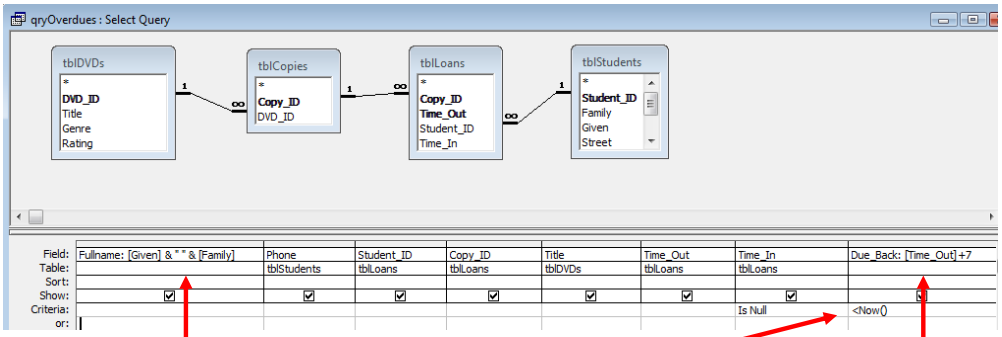
The report is based on the complex query below which uses the links between three tables.



A formula has been added to count the number of copies. Concatenation links number of copies with the word 'copies' so Mme Martin can immediately see the number of copies of each video.

2. List of overdue DVDs using complex queries, derived fields, additional criteria and concatenation

Mme Martin wants a list of overdue videos and needs the name and phone number of the borrowers. By linking tables this query provides details of DVDs and borrowers' names and phone numbers.

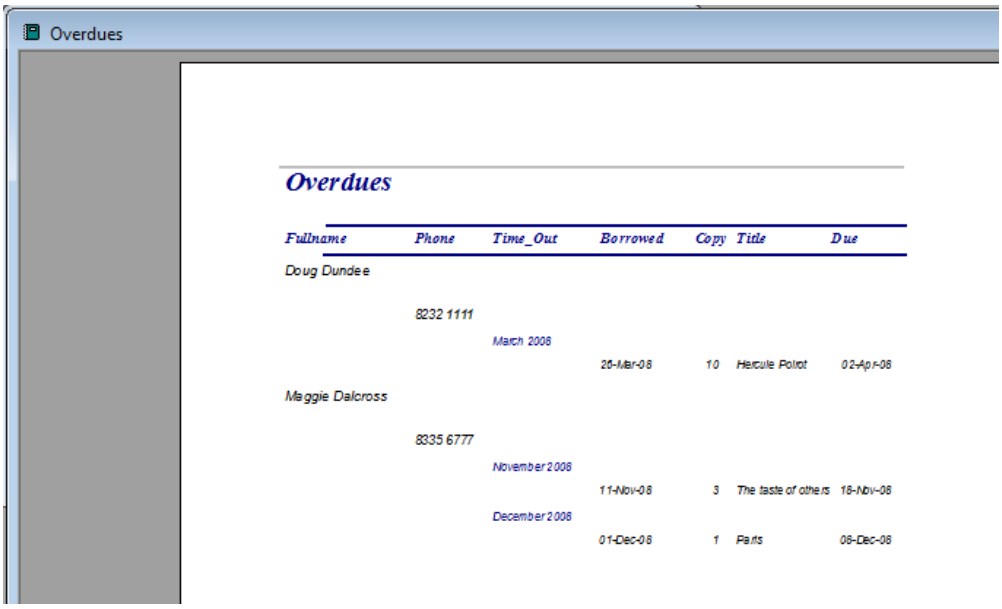


The new field Fullname concatenates Given and Family for easy reading.

The search finds due dates that have passed (ie < today's date) **and** videos not returned ie Time_In is empty.

Since her lessons are weekly she allows a 7 day loan period. The calculated (derived) field Due_Back calculates the due date based on 7 days from Time_Out.

A report (shown below) has been generated based on this query.

