

Harbour Guide

ADS Overview

Advantage Database Server RDD

Description

RDDADS is an RDD for the Advantage Database Server, an xBase data server by Extended Systems <www.advantagedatabase.com>. The RDD was written by Alexander Kresin <alex@belacy.belgorod.su> Additional code and documentation was added by Brian Hays <bhays@abacuslaw.com>.

Your Harbour application can access a remote database server for a true client/server architecture, or it can use the "local server" ADSLOC32.DLL for stand-alone or even small network installations.

For using this RDD you need to have:

ACE32.DLL (Advantage Client Engine),
AXCWS32.DLL (communication layer for remote server) or
ADSLOC32.DLL (local server)

You need also to create ace32.lib with the help of implib.exe: implib
ace32.lib ace32.dll

Then build rddads.lib using make_b32.bat or make_vc.bat.

For building executables don't forget to include the ace32.lib and rddads.lib in the make file or link script.

You also need to include in your PRG file following lines:

```
REQUEST _ADS  
rddRegister( "ADS", 1 )  
rddsetdefault( "ADS" )
```

By default RDDADS is tuned for remote server and cdx indexes. To change this you may use these commands defined in ads.ch:

```
SET SERVER LOCAL SET SERVER REMOTE
```

```
SET FILETYPE TO NTX SET FILETYPE TO ADT SET FILETYPE TO CDX
```

or functions AdsSetServerType(), AdsSetFileType(). See the header file ADS.CH for details.

Note that the default local server (ADSLOC32.DLL) is useable for file sharing on a small network. The default DLL is limited to 5 users, but an unlimited version is available from Extended Systems.

MAX OPEN TABLES: The server (even local) has its own setting for Max Tables allowed open. For the Local Server, it can be set in ADSLOCAL.CFG. The default is only 50! For the Windows Remote Servers, use the Configuration Utility, or increase the setting for the TABLES configuration value in the Advantage Database Server configuration registry key using the Registry Editor. For NetWare, edit the configuration file ADS.CFG.

See ACE.HLP under ADSLOCAL.CFG, or the Advantage Error Guide for error 7005.

Every attempt has been made to make the rdd compliant with the standard dbfc dx rdd at the .PRG level. One important difference is the handling of **structural indexes**. **ACE will always automatically open an index with the same name as the data file.** There is no way to turn this feature off.

Be sure to use the command SET DEFAULT TO (cDir) and not its equivalent Set() function call. The Set() function will not make the call to ADS to change its internal setting, but the command will. The same is true for DATEFORMAT, DELETE, and EPOCH.

For programmers who are already familiar with the ACE engine, this also means there are some differences between the RDDADS in Harbour and the parallel ACE documentation.

1) In ACE, skipping backwards to BOF goes to the phantom record and sets the record number to 0. In RDDADS, the record pointer stays at the Top record and only the BOF flag is set to True.

2) In RDDADS, a filter expression can be used that may not be valid on the

server (because of references to public variables or User-Defined Functions). In these cases, all data will come back from the server but will be filtered by the application running on the client. These situations lose the benefits of having a data server and should be avoided, but they will function as they would in a Clipper program.

One problem with this scenario is that index key counting functions that are supposed to give an accurate count respecting the filter (e.g. `dbOrderInfo(DBOI_KEYCOUNT)`) will return the values the Server knows about, so the counts will be inaccurate.

3) When setting a relation, the expression must be one that can be evaluated by the Advantage Expression Engine. UDFs will fail.

ADSBlob2File()

Write a Binary (memo) field's contents to a file

Syntax

```
ADSBlob2File(cFileName, cFieldName) --> lSuccess
```

Arguments

<cFileName> File to create. If it already exists, it will be overwritten on success and destroyed on error.

<cFieldName> Field in the current workarea that contains binary data.

Returns

<lSuccess> True if the file is successfully written.

Description

See ACE.HLP for full details about the Advantage Database Server.
ADSBlob2File() is a wrapper for AdsBinaryToFile.

Status

Ready

Compliance

Harbour extension

Platforms

Windows 32-bit only

Files

Library is RddAds Header is ads.ch

See Also:

[ADSFile2Blob\(\)](#)

ADSFile2Blob()

Save a Binary file to a field

Syntax

```
ADSFile2Blob(cFileName, cFieldName, <nBinaryType>) --> lSuccess
```

Arguments

<cFileName> File to read. Can be in UNC format. A common example is an image file.

<cFieldName> Field in the current workarea to contain the binary data.

