Calculate Database Size

Jochen Hein

$Id: dbsize.xml,v 1.1 2002/08/22 09:24:55 jhein Exp $

Table of Contents

Calculate the database size 1
The code 1
Example files 3

Calculate the database size

If you are creating a database with a custom load or perform a database migration you will have to create your own `DBSIZE.TPL`. This file will be used to create the tablespaces and describes the distribution over the file systems. The export will create a file for you, but most of the time you have other sizes for the filesystems or plan to move some tablespaces around.

It is not hard to create the file manually, but checking the sizes for several tries is tendious. So this is a perfect case for a script. To run the script, change into the dircetory where your `DBSIZE.TPL` resides and run:

```
$ awk -f dbsize.awk DBSIZE.TPL
...
```

The output is a table with all planned SAPDATAs and the size that is currently needed for them. The syntax is currently not exactly what `DBSIZE.TPL` would need, but it is not neccessary that the values in the file are correct.

The code

The code has been written in AWK, a Unix scripting language, that is well suited for maniputating text files. As you will see, nothing magic hase been done here.

The main code of the program is structured as follows:

Main Listing

```
#!/usr/bin/awk -f
# Usage:
# awk -f dbsize.awk DBSIZE.TPL | sort
#
Set the field seperator
Sum up the file sizes
finally, print the sums
```

Normally AWK uses a blank and a tabulator as a field separator. To make parsing the file `DBSIZE.TPL` easier, we'll use other characters (=, semicolon, and exlamation mark) here. Each line will be split at one of these characters, and we'll find the needed values easily.

```
 =

BEGIN { FS="[=;!]"; }
```

For each line that contains a "=" we'll sum the values up. If the number of fields (NF) exceeds the expected maximum, an error is printed. Otherwise, all sizes of the datafiles are added to the sapdata usage they reside in. The array size is used as an assoziative array, the index is the sapdata name.

It might be possible to switch to a loop and eval the field number. If you do that, send me a patch...

```
 =

/\=/ {
  size[$2] += $3;
  if ( NF > 4) size[$4] += $5;
  if ( NF > 6) size[$6] += $7;
  if ( NF > 8) size[$8] += $9;
  if ( NF > 10) size[$10] += $11;
  if ( NF > 12) size[$12] += $13;
  if ( NF > 14) size[$14] += $15;
  if ( NF > 14) print "ERROR"
}
```

At the end, when all sizes are added up, we'll print them. Since we used an assoziative array, the output is not sorted. To compensate for that, just sort the output. The format is not exactly what is stored in `DBSIZE.TPL`, but in the end, it's not really needed.

```
finally, print the sums =


END {
  for ( data in size ) printf "%s:\t%15d\n", data, size[data];
}
```

Example files

Give some examples hers.