Package: adbi (via r-universe)

October 1, 2026

Title 'DBI' Compliant Database Access Using 'ADBC'

Version 0.1.4

Description In order to make Arrow Database Connectivity ('ADBC' <https://arrow.apache.org/adbc/>) accessible from R, an interface compliant with the 'DBI' package is provided, using driver back-ends that are implemented in the 'adbcdrivermanager' framework. This enables interacting with database systems using the Arrow data format, thereby offering an efficient alternative to 'ODBC' for analytical applications.

License LGPL (>= 2.1)

URL https://adbi.r-dbi.org, https://github.com/r-dbi/adbi, https://arrow.apache.org/adbc/

BugReports https://github.com/r-dbi/adbi/issues

Depends R (>= 4.2.0)

Imports DBI (>= 1.2.0), methods, adbcdrivermanager (>= 0.19.0), nanoarrow (>= 0.3.0), utils

Suggests testthat, covr, DBItest (>= 1.8.3), adbcsqlite (>= 0.11.0.1), withr, bit64, arrow

Config/Needs/website r-dbi/dbitemplate

Encoding UTF-8

Roxygen list(markdown = TRUE)

Collate 'adbi-package.R' 'AdbiDriver.R' 'AdbiConnection.R' 'AdbiResult.R' 'AdbiResultArrow.R' 'dbAppendTable_AdbiConnection_character_data.frame.R' 'dbBegin_AdbiConnection.R' 'dbBindArrow_AdbiResult.R' 'dbBindArrow_AdbiResultArrow.R' 'dbBind_AdbiResult.R' 'dbBind_AdbiResultArrow.R' 'dbClearResult_AdbiResult.R' 'dbClearResult_AdbiResultArrow.R' 'dbColumnInfo_AdbiResult.R' 'dbColumnInfo_AdbiResultArrow.R' 'dbCommit_AdbiConnection.R' 'dbConnect_AdbiDriver.R' 'dbDataType_AdbiConnection.R' 'dbDataType_AdbiDriver.R' 'dbDisconnect_AdbiConnection.R'

'dbExistsTable_AdbiConnection_Id.R'
'dbExistsTable_AdbiConnection_SQL.R'
'dbExistsTable_AdbiConnection_character.R'
'dbFetchArrowChunk_AdbiResultArrow.R'
'dbFetchArrow_AdbiResultArrow.R' 'dbFetch_AdbiResult.R'
'dbGetInfo_AdbiConnection.R' 'dbGetInfo_AdbiDriver.R'
'dbGetRowCount_AdbiResult.R' 'dbGetRowCount_AdbiResultArrow.R'
'dbGetRowsAffected_AdbiResult.R'
'dbGetRowsAffected_AdbiResultArrow.R'
'dbGetStatement_AdbiResult.R'
'dbGetStatement_AdbiResultArrow.R'
'dbHasCompleted_AdbiResult.R'
'dbHasCompleted_AdbiResultArrow.R' 'dbIsValid_AdbiConnection.R'
'dbIsValid_AdbiDriver.R' 'dbIsValid_AdbiResult.R'
'dbIsValid_AdbiResultArrow.R'
'dbListFields_AdbiConnection_Id.R'
'dbListFields_AdbiConnection_SQL.R'
'dbListFields_AdbiConnection_character.R'
'dbListTables_AdbiConnection.R'
'dbQuoteIdentifier_AdbiConnection_character.R'
'dbQuoteLiteral_AdbiConnection_character.R'
'dbQuoteString_AdbiConnection_character.R'
'dbRemoveTable_AdbiConnection_character.R'
'dbRemoveTable_AdbiConnection_Id.R'
'dbRollback_AdbiConnection.R'
'dbSendQueryArrow_AdbiConnection.R'
'dbSendQuery_AdbiConnection_character.R'
'dbSendStatement_AdbiConnection_character.R'
'dbUnquoteIdentifier_AdbiConnection.R'
'dbWriteTable_AdbiConnection_Id_data.frame.R'
'dbWriteTable_AdbiConnection_SQL_data.frame.R'
'dbWriteTable_AdbiConnection_character_data.frame.R' 'export.R'
'show_AdbiConnection.R' 'show_AdbiDriver.R' 'show_AdbiResult.R'
'show_AdbiResultArrow.R' 'utils.R'

Config/roxygen2/version 8.1.0

Config/pak/sysreqs libzstd-dev

Repository https://cynkra.r-universe.dev

Date/Publication 2026-10-01 14:45:28 UTC

RemoteUrl https://github.com/r-dbi/adbi

RemoteRef HEAD

RemoteSha ac5d3294597c453c8cb1c278b9536cd9afb3eae2

Contents

adbi 3
dbFetch_AdbiResult 4

dbSendQueryArrow_AdbiConnection 5

Index **8**

---

adbi *Adbi driver*

Description

In order to open a database connection, `DBI::dbConnect()` dispatches on a driver object, which can be instantiated by calling `adbi()`.

