

# Package ‘adbi’

January 25, 2024

**Title** 'DBI' Compliant Database Access Using 'ADBC'

**Version** 0.1.1

**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 'adbc-drivermanager' 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 (>= 3.6.0)

**Imports** DBI (>= 1.2.0), methods, adbcdrivermanager (>= 0.8.0),  
nanoarrow (>= 0.3.0)

**Suggests** testthat, covr, DBITest (>= 1.8.0), adbcsqlite (>= 0.8.0),  
withr, bit64, utils, arrow

**Config/Needs/website** r-dbi/dbitemplate

**Encoding** UTF-8

**RoxygenNote** 7.3.1

**Config/autostyle/scope** line\_breaks

**Config/autostyle/strict** false

**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'  
 'nanoarrow.R' 'show\_AdbiConnection.R' 'show\_AdbiDriver.R'  
 'show\_AdbiResult.R' 'show\_AdbiResultArrow.R' 'utils.R'

**NeedsCompilation** no

**Author** Nicolas Bennett [aut, cre],  
 Voltron Data [fnd]

**Maintainer** Nicolas Bennett <nicolas@cynkra.com>

**Repository** CRAN

**Date/Publication** 2024-01-25 17:10:05 UTC

## R topics documented:

adbi . . . . .	3
dbFetch_AdbiResult . . . . .	4
dbSendQueryArrow_AdbiConnection . . . . .	5

*adbi* 3

**Index** 7

---

*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_)

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

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

## Arguments

<code>driver</code>	A driver specification that can be evaluated (with no arguments) to give an <code>adbcdrivermanager::adbc_driver()</code> . See Details for more information.
<code>drv</code>	an object that inherits from <code>DBIDriver</code> , or an existing <code>DBIConnection</code> object (in order to clone an existing connection).
<code>...</code>	Extra arguments passed to <code>dbConnect()</code> are forwarded to <code>adbcdrivermanager::adbc_database_init()</code> .
<code>bigint</code>	The R type that 64-bit integer types should be mapped to, default is <code>bit64::integer64</code> , if <code>bit64</code> is installed and character otherwise
<code>conn</code>	A <code>DBIConnection</code> object, as returned by <code>dbConnect()</code> .
<code>force</code>	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`,
- a string of the form `pkg::fun` (where `pkg::` is optional and defaults to `fun`), which can be used to look up such a function.

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

## Value

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

**Examples**

```
adbi()
if (requireNamespace("adbcsqlite")) {
  adbi("adbcsqlite")
}
library(DBI)
con <- dbConnect(adbi())
dbIsValid(con)
dbDisconnect(con)
dbIsValid(con)
```

---

dbFetch\_AdbiResult      *Fetch result sets*

---

**Description**

When fetching results using `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**

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

**Value**

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

**Examples**

```
if (requireNamespace("adbcsqlite")) {
  library(DBI)
  con <- dbConnect(adbi::adbi("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 `dbSendQuery()` (and by extension using `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 <a href="#">DBIConnection</a> object, as returned by <a href="#">dbConnect()</a> .
statement	a character string containing SQL.
...	Other parameters passed on to methods.
params	Optional query parameters (forwarded to <a href="#">dbBind()</a> )
immediate	Passing a value TRUE is intended for statements containing no placeholders and FALSE otherwise. The default value NULL will inspect the statement for presence of placeholders (will PREPARE the statement)
bigint	The R type that 64-bit integer types should be mapped to, default is chosen according to the connection setting

**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.

**Value**

An S4 class `AdbiResult` (inheriting from [DBIResult](#)).

**See Also**

`adbi-driver`

**Examples**

```
if (requireNamespace("adbcsqlite")) {
  library(DBI)
  con <- dbConnect(adbi::adbi("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 = "integer")
  )
  dbDisconnect(con)
}
```

# Index

adbcdrivermanager::adbc\_database\_init(),  
3  
adbcdrivermanager::adbc\_driver(), 3  
adbcdrivermanager::adbc\_driver\_monkey(),  
3  
adbi, 3  
  
bit64::integer64, 3  
  
dbBind(), 6  
dbConnect(), 3, 6  
dbConnect, AdbiDriver-method (adbi), 3  
dbConnect\_AdbiDriver (adbi), 3  
dbDisconnect(), 3  
dbDisconnect, AdbiConnection-method  
(adbi), 3  
dbDisconnect\_AdbiConnection (adbi), 3  
dbFetch(), 4  
dbFetch, AdbiResult-method  
(dbFetch\_AdbiResult), 4  
dbFetch\_AdbiResult, 4  
dbGetQuery(), 5  
DBI::dbConnect(), 3  
DBIConnection, 3, 6  
DBIDriver, 3  
DBIResult, 4, 6  
dbSendQuery(), 4, 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