# Recordset (ODBC)

Article • 08/02/2021

This topic applies to the MFC ODBC classes.

A CRecordset object represents a set of records selected from a data source. The records can be from:

- A table.
- A query.
- A stored procedure that accesses one or more tables.

An example of a recordset based on a table is "all customers," which accesses a Customer table. An example of a query is "all invoices for Joe Smith." An example of a recordset based on a stored procedure (sometimes called a predefined query) is "all of the delinquent accounts," which invokes a stored procedure in the back-end database. A recordset can join two or more tables from the same data source, but not tables from different data sources.

**Note**

Some ODBC drivers support views of the database. A view in this sense is a query originally created with the SQL `CREATE VIEW` statement.

## Recordset Capabilities

All recordset objects share the following capabilities:

- If the data source is not read-only, you can specify that your recordset be updatable, appendable, or read-only. If the recordset is updateable, you can choose either pessimistic or optimistic locking methods, provided the driver supplies the appropriate locking support. If the data source is read-only, the recordset will be read-only.
- You can call member functions to scroll through the selected records.

- You can filter the records to constrain which records are selected from those available.
- You can sort the records in ascending or descending order, based on one or more columns.
- You can parameterize the recordset to qualify the recordset selection at run time.

## Snapshots and Dynasets

There are two principal types of recordsets: snapshots and dynasets. Both are supported by class `CRecordset`. Each shares the common characteristics of all recordsets, but each also extends the common functionality in its own specialized way. Snapshots provide a static view of the data and are useful for reports and other situations in which you want a view of the data as it existed at a particular time. Dynasets are useful when you want updates made by other users to be visible in the recordset without having to requery or refresh the recordset. Snapshots and dynasets can be updateable or read-only. To reflect records added or deleted by other users, call CRecordset::Requery.

`CRecordset` also allows for two other types of recordsets: dynamic recordsets and forward-only recordsets. Dynamic recordsets are similar to dynasets; however, dynamic recordsets reflect any records added or deleted without calling `CRecordset::Requery`. For this reason, dynamic recordsets are generally expensive with respect to processing time on the DBMS, and many ODBC drivers do not support them. In contrast, forward-only recordsets provide the most efficient method of data access for recordsets that do not require updates or backward scrolling. For example, you might use a forward-only recordset to migrate data from one data source to another, where you only need to move through the data in a forward direction. To use a forward-only recordset, you must do both of the following:

- Pass the option `CRecordset::forwardOnly` as the *nOpenType* parameter of the Open member function.
- Specify `CRecordset::readOnly` in the *dwOptions* parameter of `Open`.

**Note**

For information about ODBC driver requirements for dynaset support, see

**ODBC**. For a list of ODBC drivers included in this version of Visual C++ and for information about obtaining additional drivers, see **ODBC Driver List**.

## Your Recordsets

For every distinct table, view, or stored procedure that you want to access, you typically define a class derived from `CRecordset`. (The exception is a database join, in which one recordset represents columns from two or more tables.) When you derive a recordset class, you enable the record field exchange (RFX) mechanism or the bulk record field exchange (Bulk RFX) mechanism, which are similar to the dialog data exchange (DDX) mechanism. RFX and Bulk RFX simplify the transfer of data from the data source into your recordset; RFX additionally transfers data from your recordset to the data source. For more information, see Record Field Exchange (RFX) and Recordset: Fetching Records in Bulk (ODBC).

A recordset object gives you access to all the selected records. You scroll through the multiple selected records using `CRecordset` member functions, such as `MoveNext` and `MovePrev`. At the same time, a recordset object represents only one of the selected records, the current record. You can examine the fields of the current record by declaring recordset class member variables that correspond to columns of the table or of the records that result from the database query. For information about recordset data members, see Recordset: Architecture (ODBC).

The following topics explain the details of using recordset objects. The topics are listed in functional categories and a natural browse order to permit sequential reading.

### Topics about the mechanics of opening, reading, and closing recordsets

- Recordset: Architecture (ODBC)
- Recordset: Declaring a Class for a Table (ODBC)
- Recordset: Creating and Closing Recordsets (ODBC)
- Recordset: Scrolling (ODBC)
- Recordset: Bookmarks and Absolute Positions (ODBC)

- Recordset: Filtering Records (ODBC)
- Recordset: Sorting Records (ODBC)
- Recordset: Parameterizing a Recordset (ODBC)

## Topics about the mechanics of modifying recordsets

- Recordset: Adding, Updating, and Deleting Records (ODBC)
- Recordset: Locking Records (ODBC)
- Recordset: Requerying a Recordset (ODBC)

## Topics about somewhat more advanced techniques

- Recordset: Performing a Join (ODBC)
- Recordset: Declaring a Class for a Predefined Query (ODBC)
- Recordset: Dynamically Binding Data Columns (ODBC)
- Recordset: Fetching Records in Bulk (ODBC)
- Recordset: Working with Large Data Items (ODBC)
- Recordset: Obtaining SUMs and Other Aggregate Results (ODBC)

## Topics about how recordsets work

- Recordset: How Recordsets Select Records (ODBC)
- Recordset: How Recordsets Update Records (ODBC)

## See also

Open Database Connectivity (ODBC)
MFC ODBC Consume
Transaction (ODBC)