Tsearch2 and Unicode/UTF-8

Here's a quick install guide for tsearch2 and a German Unicode-database. As we're talking UTF-8, language doesn't really matter, so some very similar procedure may apply for the language of your choice – just replace de_DE.UTF-8 and other configuration options with the settings relevant to your needs. I'll assume that the database(s) you want to use tsearch2 with are using the Unicode-encoding. If not, you're reading the wrong instructions :) Furthermore, I'll assume you have installed ispell on your system. If not, refer to your operating system's docs on how to do this. This guide is based on PostgreSQL 7.4.x; there may or may not be adjustments necessary for future versions.

I. Preparations: The correct locale

First, make sure, your database-cluster has got the correct locale settings. Although you might not have experienced any problems if your locale settings actually don't reflect the usage of Unicode in your database, there are lots of things which will not work as expected – and tsearch2 is definitely one of them.

The most important locale-settings of a PostgreSQL-cluster are set at *initdb* and cannot be changed afterwards without going through *dump*, *initdb* and *reload* again. To check your settings, run the following at a command-prompt on your PostgreSQL-machine

```
[path_to_pg_binaries]/pg_controldata [path_to_db_cluster]|grep LC
```

If you get *de_DE.UTF-8* for both settings, your cluster is set up correctly; if you get anything like *de_DE@euro*, *C*, *POSIX* or *de_DE.iso885915@euro* instead, there's no other way but to dump your complete installation, stop the postmaster, move your database-cluster directory (e.g. /var/lib/pgsql/data/) out of the way to some backup location, then set your systems locale correctly:

```
export LC_ALL=de_DE.UTF-8
```

If that fails, you may display the installed locales with *locale -a*. If *de_DE.UTF-8* is missing, check your operating system's docs on how to install it. One small hint: If you're connecting remotely to your system, you may need to adjust the encoding of your client, too; e.g. in PuTTY go to Category, Window, Translation and set “Character set translation on received data” to *UTF-8*, otherwise any special characters with codes >128 won't be displayed correctly.

Once you have installed and set the needed locale, proceed with running *initdb* again:

```
su [postgres_user]
[path_to_pg_binaries]/initdb -D –locale=de_DE.UTF-8 [path_to_db_cluster] -L
[path_to_input_files]
```

Now that you've got a virgin database cluster which is supposedly set up with the correct locale, doublecheck this again with *pg_controldata*. If the output indicates that all has gone well, reload your dump.

II. Installing tsearch2 into your Database

Now you've got a nicely prepared cluster, you'll want to install tsearch2; change to your PostgreSQL-source-directory and move to the contrib/tsearch2-subdirectory. Compile and install tsearch2 thus:

```
make
make install
[path_to_pg_binaries]/pgsql -d [your_db] -U [your_usr] <[path_to]/
contrib/tsearch2.sql
```

II. Preparing the Language Files

Tsearch2 needs some dictionary-files for each language used. You'll need four files: german.aff, german.med, one german.stop and a german.stop.ispell. The first two are ispell-files, the third one is

a simple list containing all the stop-words you wish to exclude from indexing, one word per line, and the last one is derived from the third.

Concerning the stopwords-file *german.stop*, you can compile one yourself from scratch, let Google help you or whatever – it shouldn't be too difficult in any case. If it's UTF-8 encoded, do yourself a favour and omit the Byte Order Mark (BOM) – tsearch2 doesn't like it an would ignore the first line of your stopwords-list if it was there.

As stemmer and ispell-dictionaries use different procedures for sieving out the stopwords, we will in fact need two different stopwords-files, one for each. We'll use a bit of Perl and the commandline in order to generate the ispell-file from the stemmer-file. More on that later on, for now just save your stopwords in a file named *german.stop*.

The ispell-bit is a touch more tricky. Here's how you'll get what you need – again assuming you have installed ispell; you may need to adjust the paths, depending on your system layout and you should edit the Makefile if you want something specific (like support for Austrian vocabulary):

```
mkdir ~/files/ispell
cd ~/files/ispell
wget http://j3e.de/ispell/igerman98/dict/igerman98-20030222.tar.bz2
bunzip2 igerman98-20030222.tar.bz2
tar -xvf igerman98-20030222.tar
cd igerman98-20030222/
joe Makefile
make
sort -u -t/ +0f -1 +0 -T /usr/tmp -o german.med all.words
```

Now you should fire up you favourite text editor and edit *german.med*: First make sure you recode it to UTF-8, then replace the ispell-style Umlauts and ß with the corresponding proper characters (e.g. a"->ä, sS->ß,...), finally copy the two files to your ispell-dictionary directory:

```
cp german.aff german.med /usr/lib/ispell/
```

As for the stopwords-file *german.stop*, I prefer to place that in /var/lib/pgsql/data/contrib/, as it's got nothing to do with iSpell, but there's no harm in storing it alongside the *.med* and *.aff*-files.

