

Session VI

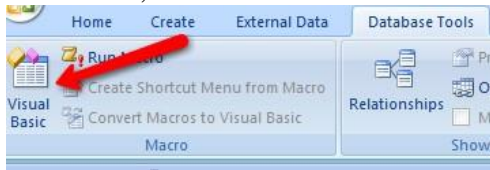
Modules:

Just as tables are the foundation of the database, modules are the foundation of any application. Remember that modules are created as are other objects in Access 2016. VBA code is written in units called subroutines and functions (procedures) both of which are stored in modules. There are 3 types of modules: Standard, Form class modules, and Report class modules.

Standard Modules:

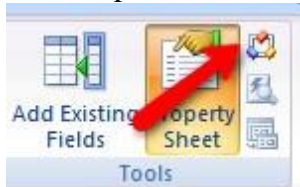
Standard Modules have general procedures that are not associated with any specific object in the database and are intended for frequently used procedures that can be run from anywhere within your database. As such, these typically apply to more than one object, but are coded once and stored in one place. This is considered to be a better programming standard in the industry and is called modular coding.

Standard modules are created the same way you would create any new object. By clicking to select the Database Tools tab and then selecting the Visual Basic button from the macro group. Access takes you to the Access Visual Basic editor (VBE). Finally, select Insert, then Module from the VBE menu.



Form Class Modules:

The form module is attached to a particular form you are designing. To view the form module, place the form in design view and then select the View Code Button.



This module is specifically attached to the form and any coding for the form will be inside this module. The code in the module can be triggered by the form's event properties, by its controls or sections, or by other code that is also contained within the module.

Report Class Modules:

To view the Report module, place the report in design view and then select the View Code Button.

Again this module is specifically attached to the report and any coding for the report will be inside this module.

Procedures:

Procedures are made up of functions and subroutines and perform actions and tasks. Whereas the function returns a value the subroutine does not. They are especially useful when a task is repetitive.

Procedures are an excellent way of dealing with repetitive or complex tasks and unless specified are considered **Public**, that is, open to the rest of the application. This takes us to the topic of **Scope**. Scope describes the visibility and accessibility of one procedure from another procedure. If a procedure is considered **Private** that means that procedure is only visible to the module in which it is written. If it is **Public** that means that it can be called or used by the rest of the application. Procedures can be made public or private by inserting the keyword **Private** or **Public** at the beginning of the first line of the procedure. As a programmer you will learn that there are advantages of restricting the scope of your procedures and of your variables. Additionally, if a procedure is made private, less computer resources are used because memory storage is released quickly.

Properties:

Objects have certain characteristics called properties. Another way to think of a property is a named attribute of an object that defines a certain characteristic, such as size, color, or an appearance aspect, such as whether the object is visible or whether it is automatically centered. The objects that you have been working with are referred to as Microsoft Access objects. They are created and maintained by the Access programming environment and are related to the user interface and modules.

Methods:

A method is a procedure that acts on an object. For example, the undo method for a control or a form lets you reset the control or the form when its value has been changed. You would use this method in VBA to clear a change to a record that contains an invalid entry.

Classes:

A class is a definition for an object that includes the object name, its properties and methods, and events associated with it. When you create an **instance** of a class, you create a new object with all of the characteristics defined by that class. The properties and methods associated with one form in the database are the same as the properties and methods of the form object class. Objects in Access can contain other objects. For example, form objects contain control objects, and table objects contained field objects. In Access, it is common to group related objects together.

When you create a form it is an object named form, with the properties, methods, and events that you expect with a form. If you create a new form or **instantiate** a form you are creating a new object that has all the properties and methods from the Form Object class. Similarly, if you create a report you are creating a report from the Report Object class.

Collections:

A collection is an object that contains a set of related objects (objects of the same class). For example, a forms collection contains all open forms of the database, and a reports collection contains all of the reports. The collection is not created by the user; rather, it is simply a means of referring to all the open forms or all the open reports, VBA, macros, and queries identify an object by specifying the collection to which it belongs.

Form Events:

All Window applications are **event driven**. Some of the more popular events for forms are discussed below.

Event	Description
Current event	Is one of the more commonly coded events. It happens each time focus moves from one record to another. The Current event is a great place to put code when you want to execute whenever the user displays a record.
BeforeInsert	Occurs when the first character is typed in a new record but before the new record is actually created.
AfterInsert	Occurs after the record has actually been inserted. You can use it to re-query a recordset after a new record is added.
BeforeUpdate	Occurs when the user tries to move to a different record or when the Records, Save Record command is executed. You can use the BeforeUpdate event to cancel the update process when you want to perform complex validations.
AfterUpdate	Occurs after the changed data in a record is updated. You might use this event to re-query combo boxes on related forms perhaps to log record changes.
Dirty	Occurs when the contents of the form, or the text portion of a combo box, change. It also occurs when you programmatically change the text property of a control.
Undo	Executes before changes to a row are undone. The Undo event initiates when the user clicks the undo button on the quick Access toolbar, presses the Esc key or executes code that attempts to undo changes to the row. If you cancel the undo event, the changes to the row are not undone.
Delete	Occurs when a user tries to delete a record but before the record is removed from the table. This is a great way to place code that allows deleting a record only under certain circumstances. If the delete event is canceled, the BeforeDelConfirm and AfterDelConfirm events never execute, and the record is never deleted.
BeforeDelConfirm	Takes place after the Delete event but before the DeleteConfirm dialog box is displayed. If you cancel that BeforeDelConfirm event, the record being deleted is restored from the delete buffer, and the Delete Confirm dialog box is never displayed.
AfterDelConfirm	Occurs after the record is deleted or when the deletion is canceled.

