The extrelation package

Fabian Pijcke

2017/03/01 — v0.1

Abstract

This package supports research in the field of relational databases by providing simple commands and environments to produce relations given by extension. It supports both vertical partitioning (data-wise) and horizontal partitioning of the attributes and is easy to use if only basic relations are needed.

1 Introduction

Research papers in the field of relational databases often make use of relations given by their extension. Especially within examples or when illustrating proofs working on database construction. Usually producing such relations is a hassle and inconsistencies in style come easily. This package addresses both issues.

2 Simple Relations

Typesetting a relation composed of N attributes and some N-tuples is as easy as what follows.:

Code : `\extrelation {}{A, B, C}{1, 2, 3 ; 4, 5, 6}`

Result :

A	B	C
1	2	3
4	5	6

The attributes are to be placed as the second mandatory argument and must be comma-separated. The tuples form the third and last attribute and are a semicolon-separated list of tuples, which themselves are colon-separated lists of values.

2.1 Relation Name

The first mandatory attribute allows one to set a relation name. You should not put any LaTeXcode here, for stylizing is done through the `relfun` option (see Section 5.4

Code : `\extrelation {Emp}{Name, Salary}{A, 42; B, 21}`
Result :

Emp	**Name**	**Salary**
	A	42
	B	21

If the first attribute is empty, then no column is generated for the relation name. This may be important to know if you use the `extrelationenv` environment without the `extrelationrow` utility command.

2.2 Attribute Names

The second mandatory argument is a list of comma-separated attribute names. Again no LaTeXcode should be put here.

2.3 Tuples

The last mandatory argument contains the tuples to be put in the relation. You should also avoid to put LaTeXcode here, but this can be needed if you want subscripts for example. Be sure that the style (see Section 5.4) do not conflicts with your values in such case.

3 Horizontal Partitioning of the Attributes

It is not unusual to deal with relations that are horizontally split in two or more parts. For example, some papers need to distinct the key attributes from the other attributes. They usually do so by underlining the attributes that are part of the key. One can typeset a relation in which the attributes A and B are in the key, while attribute C is the only attribute outside the key, as follows:

Code : `\extrelation [attrfuns={\underline {\v }, \v }]{}{A, B; C}{a, b, c}`
Result :

$\underline{A}$	$\underline{B}$	C
a	b	c

Usually, one would then create a macro that supports this type of relation:
`\newcommand{\keyrel}{\extrelation[attrfuns={\underline{\v}, \v}]}`
More informations on the `attrfuns` option can be found in Section 5.4.

4 Vertical Partitioning

Some authors find useful to vertically partition the relation. This is done using the environment instead of the command. The following example uses a dashed line to split up the tuples of the relation in two parts.

Code:
`\[\begin {extrelationenv}{Relation}{A, B, C}`

```
\extrelationrows {a, b, c; a, d, e} \\
\cmidrule {2-4}
\extrelationrows {f, g, h}
\end {extrelationenv}\]
```

Result:

Relation	**A**	**B**	**C**
	a	b	c
	a	d	e
	f	g	h

Inside the `extrelationenv` environment, you are in an `array` environment. However we do not recommend using this fact to construct rows by yourself, as it would potentially break the visual coherence of your relations.

5 Options Reference

The following options may be set globally using the `exrsetdefault` command, or locally within the `extrelation` and `extrelationrows` command or the `extrelationenv` environment.

5.1 toprule, bottomrule

These options allow to put toprule and bottomrule.

Code : `\extrelation [toprule,bottomrule]{}{A, B}{a, b; c, d}`

Result :

A	**B**
a	b
c	d

5.2 leftrule

This option allows to put a vertical rule between the relation name and the relation content.

Code : `\extrelation [leftrule]{Rel}{A, B}{a, b; c, d}`

Result :

Rel	**A**	**B**
	a	b
	c	d

5.3 pars

This option allows to parenthesize the relation attributes.

Code : `\extrelation [pars]{r = }{A, B}{a, b; c, d}`
Result :

R = (**A** **B**)

a	b
c	d

5.4 relfun, attrfuns, confun

These options allow to change the style of the elements composing the relation. Each function can make use of the command `v`. The `attrfuns` option is particular in the sense it can be made of several functions, each of which will be used by one part of the attributes (parts are separated by a semicolon, see Section 3).

The `relfun` applies to the relation name, defaulting to `\textsc {\v }`.
Code : `\extrelation [relfun={\v }]{Rel}{A, B}{a, b; c, d}`
Result :

Rel

A	**B**
a	b
c	d

The `attrfuns` applies to the attribute names, defaulting to `\mathbf {\v }`. Beware that any attribute in a part after the last function defined in `attrfuns` will not be typeset.
Code : `\extrelation [attrfuns={\underline {\v },\v }]{}{A; B}{}`
Result :

$\underline{A}$	B

The `confun` applies to the values (con stands for constants), defaulting to `\mathsf {\v }`.
Code : `\extrelation [confun=\v ]{Rel}{A, B}{a, b; c, d}`
Result :

Rel

A	**B**
a	*b*
c	*d*