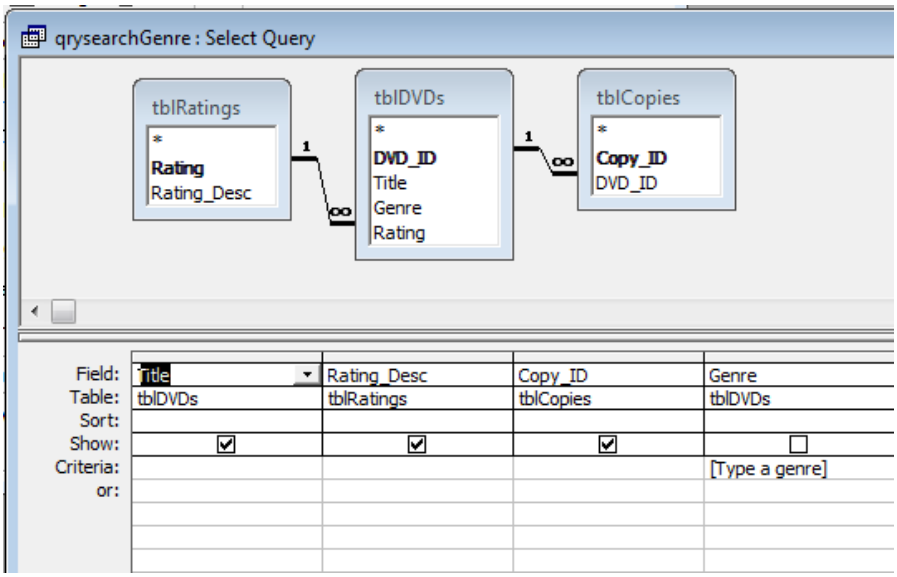


3. Search on a particular genre using complex and parameter queries

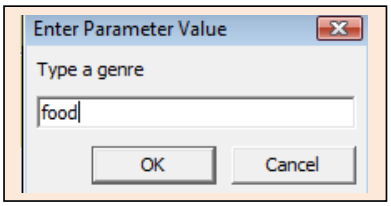
Another requirement is to find DVDs on a particular subject.

This parameter query allows Mme Martin to search on any genre. The allvideosabout report produced provides a list of titles with their ratings and copy ID.

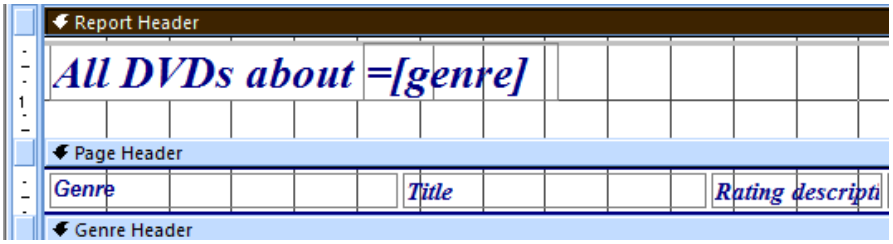
The report below is based on this complex query.



When the report is run the box below appears and this lets Mme Martin type in her chosen genre.



To make the report (allDVDsabout) even more user-friendly in the design of the report a text box has been added with input from the control source [genre]. This displays the heading with Mme Martin's input text.