Open	<p>If the code does not cancel the BeforeDelConfirm event, the AfterDelConfirm event takes place after Access displays the confirmation dialog box.</p> <p>Occurs when a form is open but before the first record is displayed. With this event, you can control exactly what happens when the form first opens.</p>
Load	<p>Happens when a form opens, and the first record is displayed; it occurs after the Open event. A forms Open event can cancel the loading of a form, but the Load event can't.</p>
Unload	<p>Happens when a form is closed but before Access removes the form from the screen. It's triggered when the user closes the form, quits the application by choosing End Task from the task list or quits Windows, or when your code closes the form. You can place code that make sure it's okay to unload the form in the unload event, and you can also use the unload event to place any code you want executed whenever the form is unloaded.</p>
Close	<p>Occurs after the unload event, when a form is closed and removed from the screen. Remember you can cancel the unload event but not the close event.</p>

The With Statement

The With statement let you perform a series of statements on a specific object, such as a form or a control on a form, without restating the name of the object.

```
lblFirstName.BackColor = 255
lblFirstName.Caption = "First Name"
lblFirstName.BorderStyle = SOLID
```

The above code can also be achieved using the With statement.

```
With lblFirstName
    .BackColor = 255
    .Caption = "First Name"
    .BorderStyle = SOLID
End With
```

Bang Notations

In Session 4 we learn that the bang and dot identifier operators are used in VBA code to specifically identify objects and their properties. The bang is used to separate one object from another or from the object class, and the dot is used to separate an object from its property or method.

```
Forms!frmEmployees!cboNames.caption = "Names"
```

The above statement can also be replaced with the Me keyword which refers to the current active form or report.

Me.cboNames.caption = "Names"

The DoCmd object

The DoCmd Object carries out most macro actions in VBA code. Most macro actions have corresponding DoCmd methods, and the arguments correspond to the action arguments found in the macro window. For example, the DoCmd.OpenForm method in VBA is a code version of the Open.Form macro action. You will use a DoCmd object and its methods to create the code to open reports in the frmReportsSwitchboard form. We will discuss the format in the project

Using an If...Then...Else Statement:

An **If...Then...Else** statement conditionally executes a group of statements in VBA depending on the results of an expression. A **Control Structure** is a series of VBA statements that work together as a unit. A **VBA Statement** is a unit that expresses one kind of action, declaration, or definition in complete syntax. The **VBA expression** is a combination of keywords, operators, variables, and constants that yield a string, number, or object. Operators are used to perform arithmetic calculations, perform comparisons, combine strings, and perform logical operations. You have been using the ampersand (&) operator to create expressions in queries and reports that combine fields with other fields and strings of text.

Assigned Readings:

Please read Korol **Chapter 5** pgs. 109-128

Korol Hands-On 5-1 **Using the If...Then Statement** Pg 111

Korol Hands-On 5-2 **Using the Multi-Line If...Then Statement** Pg. 113

Korol Hands-On 5-3 **Using the If...Then AND Conditional Statement** Pg. 115-117

Hands-On 5-4 **Using the If...Then...Else Conditional Statement** Pgs. 117-118

Hands-On 5-5 **Using Nested If...Then Statement** Pgs. 120-122

Korol Hands-On 5-6 **Using the Select Case Statement** Pgs. 124-125

Korol Hands-On 5-7 **Using the Select Case Statement in a Function** Pgs. 127-128

You don't need to memorize all of the events but read through the chapters below so you are at least aware that each of the events exist.

Using Form Events Read Korol Chapter 24 and Chapter 25 Pgs. 695-746 but don't do the exercises. **Using Report Events** Read Korol Chapter 26 Pgs. 747-769 but don't do the Exercises.

Project

Now that you have some experience writing VBA code, you want to revisit some of the objects that you had created in the movie database and apply your knowledge to enhance their design. Amanda has also asked that you complete the user interface. She wants you to customize the toolbars, and startup properties to make the database even more user-friendly for MovieCam employees.

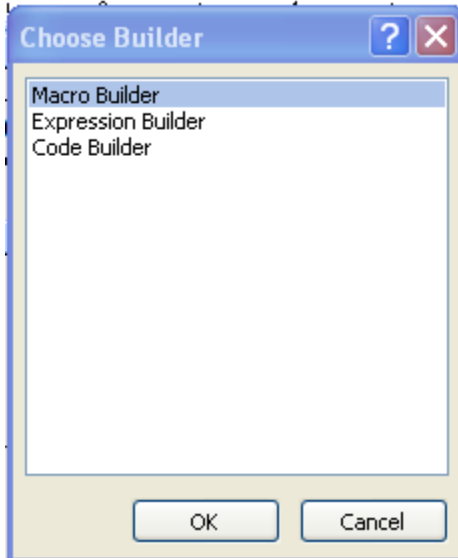
Amanda has also suggested that you split the movie database into a front-end file controlling all the database objects except for the tables, and a back-end file containing only the tables. You will then move the back-end to a shared network drive and distribute copies of the front-end to key users for testing the database at its current level of completion. You will continue development of the database in a developmental copy of the front-end that is located on a network drive to which the general user population does not have access.

There are a number of advantages to splitting a database in this manner, recall that data is stored only in tables. One advantage of splitting off the tables from the other objects is that users who need to access the data can customize their own forms, reports, and other objects while maintaining a single source of data on the network. Another advantage is speed. Particularly in a multiuser environment, allowing all users to have their own copy of the queries and forms on their own machines means that only data that they are querying, viewing, or printing has to be pulled across a network. If all objects are in one file on the network in a multiuser environment, then many people are trying to access the same queries and forms at the same time, instead of just the data in the tables. By running the queries and working with forms and reports on the local desktop machine, the activity of all these other objects does not have to be carried across a network -- just the data from the tables as needed for particular user operation.

