



More Forms

INFO/CSE 100, Spring 2005
Fluency in Information Technology

<http://www.cs.washington.edu/100>

Readings and References

- Reading
 - » *Fluency with Information Technology*
 - Chapter 15, Case Study in Database Design
- References
 - » MS Access Help files
 - keyword “form”

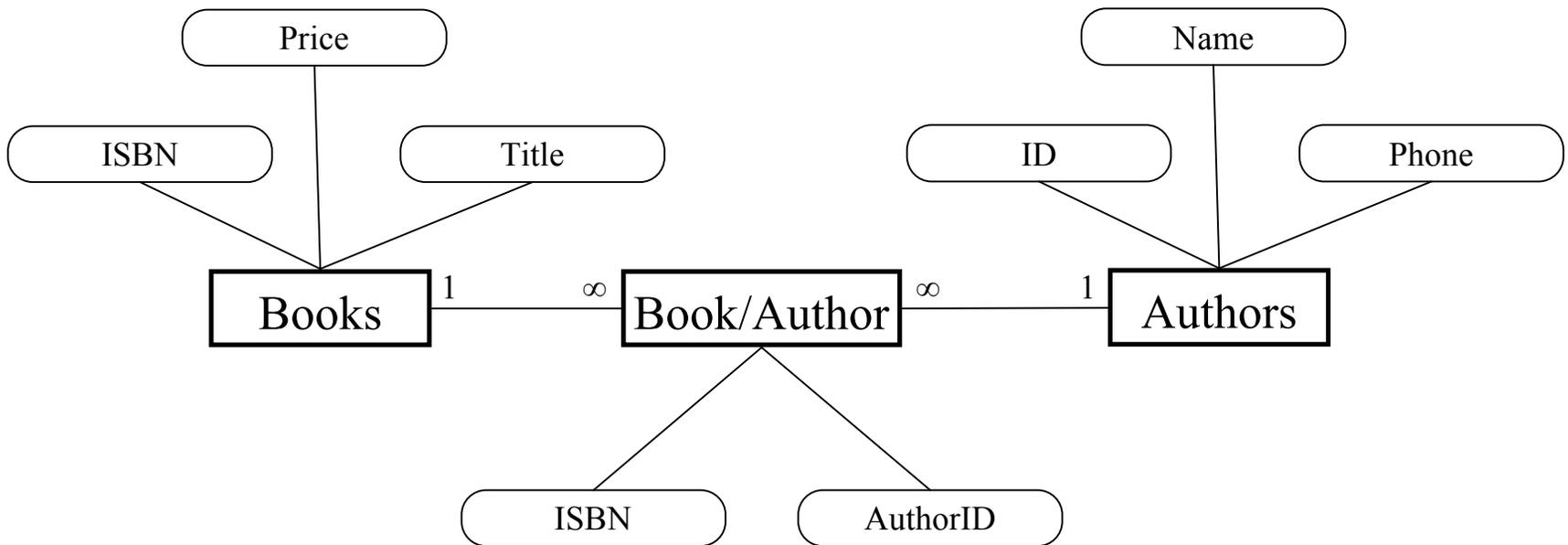
Link one book with many authors?

- We DO want:
 - » to link each book to one or more authors
- We DON'T want
 - » to specify extra fields (author1, author2, author3,...)
 - this is wasteful and limits the max number of authors
 - » to specify each book entry several times, naming a different author in each row
 - this duplicates all the other information about the book

Add a cross-reference table!

- Refine the design so that it includes another table that is a book-author cross reference
 - » Each entity in the table is a single cross reference
 - Attribute: ISBN
 - Attribute: Author ID
 - » No primary key
 - Now we can break the many-to-many relationship into two 1-to-many relationships that we already know how to implement
-

Define new cross-reference entities





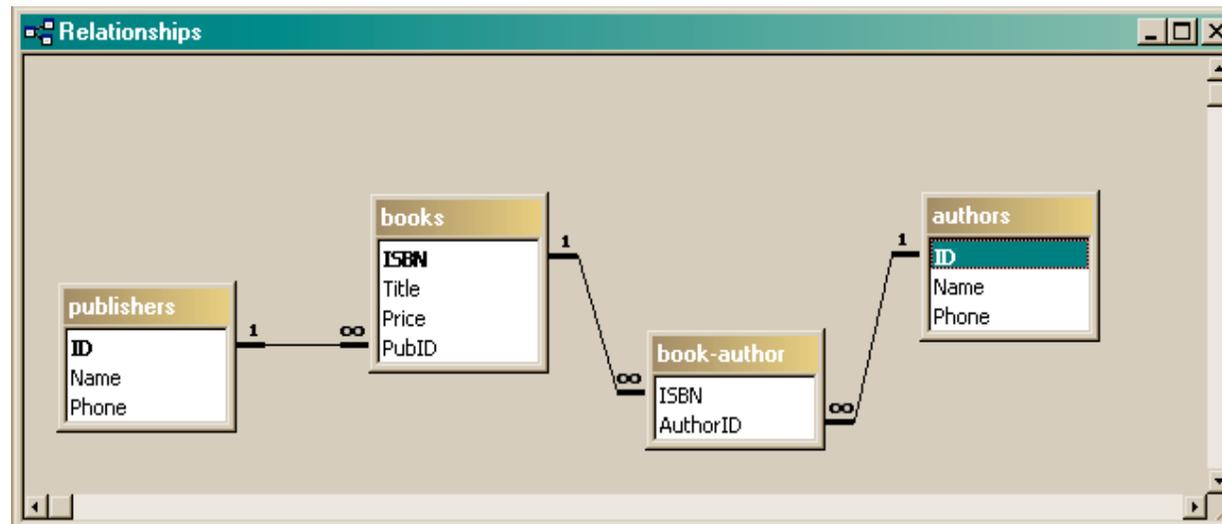
book-author table

The screenshot shows Microsoft Access with three tables defined in a database named 'little : Database [Access 2002 file format]'. The 'authors' table has columns ID, Name, and Phone. The 'books' table has columns ISBN, Title, and Price. The 'book-author' table has columns ISBN and AuthorID. The 'book-author' table is currently open in a data view, showing the following data:

ISBN	AuthorID
1-1	1
1-2	1
2-2	2
2-2	3
*	

At the bottom of the window, there is a status bar with the text 'An author of the book' on the left and 'NUM' on the right.

