

ACCESS

IN VISUAL BASIC

- Microsoft jet database engine is used retrieve data from and store in a database(Ex. Access).
- Two ways to connect VB program to a database:
 - 1. Writing code
 - 2. Data control
- Data Control (prefix-dat)
 - Enables you to move from record to record and to display information inside other control instances created on form.
 - Setting properties of Data Control, use button to locate different records without writing any code.
 - Can only connect to existing databases

- Properties:
 - **Caption** defines text displayed inside Data Control
 - **Connect** provides information about type of database to be used
 - Access, dBase, FoxPro, and Paradox (default is Access)
 - **DatabaseName** identifies name and location of database file.
 - **EditMode** contains status of current record during edit
 - Supported by Recordset object(provides view of table or query stored in database) and can assume following values:
 - 1. dbEditName- no editing operation is in progress
 - 2. dbEditInProgress- indicates edit method is called and current record occupies copy buffer
 - 3. dbEditAdd- indicates AddNew method has been called and current record in copy buffer is **new** record has not been saved to database table or query
 - **RecordsetType** determines which operations can be performed on table or query identified when set RecordSource property

- By default, property is set to (1) 1-Dynaset
 - Dynaset is a dynamic view of information contained in database table or query.
 - Used to view and change contents of rows from one or more tables, depending on need
 - (2) 0-Table yields a copy of a table
 - (3) 2-Snapshot is read-only, forward-only
- Identify database table or query
- Set RecordSource property to table or query that already exists.
 - 1. Select Database name
 - 2. Select table or query
- Recordset Object (rst) – created by data control to store information about the database object
 - Use Recordset object at run time to locate different records, add, change, or delete records
 - RecordSet object is not a visible object (cannot create an instance via toolbox)- reference RecordSet in code

- Recordset object provides a view of a table or query stored in database. It does not respond to any event.
- Recordset Properties
 - **BOF**
 - 1. When current record points in positioned just before 1st record, setting is True
 - 2. Otherwise setting is False
 - **EOF**
 - 1. When pointer is positioned immediately after last record, setting is True
 - 2. Otherwise is False
 - **FindFirst** method sets **NoMatch** property of **Recordset** object.

- If one or more records are found, **NoMatch** property is set False.
- If no record is found, **NoMatch** property is set to True.
- **MoveFirst** locates 1st record in Recordset object
- **MoveLast** locates last record in Recordset object
- **MoveNext** locates next record in Recordset object
- Programmer sets Connect, DatabaseName, RecordSource, and RecordSetType properties at design time.
- Data Control can retrieve information from the RecordSource at run-time.
 - Creates an instance of Recordset object and stores a reference to the object in Recordset property of Data control.
 - Once Data Control has created Recordset object, can locate records and perform operations.

- Data Control simplifies process of working with Recordset objects by providing buttons that navigates through records in Recordset object without explicitly calling methods.
- Limitations of Data Control
 - Does not support a method to remove a record from a Recordset object.
 - Performs this operation by a call to a method underlying the Recordset object stored in Recordset property of Data Control
- Recordset object is considered a Data Access Object (DAO)
 - DAO enables you to use programming language to access and manipulate data in databases.
 - Allows one to manage databases, their objects, and structures
 - Click on button located on the control instance to navigate through records without writing any code

- Using Bound Controls:
- Bound Control (data-aware control) displays changes in current record of Recordset object on the form (ex. Textbox)
 - Many controls can hold text: textbox, label, and listbox
 - imagebox and picturebox can be bound to a field in Recordset object
- Setting Bound Control Properties:
 - 1. DataSource set to an instance of data control down on same form(data control object)
 - 2. DataField set to a field in table or query

- Calling Methods of Recordset Object Explicitly
 - Note: Data Control does not explicitly support MoveFirst, MoveLast, MoveNext, and MovePrevious
 - Recordset object supports aforementioned methods
- Recordset property of Data Control stores reference to Recordset object at run-time.
- Calling an object's methods requires a period between object and methods name.
 - Example: datContact.Recordset.MoveNext
- Creating an object variable:
 - [Private | Dim] VariableName As [New] object
 - Note: **New** allows you to create new objects
 - **Object** is a placeholder for the kind of object you want to reference
 - Example: Private rstContact As Recordset (used to reference an instance of a Recordset object
 - Private rstContact AS Recordset
 - rstContact.MoveFirst
 - After declaring object variable, variable must be set to point to an existing object or to create a new instance of an object
 - Use the set statement

- Set VariableName=[New] objectExpression | Nothing
 - New creates a new instance of the object
 - ObjectExpression can consist of any instance of an object, such as a recordset, a form, or an instance of control
 - When **New** is absent, instance of the object must already exist.
 - Nothing disassociates an object variable from an actual object
 - Frees memory and system resources used by the object
 - Example: Private rstContact As Recordset
 - Set rstContact = datContact.Recordset
 - Note: assumes datContact is existing instance of Data Control
 - Set statement assigns a reference to the recordset stored in Data Control to object variable rstContact
 - Two references to the same object exist
 - Example: Private rstContact As Recordset
 - Set rstContact = datContact.Recordset
 - rstContact.MoveFirst
 - datContact.Recordset.MoveFirst
 - Note: the last two statements refer to the same record object