With a small user base of 5 to 10 simultaneous users, there may be very little noticeable difference in performance. There are many factors to be considered in terms of network bandwidth, server memory and CPU speed, desktop computer memory and CPU speed, the design of the database, and the amount of data being accessed. Amanda feels that enough users in the company may be using the database at the same time to warrant splitting the database into two parts. Maintenance is typically a little easier this way as well. By maintaining a developmental copy of the front-end database in a separate location, you can develop and test enhancements, then simply deploy the revised front-end database to the users, without disturbing the back-end data.

We've already learned how to apply table and field validation rules. By writing code in a form module, you can also validate data as the user enters it. At movie technologies, timecards are always dated the Saturday following the work week. The product managers want the database to prevent the data entry clerk from entering a date that does not fall on a Saturday. You can write an event procedure to perform this validation.

Before you write the event procedure, you will change the default settings so that the code window opens automatically when you click the build button for an event property in a form or report. This will bypass the choose builder dialog box, which opens when you click the build button. (See below)



Changing the default settings so that the code window opens automatically:

Step 1

Start Access and open the Session 6 movie database.

Step 2

Click on the **File Tab** and select the **Options** button and on the menu to the left select **Object Designers** and in the forms or reports section select **Always Use Event Procedures**.

Object Designers

Proofing

Language

Client Settings

Customize Ribbon

Quick Access Toolbar

Add-ins

Trust Center

☒ Show Property Update Options buttons

Query design

☒ Show table names

☐ Output all fields

☒ Enable AutoJoin

Query design font

Font: Segoe UI

Size: 8

SQL Server Compatible Syntax (ANSI 92)

☐ This database

☐ Default for new databases

Form/Report design view

Selection behavior

☒ Partially enclosed

☐ Fully enclosed

Form template: dmfrmTemplate

Report template: dmrptTemplate

☒ Always use event procedures

Step 3

Click the **OK** button

You will use of the BeforeUpdate event in the frmTimeCards form to test an entry in the TimeCardsDate field. If the date entered does not fall on Saturday, the entry is canceled. You'll use the DatePart function to test the day of the week entered in the TimeCardsDate field. The DatePart function returns an integer containing a specific part of the given date.

Validating the Data Entered in the TimeCardDate Field in the TimeCards Form:

Step 1

Open the **frmTimeCards** form in design view.

Step 2

Click the **TimeCardDate** text box, and then click the **Property Sheet** button in the Ribbon, if necessary, to open the property sheet.

Step 3

If necessary, click the **Event** tab, and then click the **Build** button for the **BeforeUpdate** property. Your screen should look like the one shown below: the insertion point is positioned between the first statement of the event procedure and the last.

```
Private Sub TimeCardDate_BeforeUpdate(Cancel As Integer)

End Sub
```

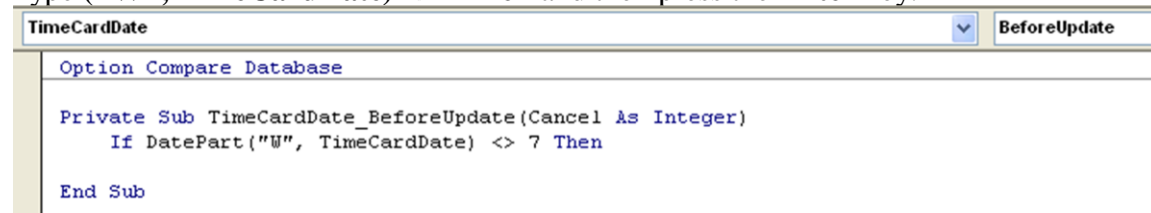
The statements you will use in the code include an If...Then conditional statement to test the entry. We'll also include a message box statement to give the user an error message if the date entered does not fall on a Saturday.

Step 4

Press the tab key to indent the line, and then type **If DatePart** (A banner opens below the function and provides syntax information for the DatePart function.)

Step 5

Type **("W", TimeCardDate) <> 7 Then** and then press the **Enter** key.



```
TimeCardDate BeforeUpdate

Option Compare Database

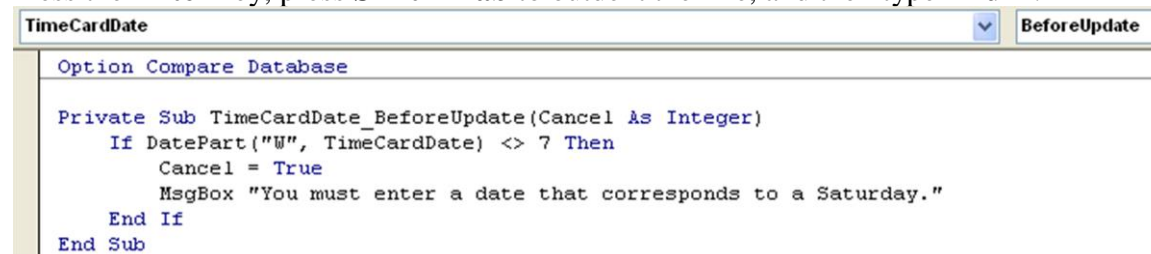
Private Sub TimeCardDate_BeforeUpdate(Cancel As Integer)
    If DatePart("W", TimeCardDate) <> 7 Then
End Sub
```

Step 6

Press the tab key to indent further, type **Cancel = True**, to move to the next line, and then type **MsgBox "You must enter a date that corresponds to a Saturday."**