Define the new relationships



Define a query that uses the relationship

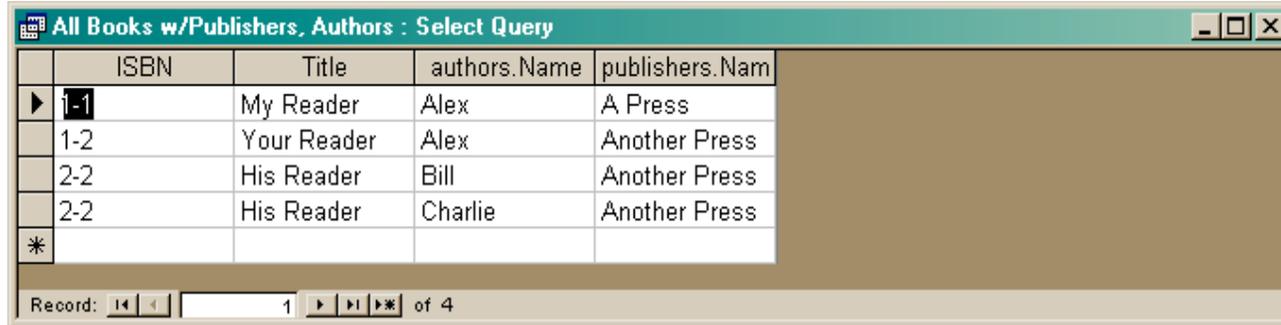
The screenshot shows a database query builder interface. At the top, a window titled "All Books w/Publishers, Authors : Select Query" displays a relationship diagram. The diagram includes four tables: publishers, books, book-author, and authors. Publishers is connected to books (1 to ∞), books is connected to book-author (1 to ∞), and book-author is connected to authors (∞ to 1). The books table has fields ISBN, Title, Price, and PubID. The authors table has fields ID, Name, and Phone. The book-author table has fields ISBN and AuthorID. Below the diagram is a table with columns for Field, Table, Sort, Show, Criteria, and or. The Field column contains ISBN, Title, Name, Name, and an empty cell. The Table column contains books, books, authors, publishers, and an empty cell. The Show column contains checkboxes: checked, checked, checked, checked, and unchecked. Below this is another window titled "All Books w/Publishers, Authors : Select Query" containing the following SQL query:

```
SELECT books.ISBN, books.Title, authors.Name, publishers.Name
FROM publishers INNER JOIN (books INNER JOIN (authors INNER JOIN [book-author] ON authors.ID = [book-author].AuthorID) ON
books.ISBN = [book-author].ISBN) ON publishers.ID = books.PubID;
```