Usage

```
adbi(driver = NA_character_, pkg = NA_character_)

## S4 method for signature 'AdbiDriver'
dbConnect(drv, ..., bigint = NULL)

## S4 method for signature 'AdbiConnection'
dbDisconnect(conn, force = getOption("adbi.force_close_results", FALSE), ...)
```

Arguments

`driver`	An ADBC driver object, a function returning one, an ADBC Driver Manager driver name or manifest path, or (when `pkg` is supplied) the name of a driver function in that package. See Details for more information.
`pkg`	An R package containing a driver function, or `NA` if no R package is explicitly specified.
`drv`	An object that inherits from DBI::DBIDriver, or an existing DBI::DBIConnection object (in order to clone an existing connection).
`...`	Extra arguments passed to `DBI::dbConnect()` are forwarded to `adbcdrivermanager::adbc_database_`
`bigint`	The R type that 64-bit integer types should be mapped to. The default, `"integer-strict"`, maps them to `integer` and fails with an error for values outside its range, whereas `"integer"` silently returns `NA` for them. Likewise, `"numeric-strict"` maps them to `numeric` and fails for values beyond 2^53 in magnitude, which a double may not represent exactly, whereas `"numeric"` silently rounds them to the nearest double. Both `"character"` and `"integer64"` are lossless, returning decimal strings and bit64::integer64 values respectively. The latter requires the bit64 package.
`conn`	A DBI::DBIConnection object, as returned by `DBI::dbConnect()`.
`force`	Close open results when disconnecting

Details

To specify the type of ADBC driver, `adbi()` accepts as `driver` argument

- an object inheriting from `adbc_driver`,
- a function that can be evaluated with no arguments and returns an object inheriting from `adbc_driver`,
- an ADBC Driver Manager driver name or manifest path.

Use `pkg` to load a driver provided by an R package. By default, the driver function has the same name as the package; supply a character `driver` to use a different function. For example, `adbi(pkg = "adbcsqlite")` calls `adbcsqlite::adbcsqlite()`, while `adbi("adbc_driver_monkey", pkg = "adbcdrivermanager")` calls `adbcdrivermanager::adbc_driver_monkey()`.

For compatibility, a character `driver` that names an installed R package with a same-named driver function still selects that function, so `adbi("adbcsqlite")` is equivalent to `adbi(pkg = "adbcsqlite")`. Strings of the form `pkg::fun` are deprecated; use `adbi("fun", pkg = "pkg")` instead.

As default, an `adbcdrivermanager::adbc_driver_monkey()` object is created.

Value

A connection object (S4 class `AdbiConnection`, inheriting from DBI::DBIConnection) is returned by `DBI::dbConnect()`, while `DBI::dbDisconnect()` returns `TRUE` invisibly.

Examples

```
adbi()
## Not run: adbi("sqlite")
if (requireNamespace("adbcsqlite")) {
  adbi(pkg = "adbcsqlite")
}
library(DBI)
con <- dbConnect(adbi())
dbIsValid(con)
dbDisconnect(con)
dbIsValid(con)
```

`dbFetch_AdbiResult` *Fetch result sets*

Description

When fetching results using `DBI::dbFetch()`, the argument `n` can be specified to control chunk size per fetching operation. The default value of `-1` corresponds to retrieving the entire result set at once, while a positive integer will try returning as many rows (as long as `n` does not exceed the available number of rows), in line with standard DBI expectations. As data transfer is mediated by Arrow data structures, which are retrieved as array chunks, the underlying chunk size can be used by passing an `n` value `NA`.

Usage

```
## S4 method for signature 'AdbiResult'
dbFetch(res, n = -1, ...)
```

Arguments

res	An object inheriting from DBI::DBIResult, created by DBI::dbSendQuery().
n	maximum number of records to retrieve per fetch. Use n = -1 or n = Inf to retrieve all pending records. Some implementations may recognize other special values.
...	Other arguments passed on to methods.

Value

A data.frame with the requested number of rows (or zero rows if DBI::dbFetch() is called on a result set with no more remaining rows).

Examples

```
if (requireNamespace("adbcsqlite")) {
  library(DBI)
  con <- dbConnect(adbi::adbi(pkg = "adbcsqlite"), uri = ":memory:")
  dbWriteTable(con, "swiss", swiss)
  res <- dbSendQuery(con, "SELECT * from swiss WHERE Agriculture < 30")
  dbFetch(res)
  dbClearResult(res)
  dbDisconnect(con)
}
```

---

dbSendQueryArrow_AdbiConnection

Create result sets

Description

Creating result sets using DBI::dbSendQuery() (and by extension using DBI::dbGetQuery()) mostly follows DBI specification. One way where adbi deviates from DBI mechanisms is how the bigint setting is not only per connection, but the per-connection setting can be overridden on a result set basis. As default, the connection setting is applied, but passing one of the accepted values as bigint when creating a result set will subsequently use that setting for all fetches using this result set.