Step 7

Press the **Enter** key, press **Shift + Tab** to outdent the line, and then type **End If**.



```
TimeCardDate BeforeUpdate

Option Compare Database

Private Sub TimeCardDate_BeforeUpdate(Cancel As Integer)
    If DatePart("W", TimeCardDate) <> 7 Then
        Cancel = True
        MsgBox "You must enter a date that corresponds to a Saturday."
    End If
End Sub
```

Step 8

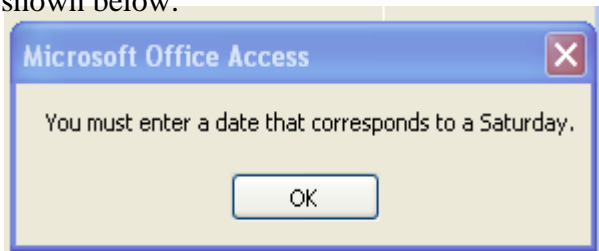
Click **debug** on the menu bar, and then click **compile**. Translating modules from VBA to a translated form is called **compilation**. Access will compile all the modules in the session database and check them for syntax errors.

Step 9

Close the Visual Basic window; close the property sheet, and then switch to Form view to test the procedure.

Step 10

Add a **new** record, type **200** in the Time Card No field, type **9/21/16** in the TimeCardsDate field, and then press the **Enter** key. You will see the error message shown below:



Step 11

Click the **OK** button, and then press the **Esc** key twice to cancel the entry.

Step 12

Save your changes and close the form.

Using a select case statement requires even less code to type and is more efficient than using several Else If constructs. Next you will add the code to the frmDataSwitchboard form using the Select Case statement to open a particular form based on the user's choice from the switchboard.

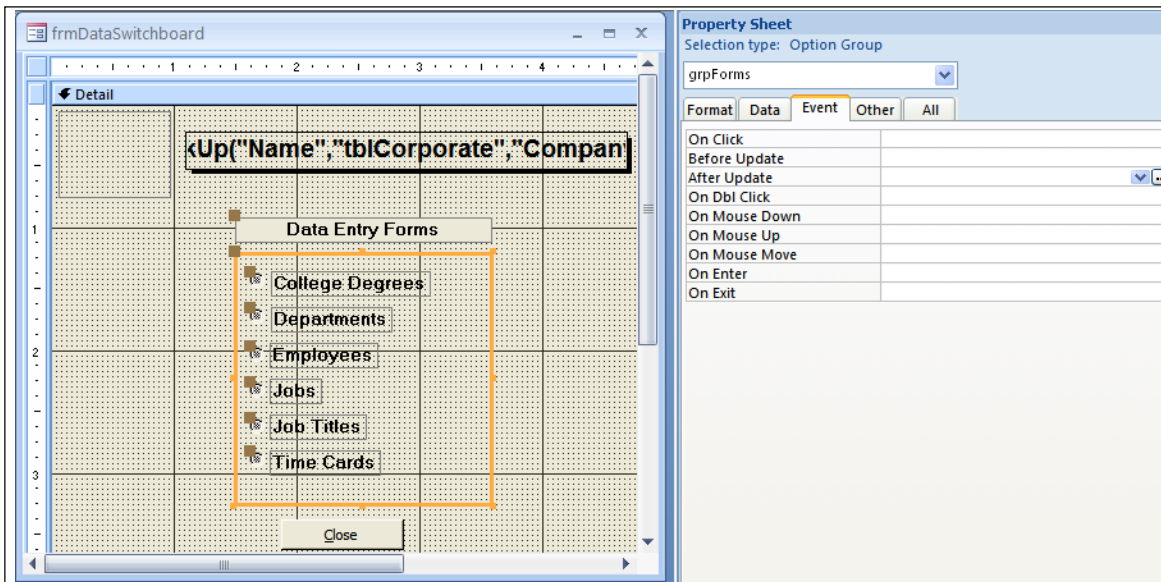
Adding code to the frmDataSwitchboard form:

Step 1

Open the frmDataSwitchboard form in design view.

Step 2

Click the frame of the option group once to select it, click the **Properties** button on the Ribbon, and then click the **Event** tab on the property sheet (if necessary) see below:



Step 3

Click the **After Update** property text box, and then click Event Procedure. The code window opens in the Visual Basic window. The new event procedure for the option groups after update event property contains a Private Sub and End Sub statements. You can now enter the statements to open a form, based on the option the user selects in the option group.

Step 4

Press the **Tab** key, type **Select Case grpForms**, press the **Enter** key, press the Tab key to indent, and then type **Case 1**.

Step 5

Press the **Enter** key, press the **Tab** key, and then type. **DoCmd.OpenForm "frmDegrees"**.

Step 6

Press the enter key, and copy both the statements you just entered and paste them in the blank line you just created before the end sub statement, change Case 1 to **Case 2**, and then change frmDegrees to **frmDepartments**. If necessary, indent or outdent these statements so that they are aligned as shown below:

```

Private Sub grpForms_AfterUpdate()
    Select Case grpForms
        Case 1
            DoCmd.OpenForm "frmDegrees"
        Case 2
            DoCmd.OpenForm "frmDepartments"
    End Sub

```

Step 7

Repeat steps six, changing the Case 2 to **3** and the form to **frmEmployees**.

Step 8

Repeat steps six, change the Case 3 to **4** and a form to **frmJobs**.

