Recordset: Requerying a Recordset (ODBC)

Article • 08/02/2021

This topic applies to the MFC ODBC classes.

This topic explains how you can use a recordset object to requery (that is, refresh) itself from the database and when you might want to do that with the Requery member function.

The principal reasons for requerying a recordset are to:

- Bring the recordset up to date with respect to records added by you or by other users and records deleted by other users (those you delete are already reflected in the recordset).
- Refresh the recordset based on changing parameter values.

Bringing the Recordset Up to Date

Frequently, you will want to requery your recordset object to bring it up to date. In a multiuser database environment, other users can make changes to the data during the life of your recordset. For more information about when your recordset reflects changes made by other users and when other users' recordsets reflect your changes, see Recordset: How Recordsets Update Records (ODBC) and Dynaset.

Requerying Based on New Parameters

Another frequent — and equally important — use of Requery is to select a new set of records based on changing parameter values.

> **Tip**
>
> Query speed is probably significantly faster if you call `Requery` with changing parameter values than if you call `Open` again.

Requerying Dynasets vs. Snapshots

Because dynasets are meant to present a set of records with dynamic up-to-date data, you want to requery dynasets often if you want to reflect other users' additions. Snapshots, on the other hand, are useful because you can safely rely on their static contents while you prepare reports, calculate totals, and so on. Still, you might sometimes want to requery a snapshot as well. In a multiuser environment, snapshot data might lose synchronization with the data source as other users change the database.

To requery a recordset object

1. Call the Requery member function of the object.

Alternatively, you can close and reopen the original recordset. In either case, the new recordset represents the current state of the data source.

For an example, see Record Views: Filling a List Box from a Second Recordset.

> **Tip**
>
> To optimize `Requery` performance, avoid changing the recordset's **filter** or **sort**. Change only the parameter value before calling `Requery`.

If the `Requery` call fails, you can retry the call; otherwise, your application should terminate gracefully. A call to `Requery` or `Open` might fail for any of a number of reasons. Perhaps a network error occurs; or, during the call, after the existing data is released but before the new data is obtained, another user might get exclusive access; or the table on which your recordset depends could be deleted.

See also

Recordset (ODBC)
Recordset: Dynamically Binding Data Columns (ODBC)
Recordset: Creating and Closing Recordsets (ODBC)