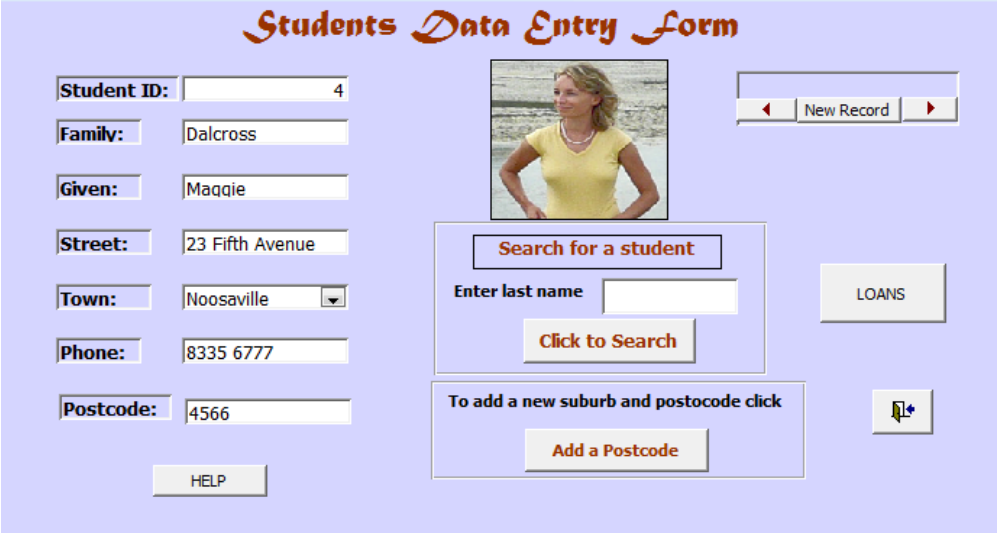


Once the user input is added, the following report is produced.

4. A user-friendly interface – Student details - using complex queries, expression builder and concatenation

Many features have been added to make the database easy for Mme Martin to use

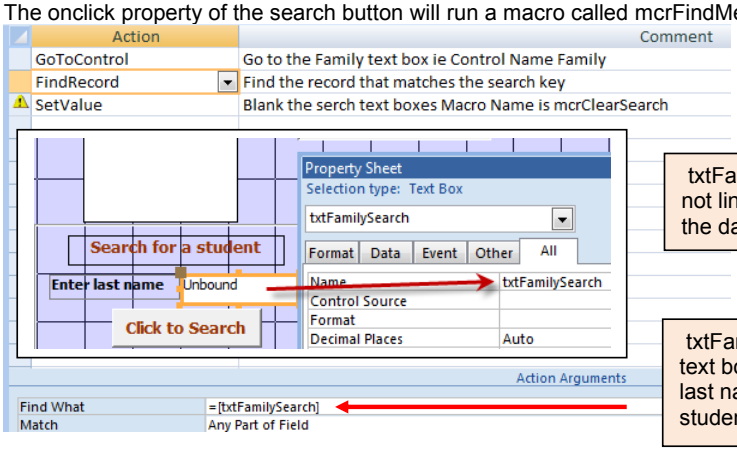
- The student data entry form (frmMemberEdit) includes the student's photo.
- A search button allows Mme Martin to search for a student by typing in last name and uses the expression builder facility to create the parameter query.
- By clicking the LOANS button she can easily see outstanding loans for this student.
- The HELP button provides assistance on using this screen
- The exit button closes the form



The search facility has been developed by creating the macro mcrFindMember

The **search** feature allows easy searching on Last name

The onclick property of the search button will run a macro called mcrFindMember



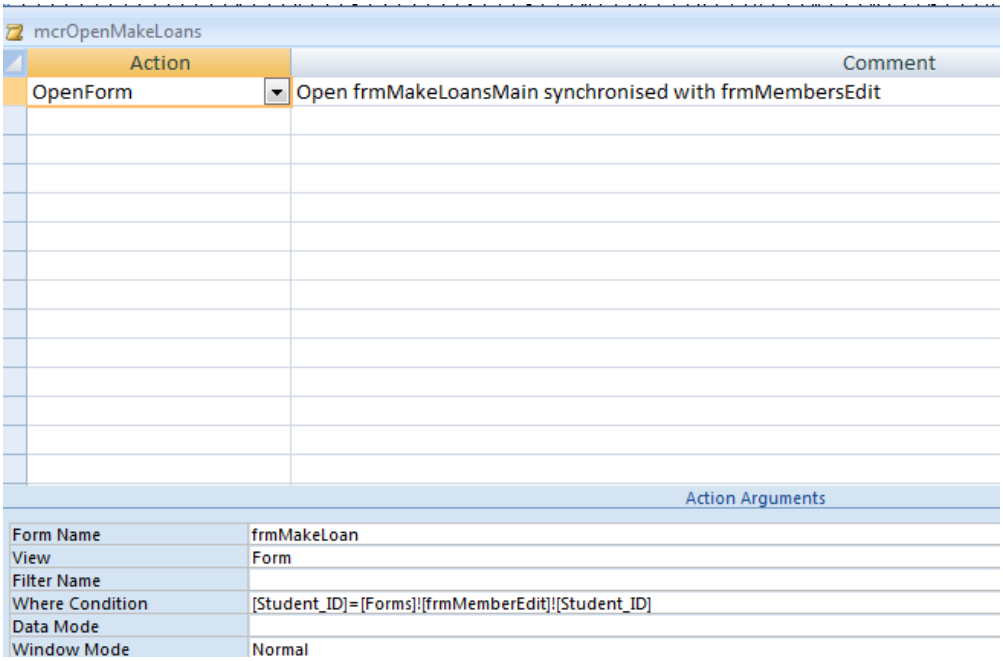
txtFamilySearch is unbound as it is not linked to a field in any table in the database

