Interfacing with databases from R

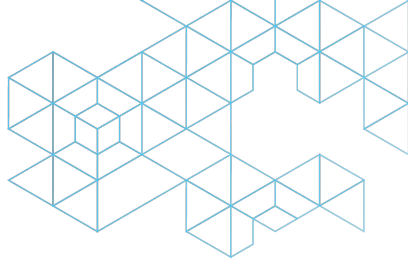
Julia Silge

https://juliasilge.com/

Hello! I’m Julia Silge

Data Scientist, Stack Overflow

@juliasilge



stackoverflow

Where is the data that you need to analyze?

Databases

	Name	dplyr support	Connect via R package	RStudio Pro Driver available
	Amazon Redshift	Yes		Yes
	Apache Hive	Yes		Yes
	Apache Impala	Yes		Yes
	Google BigQuery	Yes	bigrquery	No
	Microsoft SQL Server	Yes		Yes
	MonetDB	Yes	MonetDBLite	No
	MySQL	Yes	RMariaDB	No
	Oracle	Yes		Yes
	Other Databases			
	PostgreSQL	Yes	RPostgreSQL	Yes
	SQLite	Yes	RSQLite	No
	Salesforce	None		Yes
	Teradata	Yes (dbplyr dev)		Yes

https://db.rstudio.com/databases

Demo time!

Databases

	Name	dplyr support	Connect via R package	RStudio Pro Driver available
	Amazon Redshift	Yes		Yes
	Apache Hive	Yes		Yes
	Apache Impala	Yes		Yes
	Google BigQuery	Yes	bigrquery	No
	Microsoft SQL Server	Yes		Yes
	MonetDB	Yes	MonetDBLite	No
	MySQL	Yes	RMariaDB	No
	Oracle	Yes		Yes
	Other Databases			
	PostgreSQL	Yes	RPostgreSQL	Yes
	SQLite	Yes	RSQLite	No
	Salesforce	None		Yes
	Teradata	Yes (dbplyr dev)		Yes

https://db.rstudio.com/databases

What made that possible?

DBI

The DBI package defines a common interface between the R and database management systems (DBMS). The interface defines a small set of classes and methods similar in spirit to Perl's DBI, Java's JDBC, Python's DB-API, and Microsoft's ODBC. It defines a set of classes and methods defines what operations are possible and how they are performed:

- connect/disconnect to the DBMS
- create and execute statements in the DBMS
- extract results/output from statements
- error/exception handling
- information (meta-data) from database objects
- transaction management (optional)

DBI separates the connectivity to the DBMS into a "front-end" and a "back-end". Applications use only the exposed "front-end" API. The facilities that communicate with specific DBMSs (SQLite, MySQL, PostgreSQL, MonetDB, etc.) are provided by "drivers" (other packages) that get invoked automatically through S4 methods.

https://github.com/r-dbi/DBI

odbc

The goal of the odbc package is to provide a DBI-compliant interface to Open Database Connectivity (ODBC) drivers. This allows for an efficient, easy to setup connection to any database with ODBC drivers available, including SQL Server, Oracle, MySQL, PostgreSQL, SQLite and others. The implementation builds on the nanodbc C++ library.

- Installation
  - Windows
  - MacOS
  - Linux - Debian / Ubuntu
  - R
- Connecting to a Database
  - Connection Strings
  - DSN Configuration files
- Usage
  - Table and Field information
  - Reading
  - Writing

https://github.com/r-dbi/odbc

STEP 1

Installing drivers

- Windows is usually bundled with ODBC drivers.
- For MacOS, use homebrew, i.e.
  - `brew install unixodbc`
  - `brew install freetds --with-unixodbc`
- RStudio offers professional drivers for enterprise customers

STEP 2

Creating a connection

- Using the DBI and odbc packages, we can create a connection.

```
con <- DBI::dbConnect(odbc::odbc(),
                      driver = "libtdsodbc.so",
                      database = database,
                      uid = paste0("STACKEXCHANGE\\", username),
                      pwd = password,
                      Server = name,
                      port = 1433)
```

STEP 3

Creating a source

- Using the dbplyr package, we can create a data source.

```
library(dplyr)
library(dbplyr)

src <- src_dbi(con)
```

- The source object will tell you what tables are available in the database.

Managing authentication

keyring

> Access the System Credential Store from R

Platform independent API to access the operating systems credential store. Currently supports:

- Keychain on macOS,
- Credential Store on Windows,
- the Secret Service API on Linux, and
- environment variables on all platforms. Additional storage backends can be added easily.

https://github.com/r-lib/keyring

Let’s build a package!

