# Data publishing

**Siep Kroonenberg**

Economics Department, Groningen University

**Abstract**

This paper demonstrates how commercial software and LaTeX can work together in data publishing.

**Keywords:** database, spreadsheet, table

## Introduction

One of (LA)TeX's strong suits is automation. Every LaTeX-user knows about LaTeX's features for automatic cross-references and automatic tables of contents. But some people may not realize how convenient LaTeX can be for publishing data from *commercial* programs. Therefore I present here a couple of real-life examples.

CCSO, the economic research group with which I am associated specializes in the building of econometric models, for forecasting, policy analysis and theoretical exercises. The data for a model typically consist of several hundreds of time series. Now and then we publish these databases, both the data themselves and associated descriptive information.

## 1 The descriptive database

The descriptive database contains for each time series a description in economic terms and a definition which contains either its source or a definition in terms of other series. A series taken directly from a published source (in this case, the 'CBS Kwartaalrekeningen') might have the following database entry:

```
CNAME       COGRABS
TEXNAME     C_o
DESCRP      Consumption of general
            government at current prices
DEFINITION  1000 * CBS KR 1.3.1 row 5
```

For a constructed series, we might have

```
CNAME       CP
TEXNAME     \dot{C}_p
DESCRP      Growth rate of consumption of
            households at constant prices
DEFINITION  100*(CPKLABS / CPKLABS(-1) - 1)
```

`CNAME` indicates the name used in the computermodels and the spreadsheets; `TEXNAME` the TeX code used in papers; `DESCRP` is the definition in economic terms, and `DEFINITION` specifies either the source or how it is computed from other series.

We wish to emulate and improve upon a 'native' dBase report, in which each field has its own column, and long fields are wrapped within their column.

There are two problems associated with using the `tabular` environment for this purpose: first, a tabular environment can't span more than one page, which is not nearly enough, and second, a tabular environment justifies text even within a narrow column, which can be very ugly.

The first problem is solved by using Theo Jurriëns' `\supertabular` environment, which is defined in the `supertab` style file or package. This environment also allows the specification of a table header which is repeated on every page.

We solve the second problem in a less elegant way: we wrap each long field in a parbox of appropriate width:

```
\newcommand{\px}[1]{{\parbox[t]{1in}%
  {\raggedright #1}}}
\newcommand{\py}[1]{{\parbox[t]{1.3in}%
  {\raggedright #1}}}
```

The following dBase commands create a file `db94tex.txt` containing a listing in LaTeX format:

```
set alternate to db94tex
set talk off
set console off
set alternate on
go top
do while .not. eof()
do dbrec
skip
enddo
set alternate off
close alternate
set console on
set talk on
```

This is the procedure `dbrec` that was called above:

```
procedure dbrec
?'$'+trim(texname)+'$ &'
?'\tt '+trim(cname)+' &'
?'\px{'+trim(descrp)+'} &'
?'\py{'+trim(definition)+'} \\'
?
return
```

A fragment of output from these commands:

```
$C_o$ &
\tt COGRABS &
\px{Consumption of general government at
  current prices} &
\py{1000 * CBS KR 1.3.1 row 5} \\

$\dot{C}_p$ &
\tt CP &
\px{Growth rate of consumption of
  households at constant prices} &
\py{100 * (CPKLABS / CPKLABS(-1) - 1)} \\
```

The following LaTeX code will process the file `db94tex.txt` (remember to include `supertab` as a documentstyle option):

```
{
\renewcommand{\arraystretch}{2}
% for extra vertical space between records;
% a pair of braces limits this redefinition
% to the table

\tablehead{& \bf Series & \bf Description & %
  \bf Source/definition\\}
\begin{supertabular}%
  {@{}p{0.3in}p{0.5in}p{1in}p{1.3in}@{}}
\input db94tex.txt
\end{supertabular}
}
```

And we get:

| | **Series** | **Description** | **Source/definition** |
|---|---|---|---|
| $C_o$ | COGRABS | Consumption of general government at current prices | 1000 * CBS KR 1.3.1 row 5 |
| $\dot{C}_p$ | CP | Growth rate of consumption of households at constant prices | 100 * (CPKLABS / CPKLABS(-1) - 1) |

### The alternative: tabbing

The example can be easily adapted for a tabbing environment; in fact, that is what we actually did, ignorant of the blessings of `supertabular`. We even managed to press the page header into service as a repeating table header. But now we know better.

## 2 The timeseries spreadsheet

Our timeseries are stored in spreadsheets, such as

| | A | B | C | D | E |
|---|---|---|---|---|---|
| 1 | | BPRGRABS | BPRKLABS | COGRABS | COKLABS |
| 2 | | $B\!P\!R$ | $bpr$ | $C_o$ | $c_o$ |
| 3 | 1980Q1 | 80500 | 83831 | 14100 | 15041 |
| 4 | 1980Q2 | 86400 | 87945 | 17700 | 18233 |
| 5 | 1980Q3 | 82000 | 82501 | 13700 | 13842 |
| 6 | 1980Q4 | 87800 | 87800 | 14700 | 14700 |
| 7 | 1981Q1 | 84800 | 84250 | 15000 | 15643 |
| 8 | 1981Q2 | 89700 | 87945 | 18500 | 18780 |

The first column contains dates; `1981Q1` indicates the first quarter of 1981. The first row contains the names used in the computer model; the second their TeX equivalents (in practice, we add the TeX names only when we are going to print the database; for this, we use the descriptive database discussed above).

Generating TeX code from such a spreadsheet requires fewer tricks than in the dBase case since we don't have to wrap lines within cells. However, it can't be automated as effectively because, in spite of their macro capabilities, spreadsheets are not really geared towards batch processing. Maybe some day I'll write a Pascal program that does it right.

Meanwhile, we'll let Lotus' print facility generate the output. With some suitable macros (not detailed here), it isn't too much of a job to massage the spreadsheet into shape: we add '`&`' alignment symbols with a macro, and a column of '`\\`' new-line symbols by hand. Printing a block to a file `timesers.prn` we get something like

```
       &  $B\!P\!R$ & ... & $\dot{C}_p$\\[1pc]
1980Q1 &      80500 & ... &    -4.4301 \\
1980Q2 &      86400 & ... &     0.7075 \\
1980Q3 &      82000 & ... &    -0.5809 \\
1980Q4 &      87800 & ... &     3.4624 \\[6pt]
1981Q1 &      84800 & ... &    -5.3955 \\
1981Q2 &      89700 & ... &     0.1937 \\
```

The lines

```
\begin{tabular}{@{}lrrrrr@{}}
\input timesers.prn
\end{tabular}
```

produce the table below:

| | $BPR$ | $bpr$ | $C_o$ | $c_o$ | $\dot{C}_p$ |
|---|---|---|---|---|---|
| 1980Q1 | 80500 | 83831 | 14100 | 15041 | -4.4301 |
| 1980Q2 | 86400 | 87945 | 17700 | 18233 | 0.7075 |
| 1980Q3 | 82000 | 82501 | 13700 | 13842 | -0.5809 |
| 1980Q4 | 87800 | 87800 | 14700 | 14700 | 3.4624 |
| 1981Q1 | 84800 | 84250 | 15000 | 15643 | -5.3955 |
| 1981Q2 | 89700 | 87945 | 18500 | 18780 | 0.1937 |

## 3 Caveats

Watch out for characters in the database or spreadsheet which are also markup codes for TeX, such as percent- or ampersand characters. In spreadsheets, texts should fit within their columns, or they will be truncated without warning. Also, if the block to be printed exceeds the declared linelength of the printfile, the table will be split horizontally.