txtFamilySearch is the name of the text box where the user enters the last name in order to search for a student (see screenshot below)

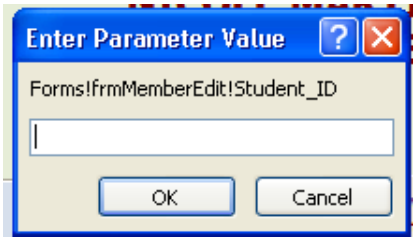
5. A user-friendly interface – Student loans - using complex queries and the expression builder

The LOANS button on frmMemberEdit runs a macro to open the openmakeloans form. This enables Mme Martin to quickly see the DVDs the current student has on loan. In order to locate the relevant student this macro has a condition where [Student_ID]=[Forms]![frmMemberEdit]![Student_ID].

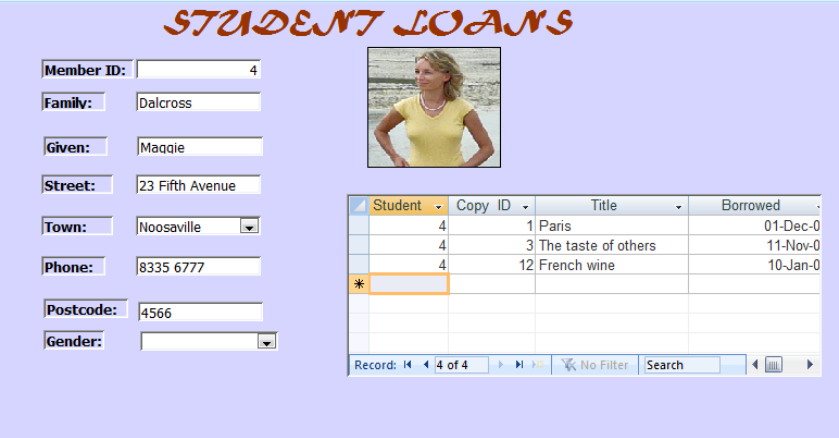
This expression opens the Loans form (frmMakeLoans) for the same student as shown on the data entry screen.



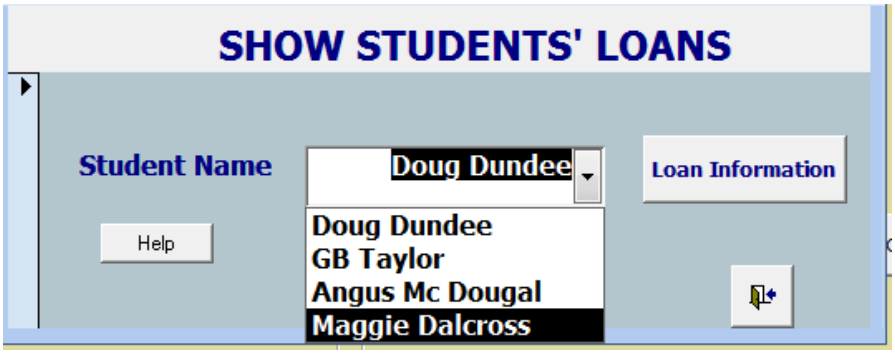
Note:
When the macro is run from the objects list not from frmMemberEdit, the user will be asked to enter a studentID. See below:



The subform is based on a complex query and relationship between the tables ensures that the form and subform are linked on Student_ID to ensure that the loans shown relate to the student.