<nBinaryType> Either ADS_BINARY (the default) or ADS_IMAGE. This parameter is for fields in DBF files. ADT tables cannot store binary and image data in standard character memo fields (they have specific field types for that).

Returns

<lSuccess> True if the file is successfully written.

Description

See ACE.HLP for full details about the Advantage Database Server. ADSFile2Blob() is a wrapper for AdsFileToBinary. Use of this function is illegal in an ADS transaction.

Status

Ready

Compliance

Harbour extension

Platforms

Windows 32-bit only

Files

Library is RddAds Header is ads.ch

See Also:

[ADSBlob2File\(\)](#)

ADSClearAOF()

Clears an Advantage Optimized Filter in the current workarea.

Syntax

```
ADSClearAOF()
```

Arguments

Returns

Description

See ACE.HLP for full details about the Advantage Database Server.

Status

Ready

Compliance

Harbour extension

Platforms

Windows 32-bit

Files

Library is RddAds Header is ads.ch

See Also:

[ADSCustomizeAOF\(\)](#)

[ADSEvalAOF\(\)](#)

[ADSGetAOF\(\)](#)

[ADSGetAOFoptLevel\(\)](#)

[ADSIsRecordInAOF\(\)](#)

[ADSRefreshAOF\(\)](#)

ADSCustomizeAOF()

Add or remove records from an existing AOF

Syntax

```
ADSCustomizeAOF( [<nRecno | aRecNos>] [, <nType>] ) --> nSuccess
```

Arguments

<nRecno | aRecNos> Can be either a single record number or an array of record numbers to add or delete from the AOF. If omitted, defaults to the current record.

<nType> The type of operation:

ADS_AOF_ADD_RECORD	Add the record to the AOF (set the bit). This is the default operation.
ADS_AOF_REMOVE_RECORD	Remove the record from the AOF (clear the bit).
ADS_AOF_TOGGLE_RECORD	Switch the record into or out of the AOF.

Returns

<nError> ADS error code, or 0 for success.

Description

An Advantage Optimized Filter (AOF) consists of a bitmap of the records in the database. If bit 5 is on, record 5 is considered a visible record. If bit 5 is off, record 5 is not visible. It does not "pass the test". Initially, the bits are set by the Server according to a filter expression from SET FILTER TO or adsSetAOF(). But by using ADSCustomizeAOF() you can add or remove records at will from the visible set. This is useful for tagging records or for refining a result set after the data has been retrieved from the server.

The maximum number of records that can be customized in a single call is 16,383, so <aRecNos> must not be longer than this.

Calls to AdsCustomizeAOF must be made after an application has created a filter with a call to AdsSetAOF. To create a completely empty record set (to which records can be added with calls to AdsCustomizeAOF), use ".F." as the filter expression given to AdsSetAOF. To create a completely full record set (from which records can be removed), use ".T." as the filter expression.

WARNING: Always start with a FULLY optimized AOF! If an application must use a filter expression that is not fully optimized as the starting point for customization, the ADS_RESOLVE_IMMEDIATE option should be used with the call to AdsSetAOF. Otherwise, the dynamic filter resolution that occurs on the server will automatically remove records that have been added through the AdsCustomizeAOF calls. The filter expressions ".T." and ".F." both result in fully optimized AOFs regardless of available indexes.

See ACE.HLP for full details about the Advantage Database Server.

Status

Ready

Compliance

Harbour extension

Platforms

Windows 32-bit

Files

Library is RddAds Header is ads.ch

See Also:

[ADSClearAOF\(\)](#)

[ADSEvalAOF\(\)](#)

[ADSGetAOF\(\)](#)
[ADSGetAOFoptLevel\(\)](#)
[ADSIIsRecordInAOF\(\)](#)
[ADSRefreshAOF\(\)](#)

ADSEvalAOF()

Evaluate a filter expression to determine its optimization level

Syntax

```
ADSEvalAOF(<cFilter>) --> nOptimizationLevel
```

Arguments

<cFilter> Expression to test.

Returns

ADS_OPTIMIZED_NONE. IMPORTANT NOTE: These values are NOT the same as those returned by dbOrderInfo().

Description

See ACE.HLP for full details about the Advantage Database Server.

Status

Ready

Compliance

Harbour extension

Platforms

Windows 32-bit

Files

Library is RddAds Header is ads.ch

See Also:

[ADSClearAOF\(\)](#)
[ADSGetAOF\(\)](#)
[ADSGetAOFoptLevel\(\)](#)
[ADSIsRecordInAOF\(\)](#)
[ADSRefreshAOF\(\)](#)

ADSGetAOFoptLevel()

Returns optimization level of the current AOF filter

Syntax

```
ADSGetAOFoptLevel() --> nOptimizationLevel
```

Arguments

Returns

ADS_OPTIMIZED_NONE.