Query By Example

actual SQL

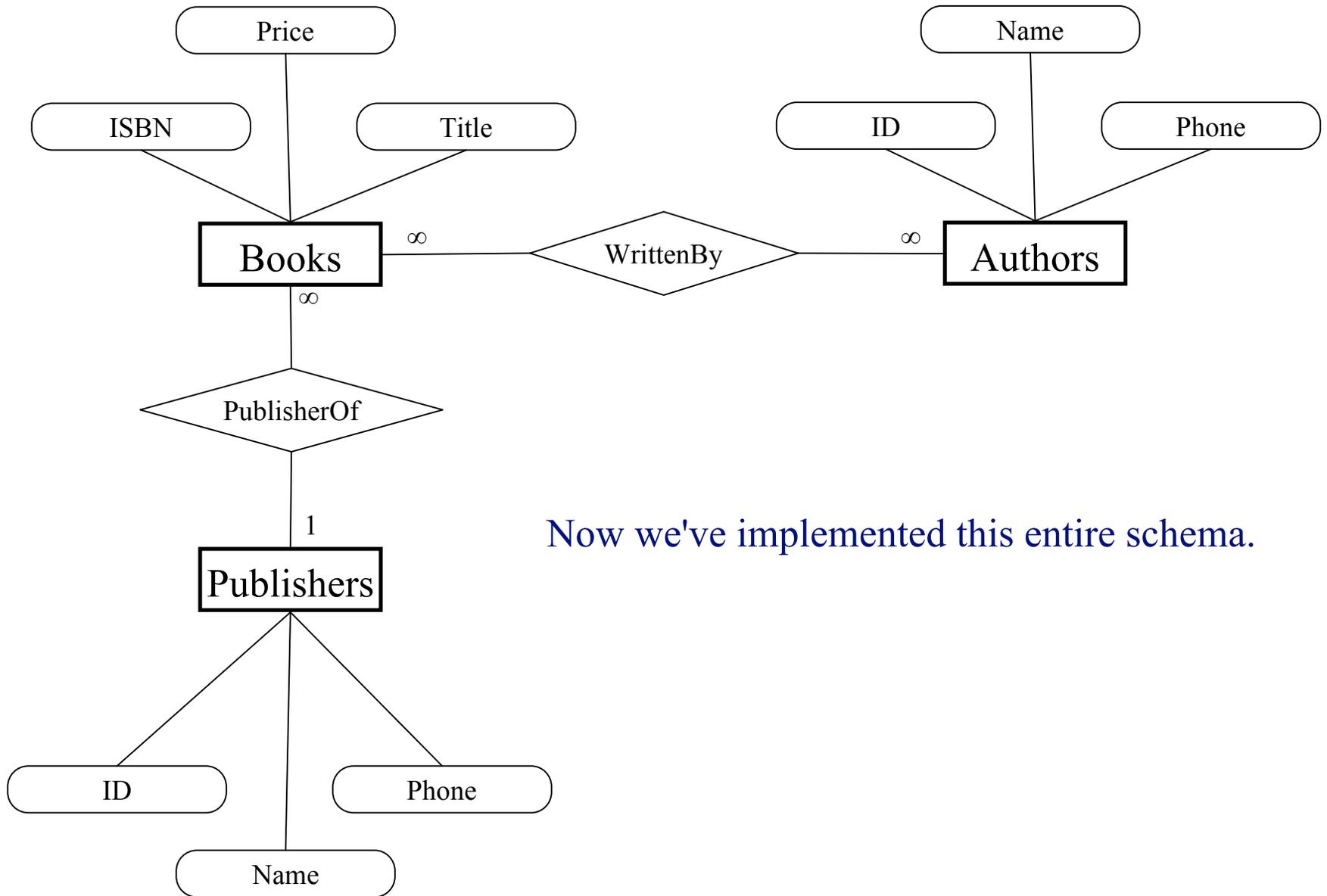
Get the new view of the data



	ISBN	Title	authors.Name	publishers.Nam
▶	1-1	My Reader	Alex	A Press
	1-2	Your Reader	Alex	Another Press
	2-2	His Reader	Bill	Another Press
	2-2	His Reader	Charlie	Another Press
*				

Record: 1 of 4

- Notice that this view has redundant data
 - » That's okay, because we are not storing it this way, just presenting it
 - » The redundant items (Alex, Another Press) came from a single entry in a table – they are guaranteed to be identical

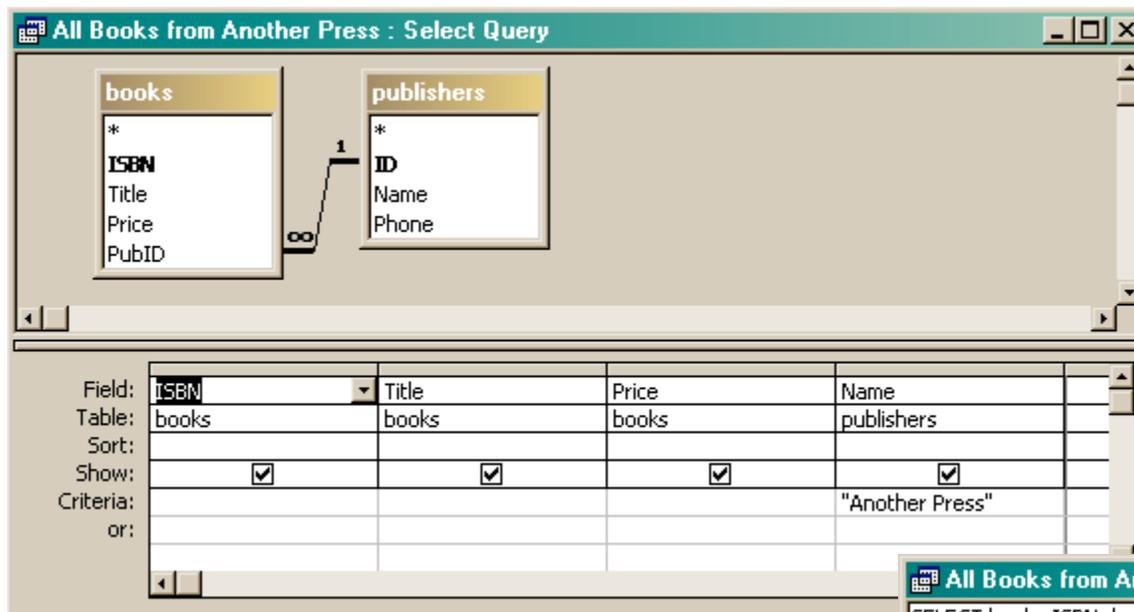


Now we've implemented this entire schema.

View: All Books from “Another Press”

ISBN	Title	Price	Name
1-2	Your Reader	\$12.00	Another Press
2-2	His Reader	\$25.00	Another Press

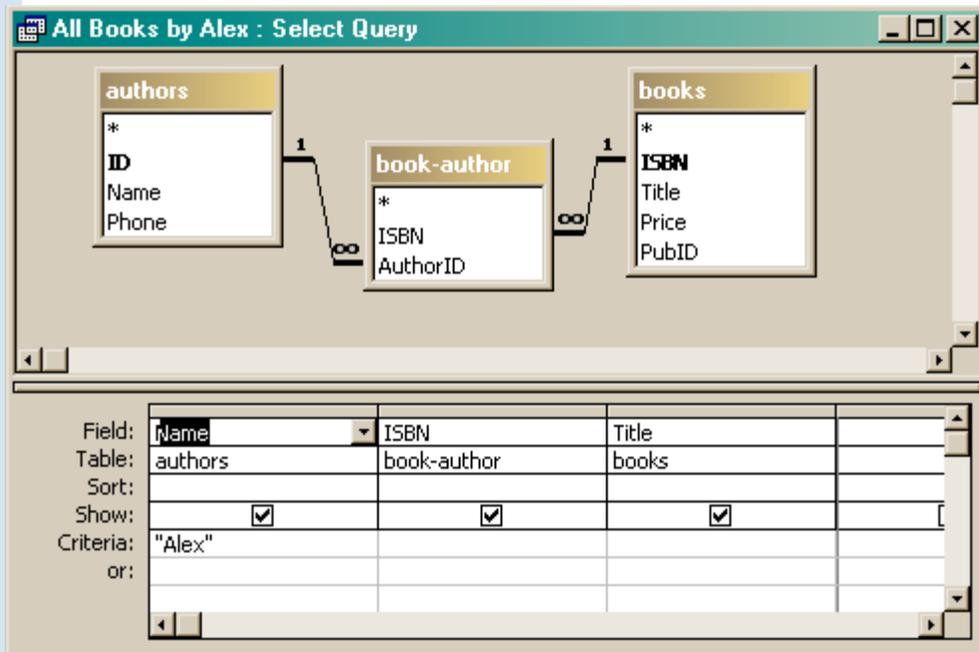
Record: 3 of 3



```
SELECT books.ISBN, books.Title, books.Price, publishers.Name
FROM publishers INNER JOIN books ON publishers.ID=books.PubID
WHERE (((publishers.Name)="Another Press"));
```