Step 9

Complete the code (starting with the Case 5 statement) as shown below: Don't forget to enter **End Select**

```

Private Sub grpForms_AfterUpdate()
    Select Case grpForms
        Case 1
            DoCmd.OpenForm "frmDegrees"
        Case 2
            DoCmd.OpenForm "frmDepartments"
        Case 3
            DoCmd.OpenForm "frmEmployees"
        Case 4
            DoCmd.OpenForm "frmJobs"
        Case 5
            DoCmd.OpenForm "frmJobTitles"
        Case 6
            DoCmd.OpenForm "frmTimeCards"
    End Select
End Sub

```

Step 10

Compile your code, close the Visual Basic window, close the Property sheet, and switch to Form view to test the procedure.

Step 11

Click each of the forms in the option group to be sure that the correct form opens, and then close each form to return to the switchboard.

Step 12

Save your work, and then close the switchboard to return to the Navigation Pane.

Creating custom menus and toolbars in a program is an important part of the database development process. In many cases, you want to restrict access to certain features and objects so that users cannot inadvertently delete data, view sensitive data, modify the design of an object, modify macros or VBA code, and so forth. A simple way to do this is to customize existing menu and toolbars, or create new ones.

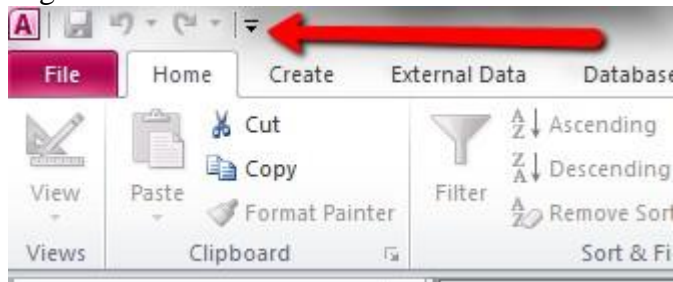
In the case of the movie technologies database, users should be given access to only those menu commands and toolbar buttons that they need to enter and edit records. You need to restrict access to the default toolbars and create a custom shortcut menu so that the command to switch to an objects design view is not available. You also need to create a custom toolbar that contains only those buttons users need to work on the database.

Before you modify the menus and toolbars for users, you want to customize the menus and toolbars you're using to design and develop the database. Because you spent a lot of time aligning controls on forms, you will add the Aligned Left and Aligned Top buttons to the formatting toolbar.

Customizing the Formatting Toolbar:

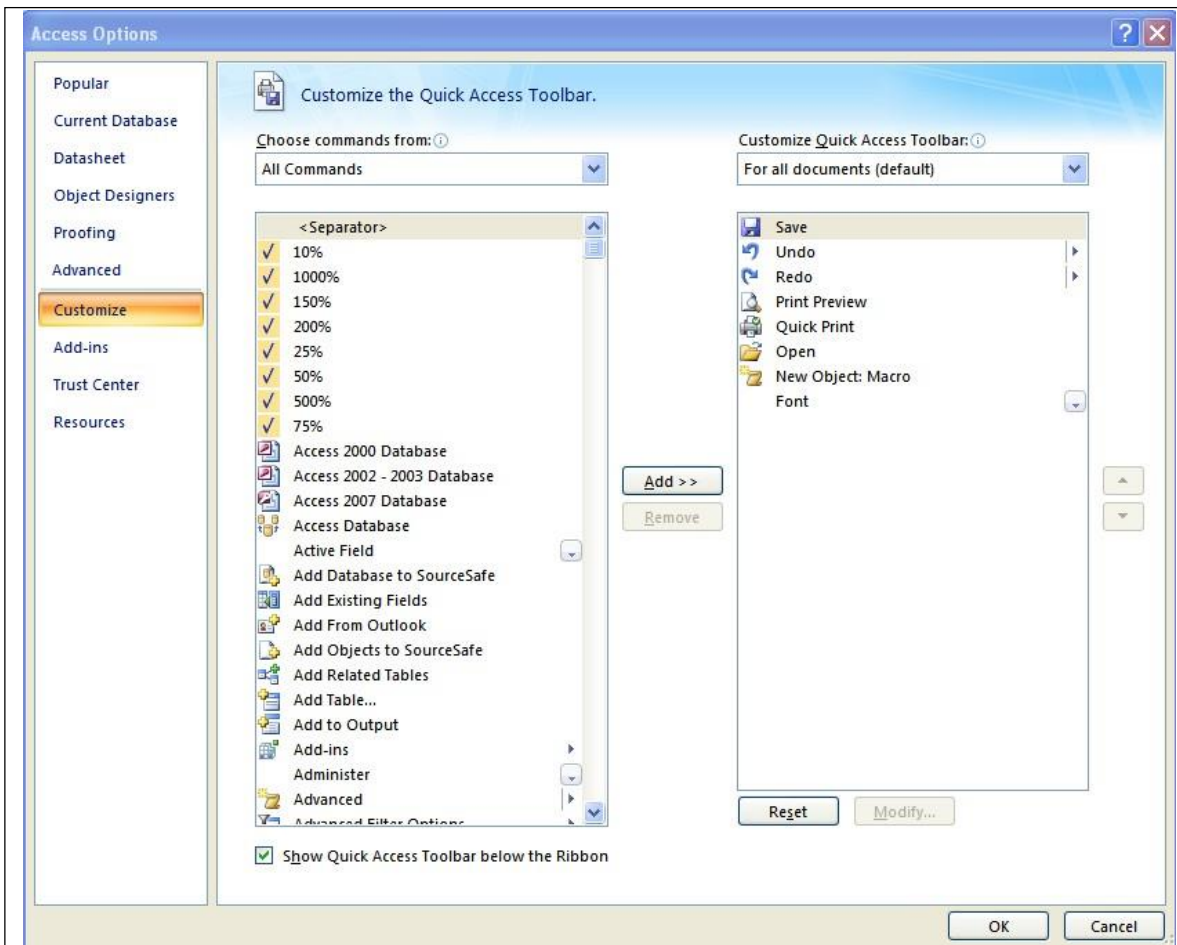
Step 1

Right-click on the toolbar. See below:



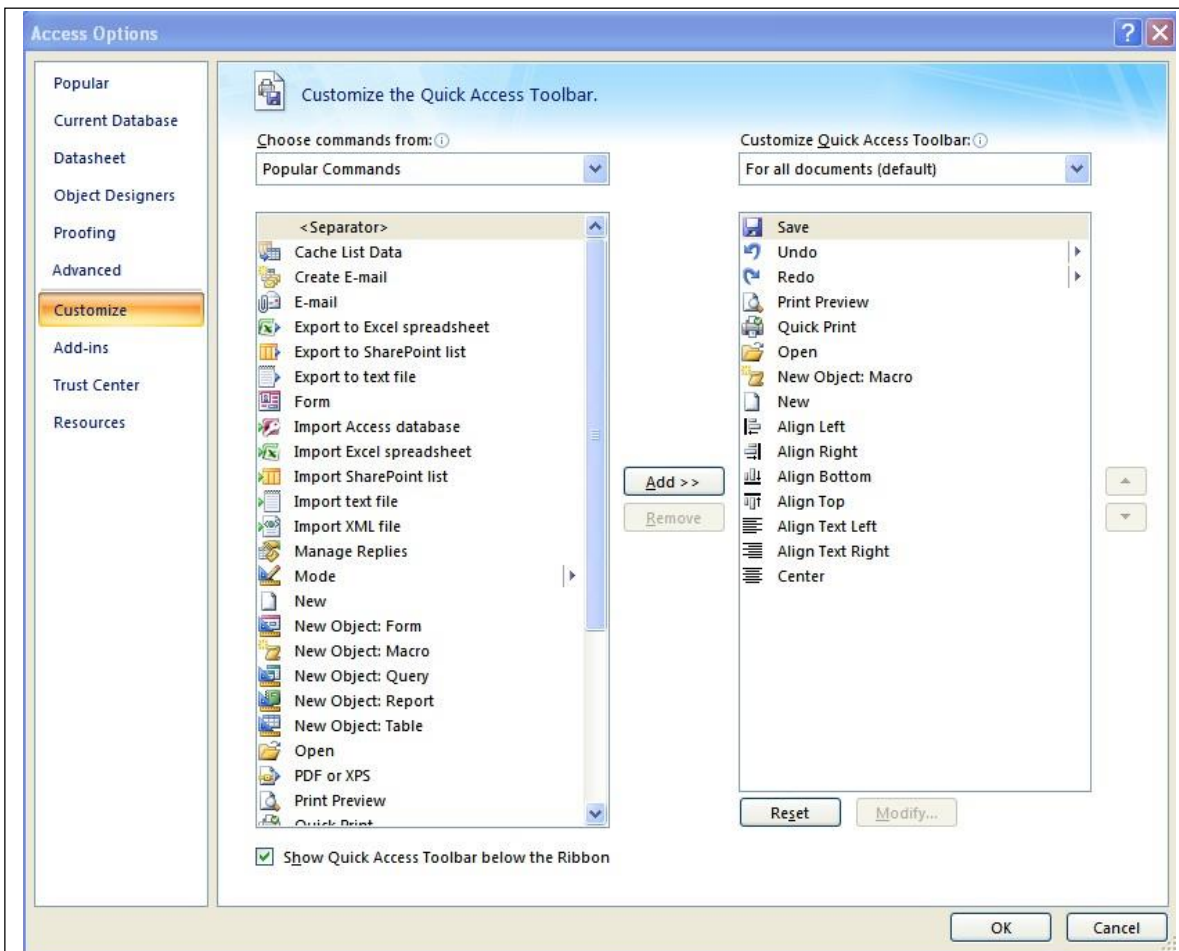
Step 2

Point on the Customize Quick Access Toolbar (see above) click and Select **More Commands** then **All Commands**



Step 4

Because we will spend a lot of time aligning controls on forms, we will add Align Left, Align Right, Align Bottom, and Align top. Since we also need to align text in the various controls let's add Align text right, Align text left, and Center by selecting these commands. To bring the command to the toolbar you need to select the Add button:



Step 5

Since we will be using the Properties Sheet on many of our objects let's place that on the customized toolbar:

Step 6

You decide to include the Close, Print, and Exit commands on the toolbar.



Step 7

Click OK to close the toolbar.

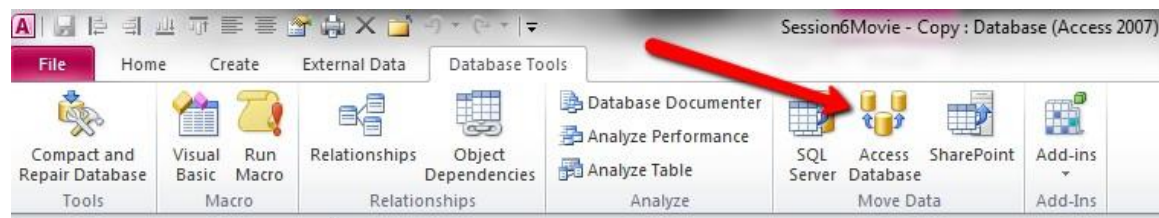
Many Access developers agree that Access databases should be split into two files: one containing only Access tables, often called the **back-end**, and another containing all other objects, often called the **front-end**.

