

# Package ‘bcputility’

May 9, 2024

**Type** Package

**Title** Wrapper for SQL Server bcp Utility

**Version** 0.4.3

**Description** Provides functions to utilize a command line utility that does bulk inserts and exports from SQL Server databases.

**License** MIT + file LICENSE

**Encoding** UTF-8

**RoxygenNote** 7.2.3

**SystemRequirements** bcp Utility

**Depends** R (>= 3.5.0)

**Imports** data.table, sf, methods

**Suggests** blob

**URL** <https://bcputility.delveds.com>,  
<https://github.com/tomroh/bcputility>

**BugReports** <https://github.com/tomroh/bcputility/issues>

**NeedsCompilation** no

**Author** Thomas Roh [aut, cre]

**Maintainer** Thomas Roh <thomas@roh.engineering>

**Repository** CRAN

**Date/Publication** 2024-05-09 03:20:02 UTC

## R topics documented:

bcpExport . . . . .	2
bcpImport . . . . .	3
createTable . . . . .	4
makeConnectArgs . . . . .	5
mapDataTypes . . . . .	6
SQLServerCLIVersions . . . . .	7

<b>Index</b>	<b>8</b>
--------------	----------

bcpExport

*Export data from SQL Server***Description**

A wrapper for a system call to the bcp utility which writes a SQL Server table or query (T-SQL) to a file.

**Usage**

```
bcpExport(
  file,
  connectargs,
  table,
  query,
  fieldterminator = "\t",
  rowterminator = ifelse(.Platform$OS.type == "windows", "\r\n", "\n"),
  bcpOptions = list("-c", "-b", 1000, "-a", 4096, "-m", 10),
  ...
)
```

**Arguments**

file	output file name
connectargs	named list of connection arguments. See <a href="#">makeConnectArgs</a> .
table	name of the source table when exporting from SQL Server
query	Transact-SQL query that returns a result set. Ignored if table is specified.
fieldterminator	character separator for columns
rowterminator	character separator for rows—new lines
bcpOptions	list of additional options to pass to the bcp utility. See details.
...	arguments to pass <a href="#">system2</a>

**Details**

The bcpOptions allows the user to include additional arguments for the call to system2. Please refer to <https://learn.microsoft.com/en-us/sql/tools/bcp-utility>. The default options are set to the defaults for bcp CLI. -b refers to number of rows to write at a time; 10,000 to 50,000 is a starting recommendation. -a refers to size of packets to be sent in bytes. -e refers to the maximum number of errors before failure.

**Value**

No return value. Operations from bcp are printed to console; see ... to redirect output

bcpImport

*Import data to SQL Server***Description**

A wrapper for a system call to the bcp utility which bulk inserts to SQL Server.

**Usage**

```
bcpImport(
  x,
  connectargs,
  table,
  fieldterminator = "\t",
  rowterminator = ifelse(.Platform$OS.type == "windows", "\r\n", "\n"),
  overwrite = FALSE,
  spatialtype = c("geometry", "geography"),
  bcpOptions = list("-b", 1000, "-a", 4096, "-m", 10),
  ...
)
```

**Arguments**

x	dataframe object or path to file
connectargs	named list of connection arguments. See <a href="#">makeConnectArgs</a> .
table	Name of the source table when importing from SQL Server. For specifying the schema in the table name see <schema>.<table> and if not specified the default is "dbo".
fieldterminator	character separator for columns
rowterminator	character separator for rows—new lines
overwrite	Whether to overwrite the table if it exists
spatialtype	spatial data type for schema <a href="https://docs.microsoft.com/en-us/sql/relational-databases/spatial/spatial-data-types-overview">https://docs.microsoft.com/en-us/sql/relational-databases/spatial/spatial-data-types-overview</a> , ignored if x is not an 'sf' object
bcpOptions	list of additional options to pass to the 'bcp' utility. See details.
...	arguments to pass to <a href="#">system2</a>

**Details**

If x is a dataframe object, `data.table::fwrite` is used to write the in memory object to disk in a temporary file that is deleted when the function exits. The `fieldterminator` and `rowterminator` are ignored in this case.

If `overwrite` is TRUE, any existing table of the same name will be deleted and the schema is inferred from `DBI::dbCreateTable`. To use a customized schema, create the schema before calling the function and use `overwrite=FALSE`.