View: All Books by Alex



Name	ISBN	Title
Alex	1-1	My Reader
Alex	1-2	Your Reader

Record: 3 of 3

```

SELECT authors.Name, [book-author].ISBN, books.Title
FROM books INNER JOIN (authors INNER JOIN [book-author] ON authors.ID=[book-author].AuthorID) ON books.ISBN=[book-author].ISBN
WHERE (((authors.Name)="Alex"));
    
```

View: All info about a given ISBN

Enter Parameter Value

ISBN?

1-1

OK Cancel

Book Info for Given ISBN : Select Query

ISBN	Title	Price	authors.Name	publishers.Nam
1-1	My Reader	\$10.00	Alex	A Press
*				

Record: 1 of 1

Book Info for Given ISBN : Select Query

publishers: ID, Name, Phone

books: ISBN, Title, Price, PubID

book-author: ISBN, AuthorID

authors: ID, Name, Phone

Field:	ISBN	Title	Price	Name	Name
Table:	books	books	books	authors	publishers
Sort:					
Show:	<input checked="" type="checkbox"/>				
Criteria:	[ISBN?]				
or:					

Book Info for Given ISBN : Select Query

```
SELECT books.ISBN, books.Title, books.Price, authors.Name, publishers.Name
FROM publishers INNER JOIN (books INNER JOIN (authors INNER JOIN [book-author] ON authors.ID = [book-author].AuthorID) ON
books.ISBN = [book-author].ISBN) ON publishers.ID = books.PubID
WHERE (((books.ISBN)=[ISBN? ]));
```

Views as Tables

- Recall that the result of a query is a table
- We have been presenting the table to the user in simple tabular form

All Books from Another Press : Select Query

ISBN	Title	Price	Name
1-2	Your Reader	\$12.00	Another Press
2-2	His Reader	\$25.00	Another Press

Record: 3 of 3

All Books by Alex : Select Query

Name	ISBN	Title
Alex	1-1	My Reader
Alex	1-2	Your Reader

Record: 3 of 3

Book Info for Given ISBN : Select Query

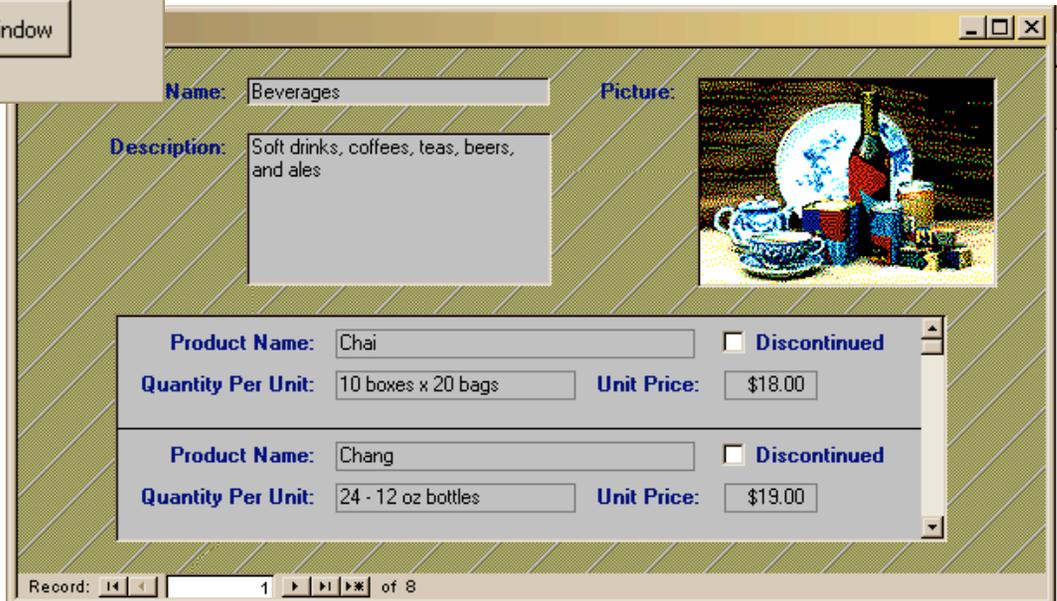
ISBN	Title	Price	authors.Name	publishers.Nam
1-1	My Reader	\$10.00	Alex	A Press

Record: 1 of 1

But tables are not pretty ...



Users need help understanding what they are looking at and what they can do with it