Description

See ACE.HLP for full details about the Advantage Database Server.

Status

Ready

Compliance

Harbour extension

Platforms

Windows 32-bit

Files

Library is RddAds Header is ads.ch

See Also:

[ADSClearAOF\(\)](#)

[ADSGetAOF\(\)](#)

[ADSIsRecordInAOF\(\)](#)

[ADSRefreshAOF\(\)](#)

ADSGetAOF()

Retrieve the filter expression used in the call to AdsSetAOF

Syntax

```
ADSGetAOF() --> cFilter
```

Arguments

Returns

<cFilter> The filter expression used in the call to AdsSetAOF.

Description

See ACE.HLP for full details about the Advantage Database Server.

Status

Ready

Compliance

Harbour extension

Platforms

Windows 32-bit

Files

Library is RddAds Header is ads.ch

See Also:

[ADSClearAOF\(\)](#)

[ADSGetAOFoptLevel\(\)](#)

[ADSIsRecordInAOF\(\)](#)

[ADSRefreshAOF\(\)](#)

ADSGetAOFnoOpt()

Return the non-optimized portion of the current filter expression

Syntax

```
ADSGetAOFnoOpt() --> cFilterFragment
```

Arguments

Returns

<cFilterFragment> If an AOF filter expression is not fully optimizable, the non-optimizable part of the expression can be retrieved with this function.

Description

See ACE.HLP for full details about the Advantage Database Server.

Status

Ready

Compliance

Harbour extension

Platforms

Windows 32-bit

Files

Library is RddAds Header is ads.ch

See Also:

[ADSClearAOF\(\)](#)

[ADSIsRecordInAOF\(\)](#)

[ADSRefreshAOF\(\)](#)

ADSRefreshAOF()

Update the filter snapshot

Syntax

```
ADSRefreshAOF()
```

Arguments

Returns

Description

See ACE.HLP for full details about the Advantage Database Server. If record updates occur after an AOF is set, the updated records may or may not be valid records for the filter. ADSRefreshAOF() re-evaluates the data to include or exclude changed records.

Status

Ready

Compliance

Harbour extension

Platforms

Windows 32-bit

Files

Library is RddAds Header is ads.ch

See Also:

[ADSClearAOF\(\)](#)

[ADSIIsRecordInAOF\(\)](#)

[ADSSetAOF\(\)](#)

ADSSetAOF()

Create an Advantage Optimized Filter

Syntax

```
ADSSetAOF( <cFilter> [, <nResolveOption>] ) --> lSuccess
```

Arguments

<cFilter> Filter expression to set.

<nResolveOption> Option to indicate how the filter should be resolved in the event that the expression cannot be fully optimized. Options are defined in ads.ch: ADS_RESOLVE_IMMEDIATE, ADS_RESOLVE_DYNAMIC.

Returns

<lSuccess> True if AOF is created.

Description

See ACE.HLP for full details about the Advantage Database Server.

Status

Ready

Compliance

Harbour extension

Platforms

Windows 32-bit

Files

Library is RddAds Header is ads.ch

See Also:

[ADSClearAOF\(\)](#)

[ADSIsRecordInAOF\(\)](#)

[ADSRefreshAOF\(\)](#)

ADSIIsRecordInAOF()

Determine if a record is in the current AOF

Syntax

```
ADSIIsRecordInAOF( [<nRecNo>] ) --> lSatisfiesFilter
```

Arguments

<nRecNo> Record number to test. Default is current record.

Returns

Description

See ACE.HLP for full details about the Advantage Database Server.

Status

Ready

Compliance

Harbour extension

Platforms

Windows 32-bit

Files

Library is RddAds Header is ads.ch

See Also:

[ADSClearAOF\(\)](#)

[ADSRefreshAOF\(\)](#)

ADSGetRelKeyPos()

Estimated key position of current record within the specified index

Syntax

```
ADSGetRelKeyPos([<xTag>]) --> nKeyPos
```

Arguments

<xTag> Index to use. Default is current index.

Returns

position of the current key in the indicated index order. The value returned is between 0.0 and 1.0, inclusive. If there are scopes set on the index order, the position returned is relative to the visible records.

Description

See ACE.HLP for full details about the Advantage Database Server.

