Lab 10.2: Creating a New Database Using DAO

Objectives

After completing this lab you will be able to:

- Create a new database using Data Access Objects (DAO)
- Create new tables using DAO
- Add tables to the database using DAO

Prerequisites

You should know how to work with dialog boxes and have completed Chapter 10 before attempting this lab.

Lab Setup

This demonstration shows what you will accomplish during the lab.

Estimated time to complete this lab: **20 minutes**.

Exercise

The following exercise provides practice working with the concepts and techniques covered in this chapter.

Exercise 1: Creating a DAO Database Programmatically

In this exercise, you will add code to the baseline application to create a DAO database using table and field information obtained from the user.

Copy the contents of \Labs\C10\Lab02\Baseline to your working directory.

The completed code for these exercises is in \Labs\C10\Lab02\Xxx, where Xxx is the exercise number.

Exercise 1: Creating a DAO Database Programmatically

In this exercise, you will add code to the baseline application to create a DAO database using table and field information obtained from the user.

About the Baseline Application

The supplied base project is an SDI application with a view derived from **CTreeView**. The database support option chosen was Header Files Only.

The user interface enables the user to enter table names and field data into a tree structure. The information is editable.

You will modify the **CMDBDoc::OnFileSaveAs** handler to create the database.

➢ **Create a new database**

1. Add a protected member **CDaoDatabase** m_daoDB to the document class.
2. Open the file and locate the code for **CMDBDoc::OnFileSaveAs**.
3. In the first **try** block, before the **while** statement, call **CDaoDatabase::Create** to create the database represented by m_daoDB. The **CFileDialog** object, dlg, has the filename the user entered. Use **CFileDialog::GetFileName** to determine the database name.

```
m_daoDB.Create( dlg.GetFileName( ) ); //Create & open database
```

➢ **Create a new table**

1. Inside the first while block, before the nested while, construct a **CDaoTableDef** object to use with the database.

```
CDaoTableDef daoTblDef( & m_daoDB );
```

2. Use the **CString** object, strTableName, to create the **CDaoTableDef** object.

```
daoTblDef.Create( strTableName );
```

Note The function **CMDBView::TableNameIterator** initialized strTableName in the while loop's conditional expression.

3. Inside the nested while loop, determine the data type the user entered. **CMDBView::FieldInfoIterator** encoded the data type in the variable nFieldType. A text field is represented by zero (0), an integer field by one (1), and a floating point field by two (2). Since these are not the same values as the constants dbText, dbLong, and dbDouble, you need to use a conditional expression to convert to the correct value. Additionally, if a field is a text field, the iterator initialized nTextFieldLength with the length parameter of the text field. If the field is numeric, nTextFieldLength contains negative one (-1).

```
nType =
    nFieldType == 0     //text field
    ?
        dbText
    :
    nFieldType == 1     //integer field
    ?
        dbLong
    :                   //Must be a floating point
        dbDouble;
```

4. Add a final statement to the inner while loop to call **CDaoTableDef::CreateField** with the appropriate information.

```
daoTblDef.CreateField( strFieldName, nType,
        nTextFieldLength );
```

➢ **Append the table to the database**

1. Between the end braces of the inner and outer while loops, call **CDaoTableDef::Append**, and **CDaoTableDef::Close**. These calls add the tabledef to the database, and close the tabledef for the next iteration.

```
daoTblDef.Append( );//Add table to the database
daoTblDef.Close( );     //Close the tabledef
```

2. Immediately following the outer **while** loop, before the closing brace of the **try** block, close the database object. Then call **CDocument::SetModifiedFlag** to indicate that the document is not modified.

```
m_daoDB.Close( );   //Close the database
SetModifiedFlag( FALSE );
```

3. The **catch** block has an embedded **try** block. Within the embedded **try** block, close the database object. Next, test to determine whether the **CDaoException** error code is 3204, which means the database already exists. If the error is NOT 3204, Use **CFile::Remove** to erase the database file.

```
m_daoDB.Close( );   //Close the database

//Database already exists error
if ( 3204 != ex->m_pErrorInfo->m_lErrorCode )
    CFile::Remove( dlg.GetFileName( ) );
```

➢ **Build and test your application**

1. To test the application, run it and use the menu or the tool bar to create a new database. Add one or more tables to the database.
2. For each table, add one or more fields, with each field being of type text, integer, or floating point.
3. After creating the database, you can use Access to verify the database schema.

The completed code for this exercise is in \Labs\C10\Lab02\Ex01.