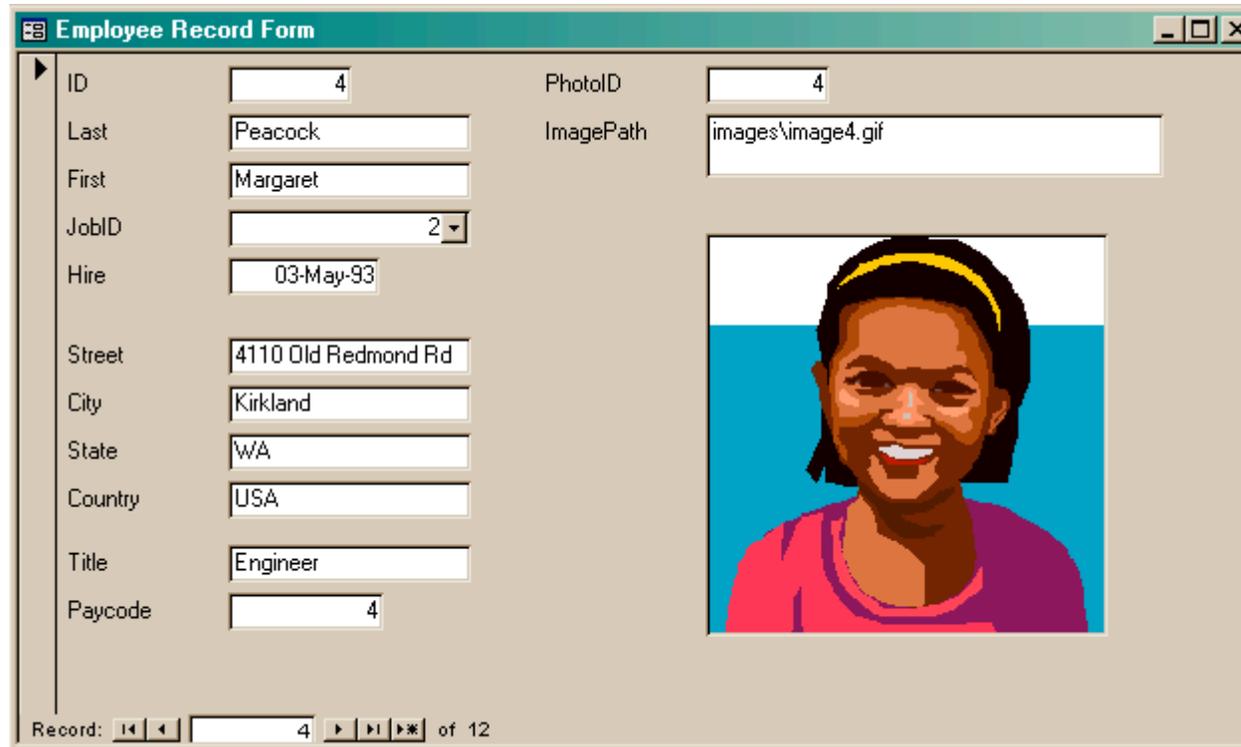


Front end and Back end

- Front end
 - » We present the data to the user with some sort of Graphical User Interface
 - Simple tabular display as we have been doing
 - MS Access provides *Forms* and *Reports* for GUIs
 - Web pages
- Back end
 - » The database stores the data in tables
 - » We use queries to construct new "virtual" tables

Forms

A form is primarily used to enter or display data in a database



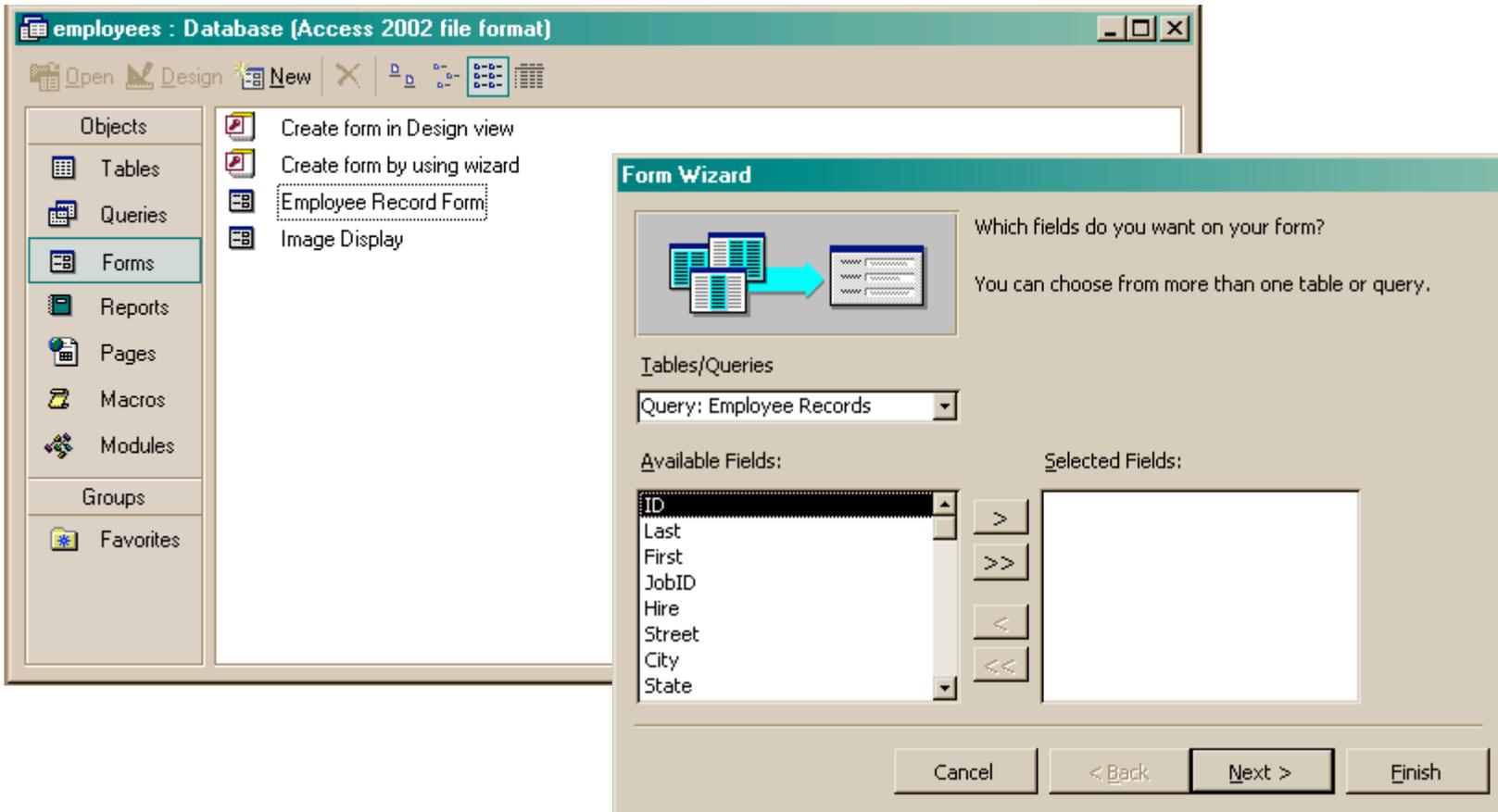
The screenshot shows a window titled "Employee Record Form" with a light beige background. It contains several input fields and a photo. The fields are arranged in two columns. The left column contains: ID (4), Last (Peacock), First (Margaret), JobID (2), Hire (03-May-93), Street (4110 Old Redmond Rd), City (Kirkland), State (WA), Country (USA), Title (Engineer), and Paycode (4). The right column contains: PhotoID (4) and ImagePath (images\image4.gif). Below the ImagePath field is a photo of a woman with dark hair, wearing a yellow headband and a purple top, smiling. At the bottom of the window, there is a record navigation bar with the text "Record: 4 of 12" and navigation icons.