Packages

dplyr

DBI

odbc

pool

dbplot

tidypredict

RStudio

Connections Pane

Professional Drivers

Best Practices

Setting up ODBC Drivers

Run Queries Safely

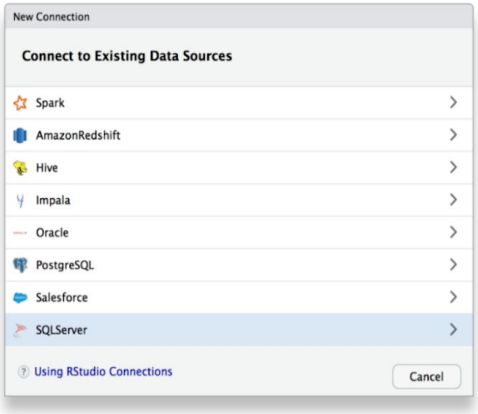
Securing Deployed Content

Databases using R

At **RStudio**, we are working to make it as easy as possible to work with databases in R. This work focuses on **three key areas**:

1. RSTUDIO PRODUCTS

- The new RStudio Connections Pane makes it possible to easily connect to a variety of data sources, and **explore the objects and data** inside the connection
- To RStudio commercial customers, we offer RStudio Professional ODBC Drivers, these are data connectors that help you connect to some of the most popular databases.



2. USE BEST-IN-CLASS PACKAGES

Build and/or document how to use packages such as: dplyr, DBI, odbc, keyring and pool

3. PROMOTE BEST PRACTICES

This website is the main channel to provide support in this area. RStudio is also working through other delivery channels, such as upcoming webinars and in-person training during our RStudio conferences.

https://db.rstudio.com/

implyr

implyr is a SQL backend to dplyr for Apache Impala, the massively parallel processing query engine for Apache Hadoop. Impala enables low-latency SQL queries on large datasets stored in HDFS, Apache HBase, Apache Kudu, Amazon S3, Microsoft ADLS, and Dell EMC Isilon.

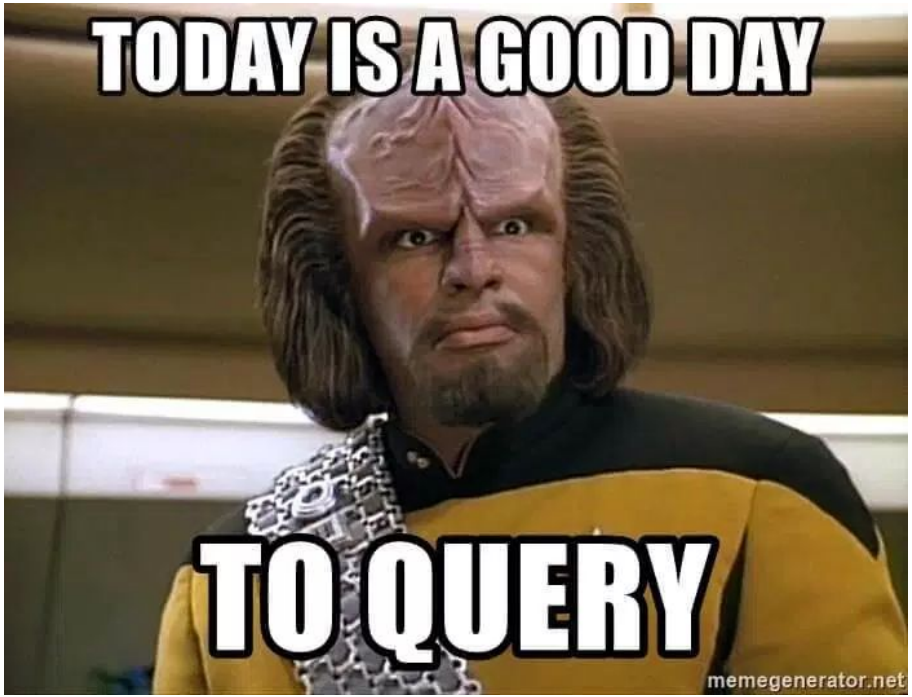
implyr is designed to work with any DBI-compatible interface to Impala. implyr does not provide the underlying connectivity to Impala, nor does it require that you use one particular R package for connectivity to Impala. Currently, two packages that can provide this connectivity are odbc and RJDBC. Future packages may provide other options for connectivity.

https://github.com/ianmcook/implyr

Ian's talk from useR! 2017

Painless ODBC + dplyr Connections to Amazon Athena and Apache Drill

https://rud.is/b/2018/04/20/painless-odbc-dplyr-connections-to-amazon-athena-and-apache-drill-with-r-odbc/



stackoverflow

Thanks!

to
Nick Larsen
Kevin Montrose
Jason Punyon

Find me at @juliasilge and https://juliasilge.com/