We ran into an issue the other day where I was tring to write a new table to a  
SQL Server Database with a non-default schema. We did end up spending a bit of time debugging and researching so We wanted to share for anyone else that runs into the issue. Using the DBI::Id function,  
allows you to specify the schema when you are trying to write a table to a SQL Server database.

DBI::dbWriteTable(con,

DBI::Id(schema = "schema", table = "tablename"),

df)

But the code above will return a strange error:



After some investigation I found a workaround to be able to write the table. For non-default schemas, a “\_” needs to in the table name for it to work.

DBI::dbWriteTable(con,

DBI::Id(schema = "schema", table = "tablename\_"),

df)

This really isn’t ideal for naming conventions so using the t-sql command sp\_rename will rename the table to what I originally wanted.

DBI::dbWriteTable(con,

DBI::Id(schema = "schema", table = "tablename"),

df)

DBI::dbGetQuery(con, "USE database;

EXEC sp\_rename '[schema].[tablename\_]', 'tablename';")

I ran into the same issues for overwriting tables as well but a workflow for doing the same is simply to use sp\_rename a couple of times.

DBI::dbGetQuery(con, "USE database;

EXEC sp\_rename '[schema].[tablename]', 'tablename\_';")

DBI::dbWriteTable(con,

DBI::Id(schema = "schema", table = "tablename\_"),

df,

overwrite = TRUE)

DBI::dbGetQuery(con, "USE database;

EXEC sp\_rename '[schema].[tablename\_]', 'tablename';")