The designer controls what it looks like and how it works,
so it can be tailored to specific needs

A Form is just a Face for a table

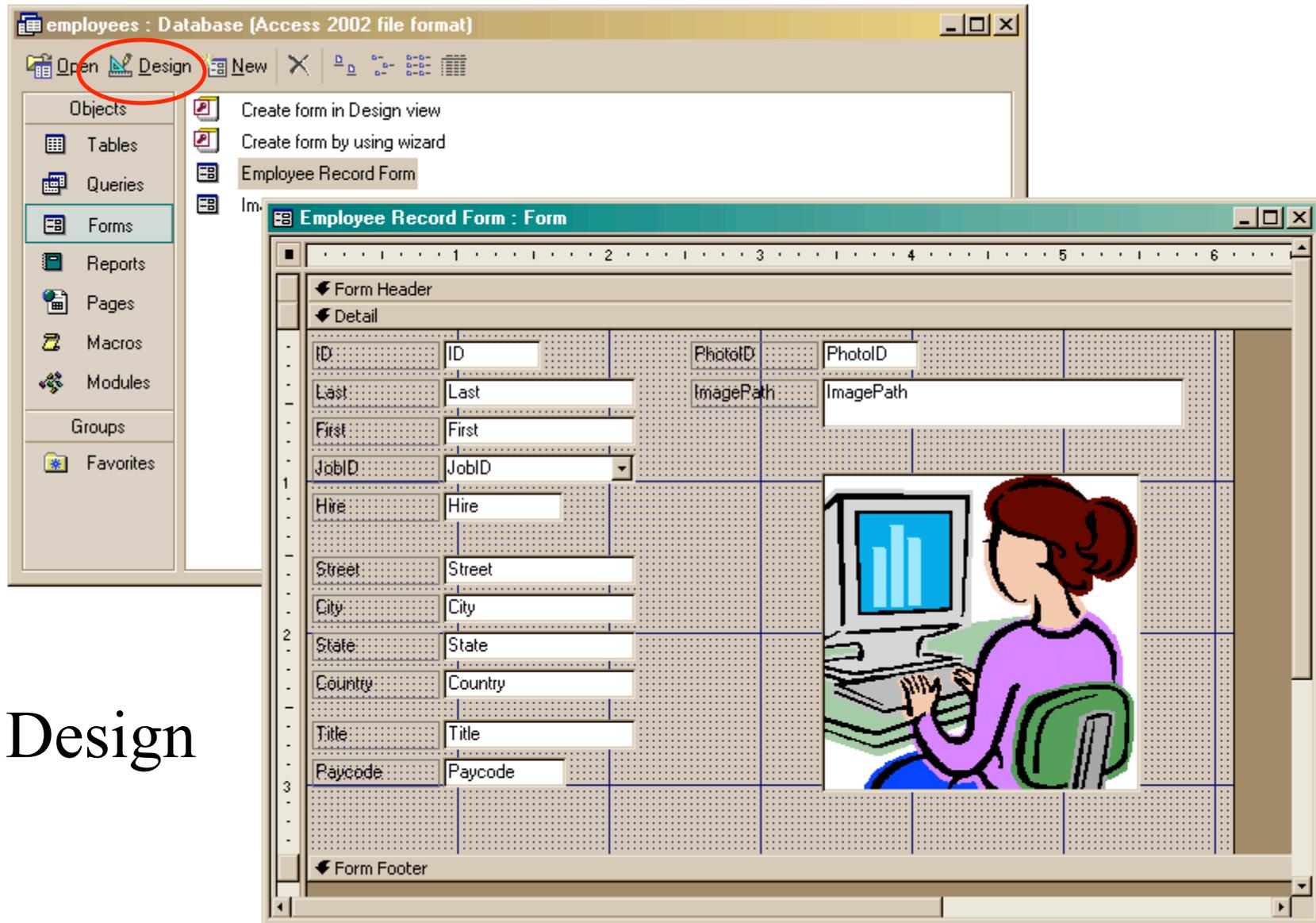
- The form lets the designer arrange the data, label it, provide some control over events, etc
 - » the **presentation**
 - » multiple presentations are possible depending on the specific needs of each user
- Underlying data comes from a table or a query
 - » the **content**
 - » single source of data ensures consistency

How does a form get built?



The Form wizard can help get you started.

But you probably want to tweak it ...