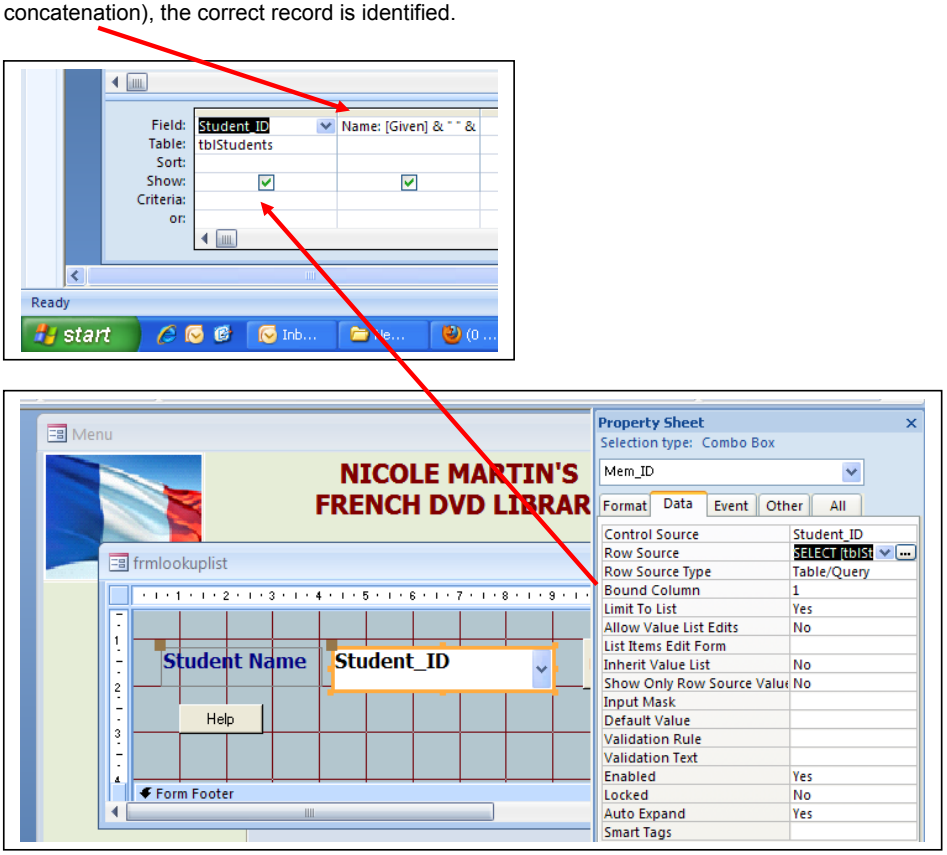


5. A user-friendly interface – Loan details - using complex queries, the expression builder and subforms



The Loan Information button runs a macro similar to the one above which opens the frmLookupList and subform.

The drop down list is generated using the Combo box function and uses the unique StudentID (hidden from the user) as the bound value so that when the full student name is selected (using concatenation), the correct record is identified.