Status

Ready

Compliance

Harbour extension

Platforms

Windows 32-bit

Files

Library is RddAds Header is ads.ch

See Also:

[ADSKeyNo\(\)](#)

[ADSKeyCount\(\)](#)

ADSKeyCount()
Retrieve the number of keys in a specified index

Syntax

ADSKeyCount([<xTag>], <cIgnoredIndexFile>, [<nFilterOption>]) --> nKeyCount

Arguments

- <xTag>** Numeric order number OR index tag name. Default is current index.
- <cIgnoredIndexFile>** This parameter is not processed. In other Harbour RDDs, the second parameter to "ordKeyCount" takes a second argument to identify a particular Index File in cases where two files are open that contain orders with the same name. The ADS driver does not support this and will select the first order with the requested name. To stay consistent with other RDDs, therefore(), the second parameter is reserved and the <nFilterOption> is passed as a third parameter.
- <nFilterOption>** Indicates if filters and/or scopes are to be respected if set.

ADS_RESPECTFILTERS	Respect filters and scopes
ADS_IGNOREFILTERS	Ignore filters and scopes
ADS_RESPECTSCOPES	Respect scopes only

Returns

<nKeyCount> The number of keys within the current index.

Description

See ACE.HLP for full details about the Advantage Database Server.

Status

Ready

Compliance

Harbour extension

Platforms

Windows 32-bit

Files

Library is RddAds Header is ads.ch

See Also:

- [ADSKeyNo\(\)](#)
- [ADSGetRelKeyPos\(\)](#)

ADSKeyNo()

Get the logical key number of the current record in the given index

Syntax

```
ADSKeyNo([<xTag>], [<nFilterOption>]) --> nKeyNo
```

Arguments

<xTag> Numeric order number OR index tag name. Default is current index.

<nFilterOption> Indicates if filters and/or scopes are to be respected if set.

ADS_RESPECTFILTERS	
ADS_IGNOREFILTERS	
ADS_RESPECTSCOPES	

Returns

<nKeyNo> The logical key number of the current record in the given index.

Description

See ACE.HLP for full details about the Advantage Database Wrapper for AdsGetKeyNum.

This function may be slow on a large database with ADS_RESPECTFILTERS set because it walks through the keys to get the current position. Compare to ADSGetRelKeyPos().

Status

Ready

Compliance

Harbour extension

Platforms

Windows 32-bit

Files

Library is RddAds Header is ads.ch

See Also:

[ADSKeyCount\(\)](#)

[ADSGetRelKeyPos\(\)](#)

ADSLocking()

Turns on/off the Advantage proprietary locking mode

Syntax

```
ADSLocking( <lMode> ) --> lPriorSetting
```

Arguments

<lMode> .T. to use the Advantage proprietary locking mode (this is the default setting if a remote server is used) or pass .F. to use "compatibility" locking.

Returns

<lPriorSetting> .T. if prior setting was for the proprietary mode.

Description

See ACE.HLP for full details about the Advantage Database Server. The Advantage Database Server has a fast Proprietary locking mode that is more efficient than traditional network locking. It is only available when using the remote server (not the local server).

If a file is opened in the proprietary mode, other applications cannot open it in a "write" mode. So if non-Advantage applications need concurrent access to the data files, use the Compatibility locking mode by calling ADSLocking(.F.).

ADSLocking() is a Get/Set function for the locking mode. It affects files at the time they are opened. So when a data file is opened, the current setting is used for that file until it is closed. Different files can have different locking modes by changing the setting before opening a second file.

Status

Ready

Compliance

Harbour extension

Platforms

Windows 32-bit

Files

Library is RddAds Header is ads.ch

See Also:

[ADSRightsCheck\(\)](#)

ADSRightsCheck()

Sets the "rights checking" setting for opening files

Syntax

```
ADSRightsCheck( <lMode> ) --> lPriorSetting
```

Arguments

<lMode> .T. to check rights upon opening data files (the default), or .F. to ignore rights

Returns

Description

See ACE.HLP for full details about the Advantage Database Server.

ADSRightsCheck() is a Get/Set function for the "rights checking" mode. If the setting is .T. when a file is opened, then the Advantage Database Server will use the rights of the connected user when opening the file. If the user does not have rights to the directory or server, then the open call will fail.

If the setting is .F., then the ADS will ignore the connected user's rights and open the file regardless. This lets you allow only Advantage-based applications to access specific data.

Status

Ready

Compliance

Harbour extension

Platforms

Windows 32-bit

Files

Library is RddAds Header is ads.ch

See Also:

[ADSLocking\(\)](#)