Design

Explore the Design capabilities

- Properties of the various controls can be set
- Controls and labels can be moved around
- Images and patterns can be applied
- Event handlers can be written just like on HTML pages with onClick, etc
 - » these are written in Basic, not JavaScript

Displaying an image

- In general, images are not stored directly in the database
 - » This would mean copying the image and storing it as part of the database file
 - The resulting database is very big
 - The image files are not available outside of the database program
- But we can easily store a link to the image file
 - » a text field containing the path to the image file
 - » use the path to find, load, and display the image

Simple Display Form

Photo Display

ID:

ImagePath:



Record: of 12

ImagePaths table



	ID	ImagePath
+	1	images\image1.gif
+	2	images\image2.gif
+	3	images\image3.gif
+	4	images\image4.gif
+	5	images\image5.gif
+	6	images\image6.gif
+	7	images\image7.gif
+	8	images\image8.gif
+	9	images\image9.gif
+	10	images\image10.gif
+	11	images\image11.gif
+	12	images\image12.gif

Record: of 13

To display a linked image

- Create a form based on a table or query that includes the path attribute
 - » include a text field on the form to hold the path
- Create an image control on the form
 - » this is where the image is actually displayed
- Create event handlers to load the image when something changes
 - » associated with the form event OnCurrent
 - » associated with the text field event AfterUpdate

text field that holds the value of the primary key for the ImagePaths table, the ID attribute

text field that holds the value of the attribute ImagePath

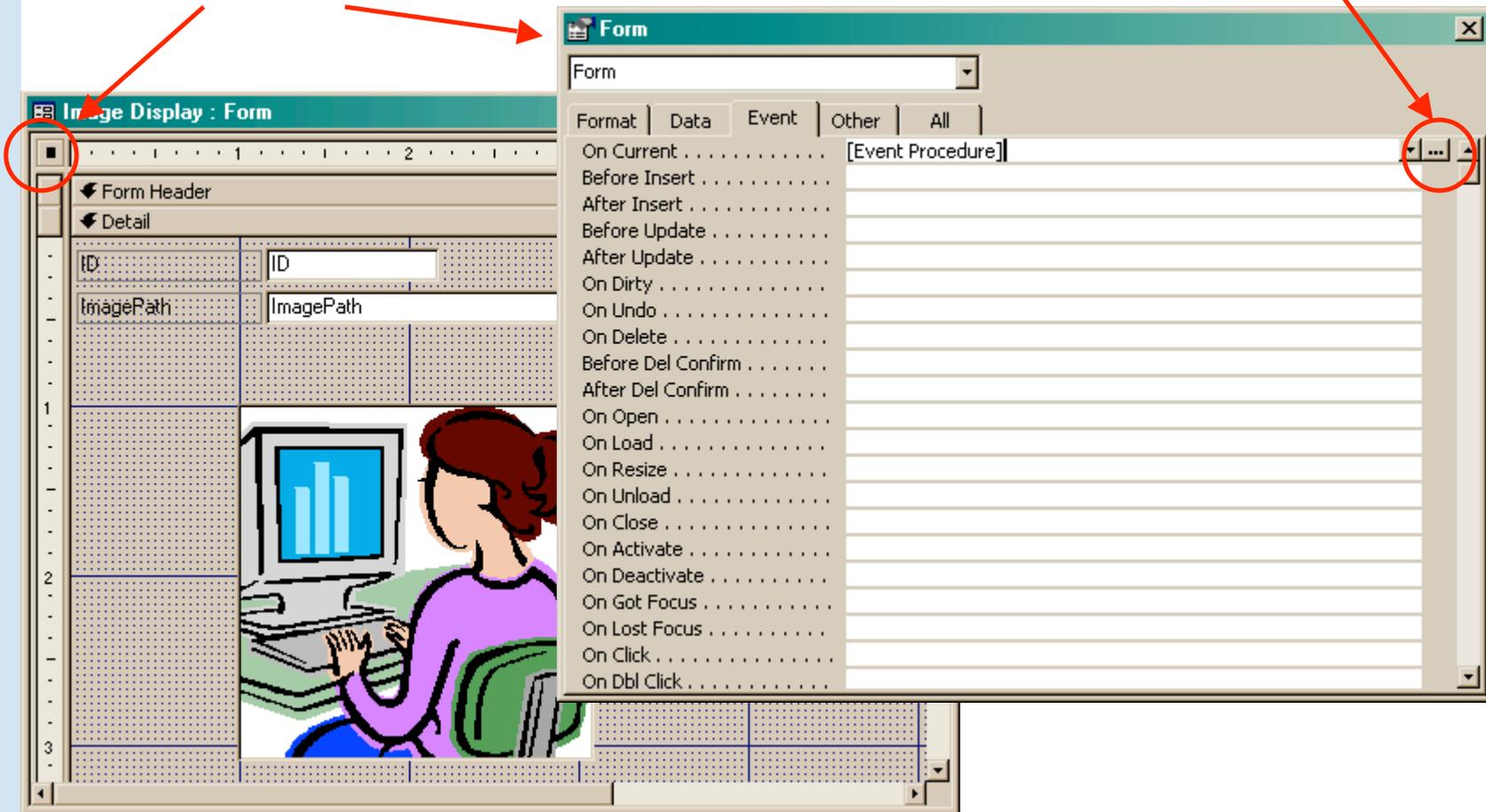
The screenshot shows a software window titled "Image Display : Form". The window contains a data grid with two columns: "ID" and "ImagePath". Below the grid is an image field displaying a cartoon illustration of a woman with brown hair in a bun, wearing a purple long-sleeved shirt, sitting at a desk with a computer monitor and keyboard. The window has a standard Windows-style title bar with minimize, maximize, and close buttons. The grid has a dotted background and is divided into sections for "Form Header" and "Detail".