Usage

```
## S4 method for signature 'AdbiConnection'
dbSendQueryArrow(
  conn,
  statement,
  ...,
  params = NULL,
  immediate = NULL,
  bigint = NULL
)

## S4 method for signature 'AdbiConnection,character'
dbSendQuery(
  conn,
  statement,
  ...,
  params = NULL,
  immediate = NULL,
  bigint = NULL
)

## S4 method for signature 'AdbiConnection,character'
dbSendStatement(
  conn,
  statement,
  ...,
  params = NULL,
  immediate = NULL,
  bigint = NULL
)
```

Arguments

conn	A DBI::DBIConnection object, as returned by `DBI::dbConnect()`.
statement	a character string containing SQL.
...	Other parameters passed on to methods.
params	Optional query parameters (forwarded to `DBI::dbBind()`)
immediate	Use `TRUE` for direct execution, `FALSE` to bind parameters, or `NULL` to inspect placeholders when the driver supports it.
bigint	The R type that 64-bit integer types should be mapped to, default is chosen according to the connection setting. See `dbConnect()` for the accepted values.

Details

Multiple open result sets per connection are supported and support can be disabled by setting `options(adbi.allow_multiple_results = FALSE)`. If not enabled, creating a new result will finalize potential other results and throw a warning.

If parameter metadata is unavailable, bind before fetching when parameters are needed. Without metadata, adbi does not check the number or names of the parameters, so a mismatch is reported only if the driver checks for it, possibly not until the statement is executed.

Without metadata, adbi cannot tell whether a statement sent with `immediate = NULL` has placeholders, so such a statement runs only once it is bound or `DBI::dbGetRowsAffected()` is called, as `DBI::dbExecute()` does. Pass `immediate = TRUE` for a statement without placeholders to run when it is sent. For an unbound statement sent with `immediate = FALSE`, `DBI::dbGetRowsAffected()` returns `NA`.

Value

An S4 class `AdbiResult` (inheriting from DBI::DBIResult).

See Also

adbi-driver

Examples

```
if (requireNamespace("adbcsqlite")) {
  library(DBI)
  con <- dbConnect(adbi::adbi(pkg = "adbcsqlite"), uri = ":memory:")
  dbWriteTable(con, "swiss", swiss)
  str(
    dbGetQuery(con, "SELECT Examination from swiss WHERE Agriculture < 30")
  )
  str(
    dbGetQuery(con, "SELECT Examination from swiss WHERE Agriculture < 30",
      bigint = "numeric")
  )
  dbDisconnect(con)
}
```

Index

adbcdrivermanager::adbc_database_init(), *3*
adbcdrivermanager::adbc_driver_monkey(), *4*
adbcsqlite::adbcsqlite(), *4*
adbi, 3

bit64::integer64, *3*

dbConnect(), *6*
dbConnect,AdbiDriver-method *(*adbi*)*, 3
dbConnect_AdbiDriver *(*adbi*)*, 3
dbDisconnect,AdbiConnection-method *(*adbi*)*, 3
dbDisconnect_AdbiConnection *(*adbi*)*, 3
dbFetch,AdbiResult-method *(*dbFetch_AdbiResult*)*, 4
dbFetch_AdbiResult, 4
DBI::dbBind(), *6*
DBI::dbConnect(), *3, 4*, *6*
DBI::dbDisconnect(), *4*
DBI::dbExecute(), *7*
DBI::dbFetch(), *4, 5*
DBI::dbGetQuery(), *5*
DBI::dbGetRowsAffected(), *7*
DBI::DBIConnection, *3, 4*, *6*
DBI::DBIDriver, *3*
DBI::DBIResult, *5*, *7*
DBI::dbSendQuery(), *5*
dbSendQuery,AdbiConnection,character-method *(*dbSendQueryArrow_AdbiConnection*)*, *5*
dbSendQuery_AdbiConnection_character *(*dbSendQueryArrow_AdbiConnection*)*, *5*
dbSendQueryArrow,AdbiConnection-method *(*dbSendQueryArrow_AdbiConnection*)*, *5*
dbSendQueryArrow_AdbiConnection, 5
dbSendStatement,AdbiConnection,character-method *(*dbSendQueryArrow_AdbiConnection*)*, *5*
dbSendStatement_AdbiConnection_character *(*dbSendQueryArrow_AdbiConnection*)*, *5*