While at RStudio are pleased to see R climbing the TIOBE charts, what we’re going to focus on this month is all the other languages, both on this list and not, that data science teams also use to do their jobs. We’re going to focus on **interoperability** with R, and how it helps data science teams get more value of all their organization’s analytic investments.

What’s less well-known, however, is that when you write code in R Markdown within the IDE, you may also embed:

* **SQL code** for accessing databases,
* **BASH code** for shell scripts,
* **C and C++ code** using the Rcpp package,
* **STAN code** for doing statistical modeling,
* **Javascript** for doing web programming,
* **and many more languages**. You can find a complete list of the many platforms supported in the language engines chapter of the book.

If you’re wondering how this could work, I’ve created a very simple example R Markdown document that demonstrates how languages can work together. It creates an in-memory database of gapminder data, queries it using SQL, prints the result of the query in R, plots the result using matplotlib in Python and saves the result as an image, and then prints the size of the image in BASH.

```{r setup, include=FALSE, echo = TRUE}

knitr::opts\_chunk$set(echo = TRUE, collapse = TRUE)

library(tidyverse)

library(rlang)

library(reticulate)

library(RSQLite)

library(DBI)

library(gapminder)

reticulate::use\_python("/usr/local/bin/python3", required = TRUE)

```

```{r gm\_db\_setup}

gapminder\_sqllite\_db <- dbConnect(RSQLite::SQLite(), ":memory:")

dbWriteTable(conn = gapminder\_sqllite\_db,"gapminder", gapminder)

country <- "Switzerland"

```

## use R variable `country` in SQL query

```{sql connection = gapminder\_sqllite\_db, output.var="gmdata"}

SELECT \* FROM gapminder WHERE country = ?country

```

## Access results of SQL query in R

```{r}

head(gmdata, 5)

## country continent year lifeExp pop gdpPercap

## 1 Switzerland Europe 1952 69.62 4815000 14734.23

## 2 Switzerland Europe 1957 70.56 5126000 17909.49

## 3 Switzerland Europe 1962 71.32 5666000 20431.09

## 4 Switzerland Europe 1967 72.77 6063000 22966.14

## 5 Switzerland Europe 1972 73.78 6401400 27195.11

```

## Plot in Python and save result as .png

```{python}

import matplotlib.pyplot as plt

plt.plot(r.gmdata.year, r.gmdata.lifeExp)

plt.grid(True)

plt.title("Switzerland Life Expectancy (years)")

plt.savefig("./SwitzerlandLifeExp.png")

```

## Show size of Python plot using BASH

```{bash}

ls -l SwitzerlandLifeExp.png

## -rw-r--r-- 1 chowe staff 26185 Jul 7 17:26 SwitzerlandLifeExp.png

```