Splitting a database this way allows you to link tables from the shared back-end to the front-end, which is often placed on local drives. Keeping the front-end on the user's computer minimizes network traffic and improves performance. The major benefit, however, is that ongoing development of the database objects can be accomplished without interfering with users working in the database. If one file is placed on a shared network drive, the forms and reports cannot be modified when users are working with them. If the database is split, however, changes can be made to the master copy on the front-end and then copied to an individual user's machine.

Finally, if you have a database where you are not locking down the user interface, or splitting the database, you can provide the user with a unique copy of the front-end. This allows the user to develop their own forms and reports without affecting the database of other users. If a user accesses the design view of a form or report and damages it, only that user's copy is affected.

Now you'll use the database splitter wizard to split the movie database. Before you do, make a backup copy of the database contained work you have done up to this point.

Splitting the Session 6 Movie database:



Step 1

The database you have loaded for this session is Session6Movie.accdb. To reduce problems before beginning the split make certain that you have preceded the file name with your name as you usually do when you submit the project. We just need to make sure your name is attached to the file name before you begin the split.

To create the file simply make a copy of the file and give it a new name. You need to have these two files in your CIS28A folder: Session6Movie.accdb and yourNameSession6Movie.accdb.

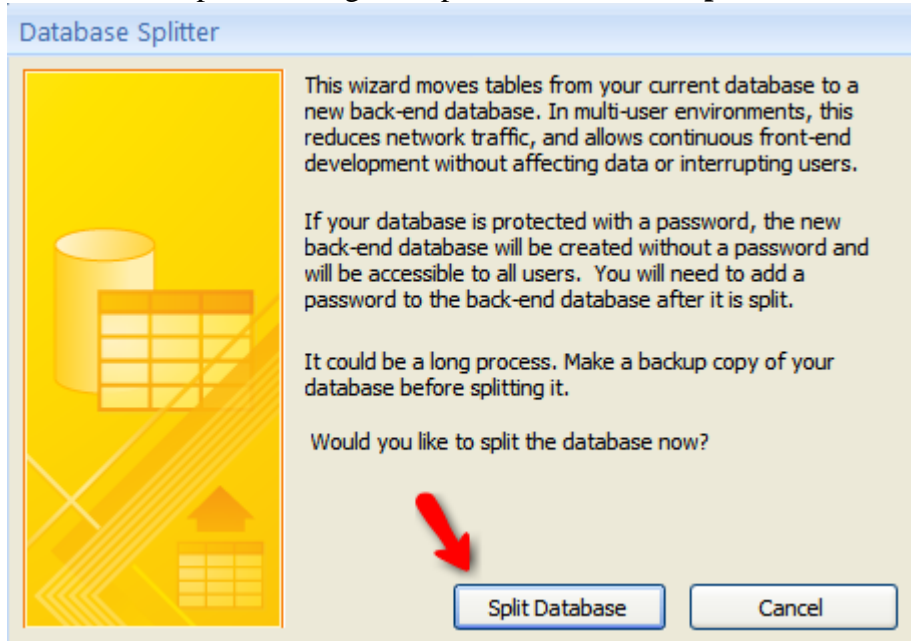
In the following steps you are working with yourNameSession6Movie.accdb

Step 2

Select the Database Tools tab on the Ribbon and then click the **Access Database** button to begin splitting the database.

Step 3

The database Splitter dialog box opens. Please select **Split Database**.

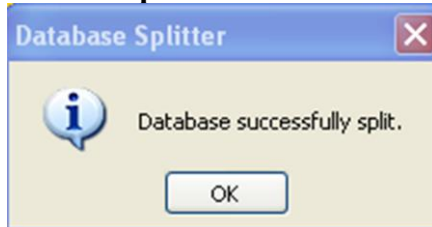


Step 4

The Database Splitter dialog box opens. The back-end database will contain the tables from the Session's database. **Note: It is important that you name the back-end file as Session6MovieBackup.accdb. I will be referencing that file when I check your project. The file will also be on my computer so you don't need to send it.**

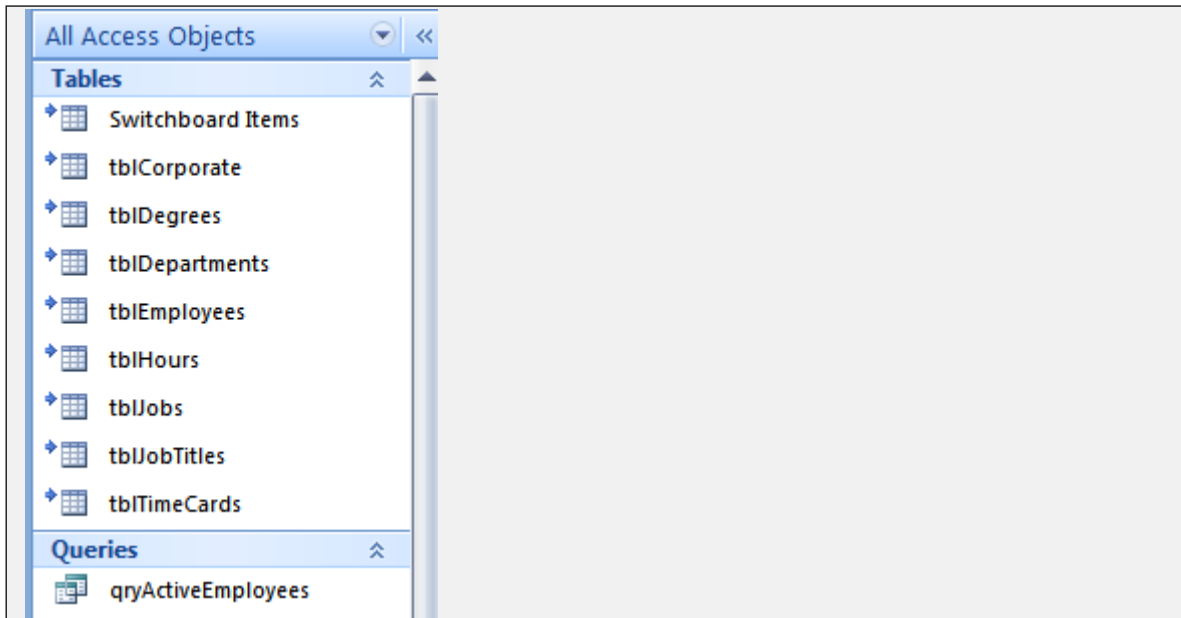
Step 5

Click on **Split** then click on **OK** to verify that the database has been successfully split.