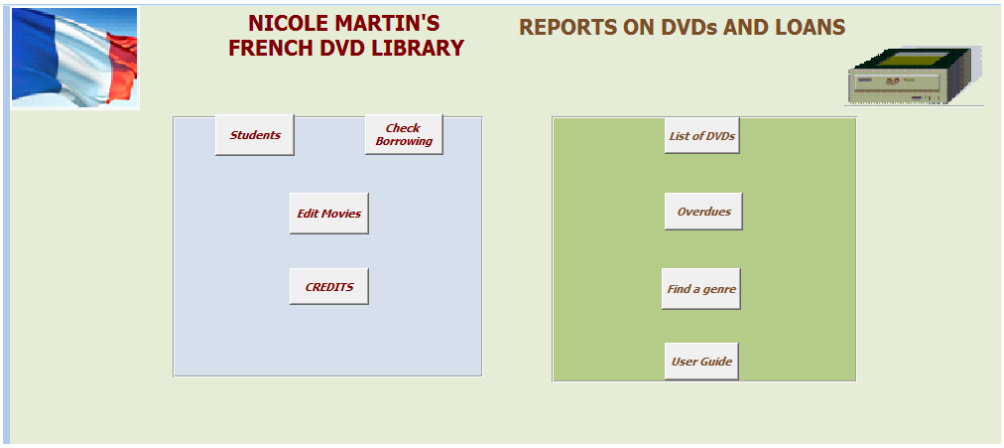
Other techniques used

A user friendly interface – Main Menu

A macro has been created to open the form called Main Menu. By saving this macro as autoexec it automatically launches the Main Menu on startup.

Buttons open forms and reports making the database simple to use.

A user guide is available via a button.

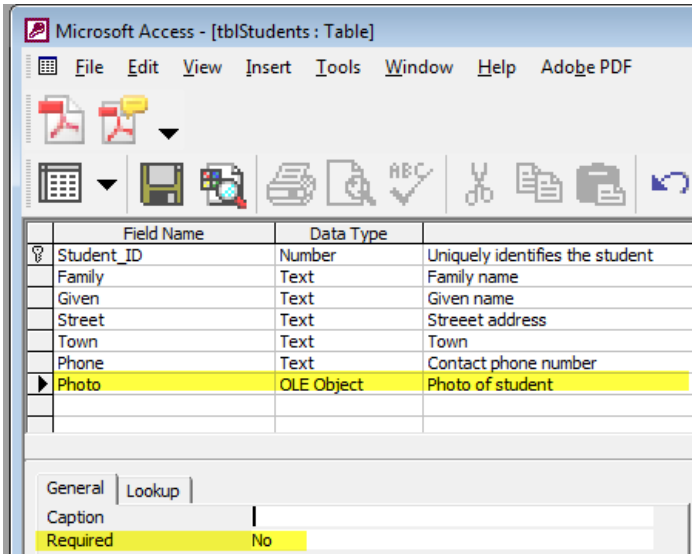


Security and privacy information

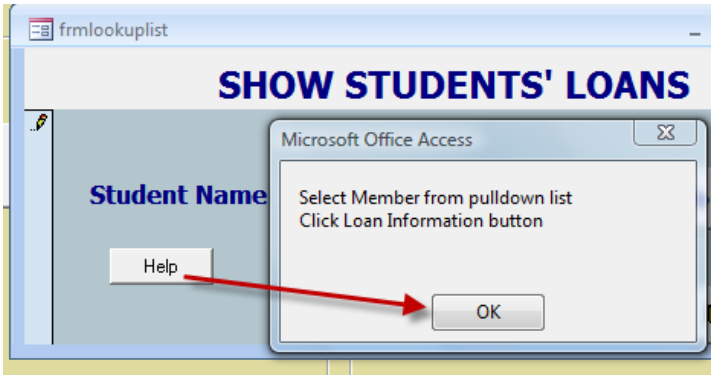
The STUDENTS table contains person information about the students which should not be available to unauthorised users. The database will be loaded onto Mme Martin's home computer. Her computer is not shared with other users and she has a password to log on. A password will also be set on the database for extra security.

The inclusion of images in the database

The photo has been incorporated into the design of the STUDENTS table (below) as an OLE object. Required is set to NO as not all students may provide a photo.



A user-friendly interface – Help facilities
The forms incorporate help buttons which give information about using the forms. Below a macro attached to the Help button has an action to display a message box. The message box incorporates the lines of text.



Word Count approximately 700

This database was based on a video store database in *Developing databases with Access* by Graeme Summers.
His Website is http://graemesummers.info