

10.1: R and databases

There are many interfaces which connect R with different database management software and there is even package [sqldf](#) which allows to work with R data frames through commands from SQL language. However, the R core also can work in the database-like way, even without serious extension. The Table 10.1.1 shows the correspondence between SQL operators and commands of R.

SELECT	[, subset()
JOIN	merge()
GROUP BY	aggregate() , tapply()
DISTINCT	unique() , duplicated()
ORDER BY	order() , sort() , rev()
WHERE	which() , %in% , ==
LIKE	grep()
INSERT	rbind()
EXCEPT	! and -

Table 10.1.1 Approximate correspondence between SQL operators and R functions.

One of the most significant disadvantages there is that many of these R commands (like [merge\(\)](#)) are slow. Below are examples of the user functions which work much faster:

Now we can operate with multiple data frames as with one. This is important if data is organized hierarchically. For example, if we are measuring plants in different regions, we might want to have two tables: the first with regional data, and the second with results of measurements. To connect these two tables, we need a *key*, the same column which presents in both tables:

Here was shown how to work with two related tables and [Recode\(\)](#) command. First table contains locations, the second—measurements. Species names are only in the first table. If we want to know the correspondence between species and characters, we might want to merge these tables. The key is [N.POP](#) column (location ID).

The [recode.r](#) collection of R functions distributed with this book contains ready-to-use [Recode\(\)](#) function and related useful tools.

There is another feature related with databasing: quite frequently, there is a need to convert “text to columns”. This is especially important when data contains pieces of text instead of single words:

(Vectorized function call [do.call\(\)](#) constructs a function call its arguments.)

There is also the *data encoding* operation which converts categorical data into binary (0/1) form. Several ways are possible:

R and TeX are friendly software so it is possible to make them work together in order to automate book processing. Such books will be “semi-static” where starting data comes from the regularly updated database, and then R scripts and TeX work to create typographically complicated documents.

Flora and fauna manuals and checklists are perfect candidates for these semi-static manuals. This book supplements contain the archived folder [manual.zip](#) which illustrates how this approach works on example of imaginary “kubricks” (see above).

This page titled 10.1: R and databases is shared under a [Public Domain](#) license and was authored, remixed, and/or curated by Alexey Shipunov via source content that was edited to the style and standards of the LibreTexts platform.