- BOF & EOF- Beginning of File and End Of File
- If first record is current one, and user tries to locate the previous record, a current record will no longer exist, and BOF property will be set to True.
- The same problem will occur if user tries to locate the next record when last record of table is current record.
 - Solution: rstContact.MovePrevious
 - If rstContact.BOF Then
 - True =no current record object
 - rstContact.MoveFirst
 - End If
 - Same is true for EOF situation
 - rstContact.MoveLast

- Adding New Records to Database
 - Object.AddNew creates a new blank record ready to be edited, rstContact.AddNew
 - If RecordsetType property of the Data Control was previously set to 1-Dynaset.
 - All new records are inserted at the end of the Recordset object, even if the recordset was presorted.
 - While in edit mode, the user should be prevented from locating a different record of performing any record editing action than updating the new record from database.
 - AddNew method operates on Recordset object when program calls AddNew method, it creates a new blank record in copy buffer.

- Updating Existing Database Records

- Make the current record available for editing by using Edit method that pertains to Recordset object.
- This process places data from current record in copy buffer.
- After making changes, can save them explicitly with the Recordset object's Update method.
- Note: using MovePrevious or MoveNext methods, the changes in the copy buffer are lost.
- Requery method of Recordset object will enter a new record in sorted order.
- Not using Requery, newly added records will always appear at the end of the Recordset object until program is run again.
- Note: Requery refreshes the Recordset object.
 - Example: Object.Edit
 - Object.Update
 - Object.Requery
- Object must be a valid instance of a recordset.

- To enable editing on the record, program must call Edit method.
 - After editing is complete, it must call Update method.
- Update method writes contents of copy buffer to underlying recordset.
 - Saves a new record created by AddNew method and existing record edited with Edit method.
- Requery method reloads contents of underlying table or query into recordset.
 - To ensure any bound text boxes will display information from a new recordset, you can call a method like MoveFirst after call Requery method.
- Recapping:
 - Edit makes current record available for editing by copying its contents to copy buffer.
 - Update writes contents of copy buffer back to recordset
 - Requery refreshes the recordset
 - Object.Delete deletes current record from a Recordset object, which removes record from current database table.

- Searching Records

- Object.FindFirst criteria:
 - The object can be any valid Recordset object.
 - The FindFirst method locates the first record in a Recordset that satisfies the criteria.
 - Criteria defines which record in the database is located.
 - If several records meet the criteria, then the first record found becomes the active record.
 - Strings must be enclosed in single quotation marks, a data must be enclosed in pound sign.
- Object.FindNext criteria:
 - User after FindFirst method to locate subsequent occurrences of records matching the criteria.
 - Example: rstContact.FindFirst "fldClientId = 3"
 - rstContact.FindFirst "fldLastName = 'Zorn' "

- Example:
 - Dim strLastName As String
 - strLastName = InputBox("Enter the Last name","Find")
 - If strLastName <> "" Then
 - rstContact.FindFirst "fldLastName = '&' '&' & strLastName
 - If rstContact.NoMatch = True Then
 - MsgBox "Cannot find " & strLastName
 - End If
 - End If
- Verifying Correctness of Data with ValidateEvent
 - Private Sub Object_Validate([index As Integer] action As Integer, Save As Integer)
 - Statements
 - End Sub
 - Where Object is any instance of a Data Control
 - Validate event occurs based on these actions:
 - 1. current record is repositioned
 - 2. record is updated
 - 3. record is deleted
 - The argument is passed to Validate using the action argument.
 - Index argument is optional and identifies the Data Control if it is a member of a control array.
 - Action argument is an integer that indicates which of several operations caused Validate event to occur.
 - Each of these actions can be expressed as a constant.
 - Statements in Validate event procedure determine which constant the action argument is set.
 - Example: vbDataActionCancel, vbDataActionMoveFirst, vbDataActionMovePrevious, MoveNext, MoveLast, AddNew, Update, Delete, Find.
 - Save argument is Boolean expression specifying whether bound data has changed.

- Example for datContact using Update method:

- Actions = cancel current action, updating record, wherever program detects invalid data.
- If input is valid, update proceeds.
 - Dim strMessage As String, intReturn As Integer
 - If Action = vbDataActionUpdate Then
 - If txtLastName.Text = "" or txtFirstName.Text = "" Then
 - strMessage = "You must enter both a first & last name."
 - action = vbDataActionCancel
 - End If
 - If IsData(txtDataAdded.Text) = False Then
 - strMessage = strMessage & Chr(vbKeyReturn) & _
 - "The date " & txtDataAdded.Text & _
 - " is not a date."
 - Action = vbDataActionCancel
 - End If
 - If IsNumeric(txtEstimatedSales.Text) = False Then
 - strMessage = strMessage & Chr(vbKeyReturn) & _
 - "The estimated sales " & txtEstimatedSales.Text & _
 - " is not a number."
 - End If
 - If Action = vbDataActionCancel Then
 - intReturn = MsgBox(strMessage, vbOkOnly, "Input")
 - End If
- End If