Step 6

View the Navigation Pane to verify that all the tables have an arrow pointing to each of the tables indicating that they are now linked.



Now that the database is split, you want to move the back-end database to DeskTop on your computer. If you move the back-end database to a new folder or drive, the links to the front-end database will need to be refreshed.

The linked table manager is a utility that allows you to refresh links to tables that are contained in a separate database. To see how this feature works, you will move the back-end database to the DeskTop on your computer. Then you open the YourNameSession6Movie front- end database, test the link tables, and then refresh the links to the back-end database.

Moving session 6 movies to C: BackEnd and update the linked tables:

Step 1

Close the session 6 movie database and then exit Access.

Step 2

Open Windows Explorer.

Step 3

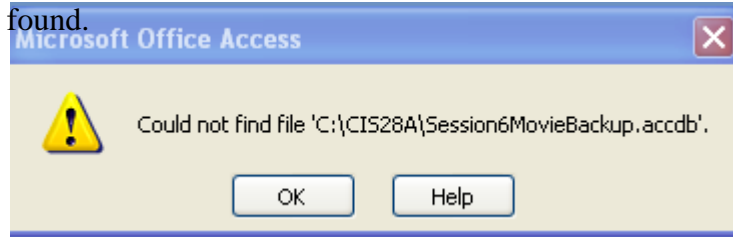
Navigate to your folder Access28A, drag the **Session6movieBackup** database that you just created (ends with Backup) to **C: BackEnd** on your computer. Do not drag a file to a different drive or you will copy it, and this exercise will not work.

Step 4

Close **Windows Explorer**, start Access, and then open YourNameSession6Movie database.

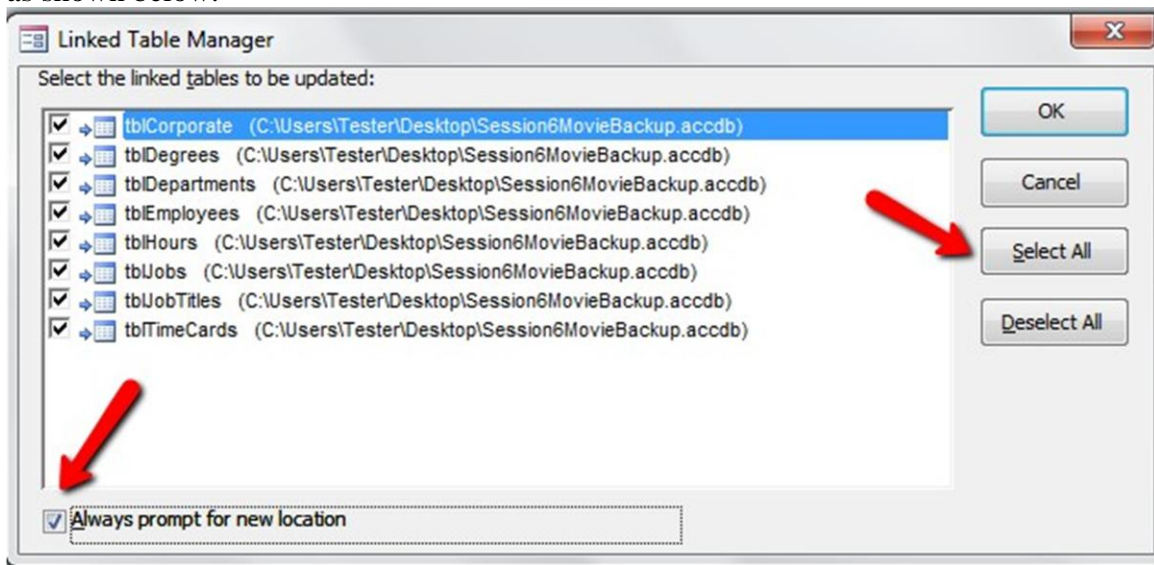
Step 5

In the Navigation Pane Double-click the tblEmployees table. An error message opens, explaining that the back-end database, **Session6movieBackup.accdb**, could not be found.



Step 6

Click the **OK** button to close the error message, click the **External Data** tab in the Ribbon, Click on **Linked Table Manager**. The linked table manager dialog box opens, as shown below.



Step 7

Click the **select all** button to select all the tables in the database, and then click the **Always prompt for new location** checkbox so that the linked table manager will prompt you for the location of the back-end file.

Step 8

Click the OK button, navigate to the C: BackEnd, click Session6MovieBackup to select it, and then click the open button.



Step 9

Click the OK button when the message box opens and tells you that all selected link tables were successfully refreshed, and then click the close button in the linked table manager dialog box.

Step 10

Open the tblEmployees table. The table should open without any error messages appearing and should contain the employees' data.

Step 11

Close the session 6 movie database, and then exit Access.

When you submit your project you do not have to submit the back-end file.

The database is split into a front and back end, and is ready to be copied for use on individual computers. The menus and switchboards are in place. Amanda is happy with your continued progress on the MovieCam database.