Recordset: Scrolling (ODBC)

Article • 08/02/2021

This topic applies to the MFC ODBC classes.

After you open a recordset, you need to access the records to display values, do calculations, generate reports, and so on. Scrolling lets you move from record to record within your recordset.

This topic explains:

- How to scroll from one record to another in a recordset.
- Under what circumstances scrolling is and is not supported.

Scrolling from One Record to Another

Class `CRecordset` provides the `Move` member functions for scrolling within a recordset. These functions move the current record by rowsets. If you have implemented bulk row fetching, a `Move` operation repositions the recordset by the size of the rowset. If you have not implemented bulk row fetching, a call to a `Move` function repositions the recordset by one record each time. For more information about bulk row fetching, see Recordset: Fetching Records in Bulk (ODBC).

> **Note**
>
> When moving through a recordset, deleted records might not be skipped. For more information, see the **IsDeleted** member function.

In addition to the `Move` functions, `CRecordset` provides member functions for checking whether you have scrolled past the end or ahead of the beginning of your recordset.

To determine whether scrolling is possible in your recordset, call the `CanScroll` member function.

To scroll

1. Forward one record or one rowset: call the MoveNext member function.
2. Backward one record or one rowset: call the MovePrev member function.
3. To the first record in the recordset: call the MoveFirst member function.
4. To the last record in the recordset or to the last rowset: call the MoveLast member function.
5. *N* records relative to the current position: call the Move member function.

To test for the end or the beginning of the recordset

1. Have you scrolled past the last record? Call the IsEOF member function.
2. Have you scrolled ahead of the first record (moving backward)? Call the IsBOF member function.

The following code example uses `IsBOF` and `IsEOF` to detect the limits of a recordset when scrolling in either direction.

```
// Open a recordset; first record is current
CCustSet rsCustSet( NULL );
rsCustSet.Open( );

if( rsCustSet.IsBOF( ) )
    return;
    // The recordset is empty

// Scroll to the end of the recordset, past
// the last record, so no record is current
while ( !rsCustSet.IsEOF( ) )
    rsCustSet.MoveNext( );

// Move to the last record
rsCustSet.MoveLast( );

// Scroll to beginning of the recordset, before
// the first record, so no record is current
while( !rsCustSet.IsBOF( ) )
    rsCustSet.MovePrev( );

// First record is current again
rsCustSet.MoveFirst( );
```

`IsEOF` returns a nonzero value if the recordset is positioned past the last record. `IsBOF` returns a nonzero value if the recordset is positioned ahead of the first record (before all records). In either case, there is no current record to operate on. If you call `MovePrev` when `IsBOF` is already TRUE or call `MoveNext` when `IsEOF` is already TRUE, the framework throws a `CDBException`. You can also use `IsBOF` and `IsEOF` to check for an empty recordset.

For more information about recordset navigation, see Recordset: Bookmarks and Absolute Positions (ODBC).

When Scrolling Is Supported

As originally designed, SQL provided only forward scrolling, but ODBC extends scrolling capabilities. The available level of support for scrolling depends on the ODBC drivers your application works with, your driver's ODBC API conformance level, and whether the ODBC Cursor Library is loaded into memory. For more information, see ODBC and ODBC: The ODBC Cursor Library.

> **Tip**
>
> You can control whether the cursor library is used. See the *bUseCursorLib* and *dwOptions* parameters to **CDatabase::Open**.

> **Note**
>
> Unlike the MFC DAO classes, the MFC ODBC classes do not provide a set of `Find` functions for locating the next (or previous) record that meets specified criteria.

See also

Recordset (ODBC)
CRecordset::CanScroll
CRecordset::CheckRowsetError
Recordset: Filtering Records (ODBC)