If `x` is a `sf` object, the geometry column is converted to binary and written to the database before conversion to `geometry/geometry` data type. The EPSG code is automatically read from the `sf` object and used as the SRID.

To override the default path to the `bcp` command line utility, set the `bcputility.bcp.path` option. To override the default path to the `sqlcmd` command line utility, set the `bcputility.sqlcmd.path` option.

The `bcpOptions` allows the user to include additional arguments for the call to `system2`. Please refer to <https://learn.microsoft.com/en-us/sql/tools/bcp-utility>. The default options are set to the defaults for `bcp` CLI. `-b` refers to number of rows to write at a time; 10,000 to 50,000 is a starting recommendation. `-a` refers to size of packets to be sent in bytes. `-e` refers to the maximum number of errors before failure.

### Value

Output from `system2`. See ... to redirect output.

---

createTable	<i>Create or drop table</i>
-------------	-----------------------------

---

### Description

Create or drop table

### Usage

```
createTable(connectargs, table, coltypes, ...)
```

```
dropTable(connectargs, table, ...)
```

```
checkTableExists(connectargs, table)
```

### Arguments

connectargs	named list of connection arguments. See <a href="#">makeConnectArgs</a> .
table	Name of the source table when importing from SQL Server. For specifying the schema in the table name see <code>&lt;schema&gt;.&lt;table&gt;</code> and if not specified the default is "dbo".
coltypes	character vector of data types with the column names as list/vector names. Use <a href="#">mapDataTypes</a> or refer to for proper format.
...	arguments to pass to <a href="#">system2</a>

### Value

No return value. Operations from `bcp` are printed to console; see ... to redirect output

---

makeConnectArgs	<i>Create a named list of connection arguments to translate to bcp and sqlcmd options</i>
-----------------	---

---

### Description

Create a named list of connection arguments to translate to bcp and sqlcmd options

### Usage

```
makeConnectArgs(  
  server,  
  database,  
  username,  
  password,  
  trustedconnection = TRUE,  
  trustservercert = FALSE,  
  azure = FALSE,  
  quotedidentifiers = FALSE  
)
```

### Arguments

server	the instance of SQL Server to which to connect
database	specifies the database to connect to
username	login ID
password	password for login ID
trustedconnection	use integrated security, username and password are not required
trustservercert	trust the server certificate
azure	use Azure Active Directory authentication, does not work with integrated authentication.
quotedidentifiers	set QUOTED_IDENTIFIER option to 'ON' for the connection between bcp/sqlcmd and SQL Server.

### Value

a list with connection arguments

---

mapDataTypes	<i>Determine SQL Server data types from data frame. Follows SQL Server data type size constraints and chooses the smallest data type size.</i>
--------------	--

---

### Description

Determine SQL Server data types from data frame. Follows SQL Server data type size constraints and chooses the smallest data type size.

### Usage

```
mapDataTypes(x, coltypes)
```

```
varChar(x)
```

```
varBinary(x)
```

```
int(x)
```

### Arguments

x                    data.frame object

coltypes            vector with names of columns to override the default data type mapping

### Value

character vector with names of columns

### Examples

```
mapDataTypes(data.frame(  
  int = 1:5L,  
  numeric = seq(0, 1, length.out = 5),  
  character = LETTERS[1:5],  
  factor = paste(LETTERS[1:5], LETTERS[1:5], sep = ''),  
  logical = c(TRUE, FALSE, TRUE, FALSE, TRUE),  
  date = seq(Sys.Date() - 4, Sys.Date(), 1L),  
  datetime = seq(Sys.time() - 5, Sys.time(), length.out = 5)  
)  
)
```

---

SQLServerCLIVersions *Check bcp and sqlcmd versions*

---

**Description**

Check bcp and sqlcmd versions

**Usage**

bcpVersion(...)

sqlcmdVersion(...)

**Arguments**

... arguments to pass [system2](#)

# Index

bcpExport, 2  
bcpImport, 3  
bcpVersion (SQLServerCLIVersions), 7  
  
checkTableExists (createTable), 4  
createTable, 4  
  
dropTable (createTable), 4  
  
int (mapDataTypes), 6  
  
makeConnectArgs, 2–4, 5  
mapDataTypes, 4, 6  
  
sqlcmdVersion (SQLServerCLIVersions), 7  
SQLServerCLIVersions, 7  
system2, 2–4, 7  
  
varBinary (mapDataTypes), 6  
varChar (mapDataTypes), 6