image field that displays the image pointed to by ImagePath

How do we change the image? Event Handlers

double click here to get this

then click here



OnCurrent event handler for the form

```
employees - Form_Image Display (Code)
Form Current
Option Compare Database

Private Sub Form_Current()
If IsNull(Me![ImagePath]) Then
    Exit Sub
End If

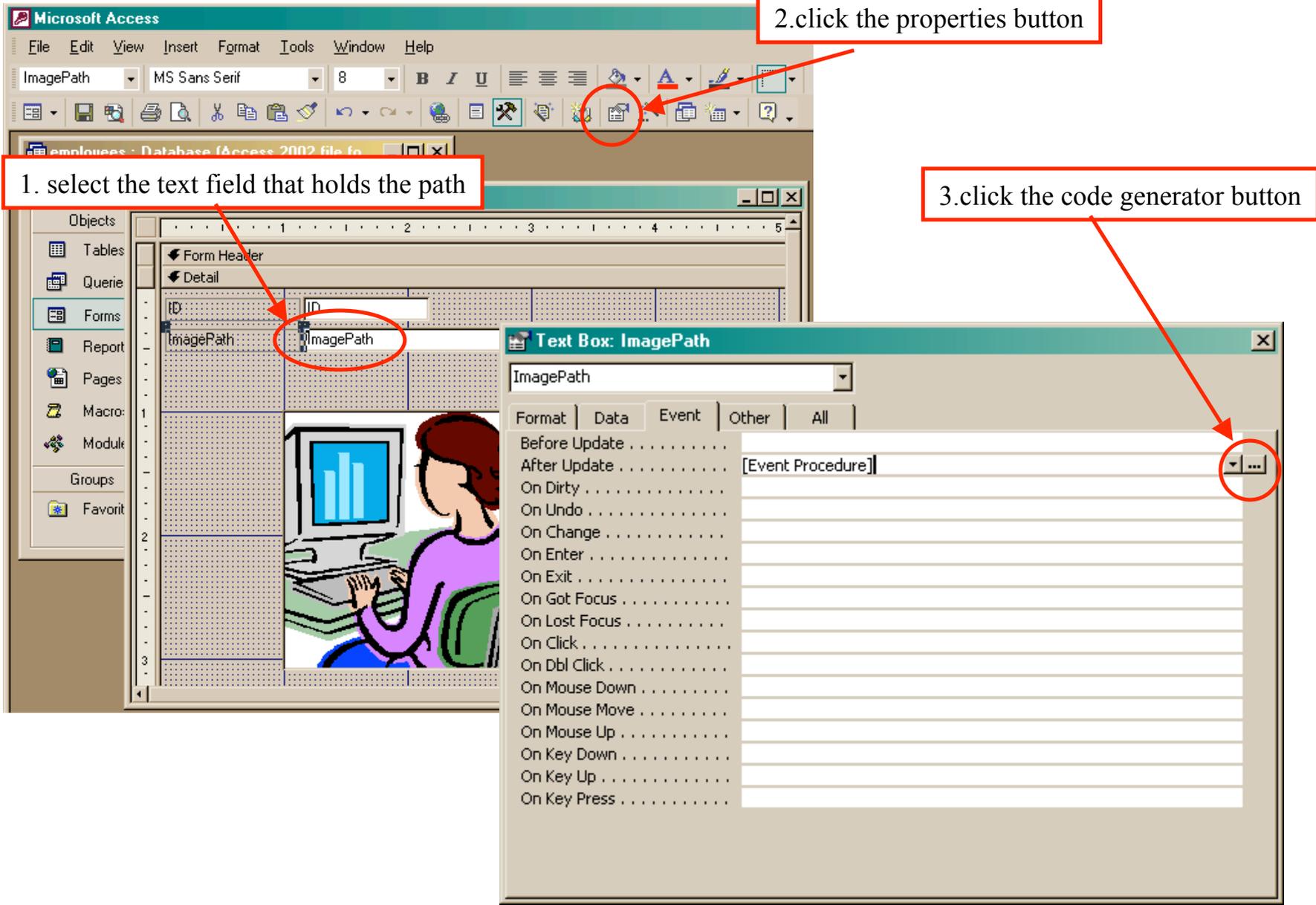
If (IsRelative(Me!ImagePath) = True) Then
    Me![ImageFrame].Picture = CurrentProject.path & "\" & Me![ImagePath]
Else
    Me![ImageFrame].Picture = Me![ImagePath]
End If
End Sub

Private Sub ImagePath_AfterUpdate()
If IsNull(Me![ImagePath]) Then
    Exit Sub
If (IsRelative(Me!ImagePath) = True) Then
    Me![ImageFrame].Picture = CurrentProject.path & "\" & Me![ImagePath]
Else
    Me![ImageFrame].Picture = Me![ImagePath]
End If
End Sub

Function IsRelative(fName As String) As Boolean
    ' Return false if the file name contains a drive or UNC path
    IsRelative = (InStr(1, fName, ":") = 0) And (InStr(1, fName, "\\") = 0)
End Function
```

ImagePath is the name of the text field that holds the path to the image on your form.

ImageFrame is the name of the Image control that displays the image on your form.



2. click the properties button

1. select the text field that holds the path

3. click the code generator button

AfterUpdate event handler for the field

```
employees - Form_Image Display (Code)
ImagePath AfterUpdate

Option Compare Database

Private Sub Form_Current()
    If IsNull(Me![ImagePath]) Then
        Exit Sub
    End If
End Sub

Else
    Me![ImageFrame].Picture = Me![ImagePath]
End If
End Sub

Private Sub ImagePath_AfterUpdate()
    If IsNull(Me![ImagePath]) Then
        Exit Sub
    End If

    If (IsRelative(Me!ImagePath) = True) Then
        Me![ImageFrame].Picture = CurrentProject.path & "\" & Me![ImagePath]
    Else
        Me![ImageFrame].Picture = Me![ImagePath]
    End If
End Sub

Function IsRelative(fName As String) As Boolean
    ' Return false if the file name contains a drive or UNC path
    IsRelative = (InStr(1, fName, ":") = 0) And (InStr(1, fName, "\\") = 0)
End Function
```

ImagePath is the name of the text field that holds the path to the image on your form.

ImageFrame is the name of the Image control that displays the image on your form.