III. Compiling the German Snowball Stemmer

The last bit of preparation would be the Snowball Stemmer Algorithm. Just change into the PostgreSQL-Source directory again and go to the contrib/tsearch2/gendict directory; then proceed like this (more info on this procedure can be found here):

```
wget http://www.snowball.tartarus.org/german/stem.c
wget http://www.snowball.tartarus.org/german/stem.h
./config.sh -n de -s -p german -i -v -c stem.c -h stem.h -C'Snowball stemmer for
German'
cd ../../dict_de/
make
make install
```

IV. Configuration of tsearch2

First configure the stemmer-dictionary by executing the sql-file prepared in the previous chapter:

```
[path_to_pg_binaries]/pgsql -d [your_db] -U [your_usr]  < /[path_to]/
contrib/dict_de.sql
```

You'll have to make one tiny update to the stemmer-setup in order to add the stopwords-functionality; execute the following SQL (alter the path if necessary):

```
UPDATE pg_ts_dict
      SET dict_initoption='/var/lib/pgsql/data/contrib/german.stop'
      WHERE dict_name = 'de');
```

Now you'll want to execute this next bit of SQL in your database to get a working german tsearch2-

configuration (check and adjust the paths where necessary). Please note, that in this initial configuration of the *de_ispell*-dictionary, we'll be using the stemmer's stopwords file as a place holder; we'll have to update that once we've got the appropriate file.

```
INSERT INTO pg_ts_cfg (ts_name, prs_name, locale)
               VALUES ('default_german', 'default', 'de_DE.UTF-8');
INSERT INTO pg_ts_dict
            (SELECT 'de_ispell',
                    dict_init,
                    'DictFile="/usr/lib/ispell/german.med",'
                    'AffFile="/usr/lib/ispell/german.aff",'
                    'StopFile="/var/lib/pgsql/data/contrib/german.stop"',
                       dict_lexize
                FROM pg_ts_dict
                WHERE dict_name = 'ispell_template');
SELECT set_curdict('de_ispell');
INSERT INTO pg_ts_cfgmap (ts_name, tok_alias, dict_name)
               VALUES ('default_german', 'lhword', '{de_ispell,de}');
        INSERT INTO pg_ts_cfgmap (ts_name, tok_alias, dict_name)
               VALUES ('default_german', 'lpart_hword', '{de_ispell,de}');
        INSERT INTO pg_ts_cfgmap (ts_name, tok_alias, dict_name)
               VALUES ('default_german', 'lword', '{de_ispell,de}');
INSERT INTO pg_ts_cfgmap
               VALUES ('default_german', 'url', '{simple}');
        INSERT INTO pg_ts_cfgmap
               VALUES ('default_german', 'host', '{simple}');
        INSERT INTO pg_ts_cfgmap
               VALUES ('default_german', 'sfloat', '{simple}');
        INSERT INTO pg_ts_cfgmap
               VALUES ('default_german', 'uri', '{simple}');
        INSERT INTO pg_ts_cfgmap
               VALUES ('default_german', 'int', '{simple}');
        INSERT INTO pg_ts_cfgmap
               VALUES ('default_german', 'float', '{simple}');
        INSERT INTO pg_ts_cfgmap
               VALUES ('default_german', 'email', '{simple}');
        INSERT INTO pg_ts_cfgmap
               VALUES ('default_german', 'word', '{de_ispell,de}');
        INSERT INTO pg_ts_cfgmap
               VALUES ('default_german', 'hword', '{simple}');
        INSERT INTO pg_ts_cfgmap
               VALUES ('default_german', 'nlword', '{simple}');
        INSERT INTO pg_ts_cfgmap
               VALUES ('default_german', 'nlpart_hword', '{simple}');
        INSERT INTO pg_ts_cfgmap
               VALUES ('default_german', 'part_hword', '{simple}');
        INSERT INTO pg_ts_cfgmap
               VALUES ('default_german', 'nlhword', '{simple}');
        INSERT INTO pg_ts_cfgmap
               VALUES ('default_german', 'file', '{simple}');
        INSERT INTO pg_ts_cfgmap
               VALUES ('default_german', 'uint', '{simple}');
        INSERT INTO pg_ts_cfgmap
               VALUES ('default_german', 'version', '{simple}');
```

V. A Stopwords-File for the ispell-Dictionary

Now our setup should be working to some extent. In order for the ispell-dictionary to work properly, we'll have to give it a new stopwords-file. I've written a short Perl-script to aid in converting the plain file used for the Snowball stemmer. Save the following script as *sw4is.pl* in your home-directory, edit to match your settings and chmod +x it.

```
#!/usr/bin/perl
# converts plain stemmer-stopwordsfile to ispell-stopwordsfile
# usage: ./sw4is.pl [language].stop.stemmer|sort|uniq > [language].stop.ispell
```

```
# Configuration
 my $server = '127.0.0.1';
 my $database = 'mypgsqldb';
 my $dbuser = 'postgres';
 my $dbpass = 'mypassword';
 my $dict = 'de_ispell';
# Do no change anything below this line
 my $infile = shift;
 use DBI;
 my $dsn = "DBI:Pg:dbname=$database;host=$server";
 my $dbh = DBI->connect($dsn, $dbuser, $dbpass)
   or die "Couldn't connect to database: " . DBI->errstr;
 my $sth = $dbh->prepare('select lexize(?,?) as lexeme;')
   or die "Couldn't prepare statement: " . $dbh->errstr;

 open (INFILE,"$infile") || die "  Error opening infile $infile.\n";

 while ($stopword = <INFILE>) {
   chomp($stopword);
   $sth->execute($dict,$stopword)
     or die "Couldn't execute statement: ".$sth->errstr;
   $sth->bind_columns( undef, \$lexeme );
   $sth->fetch();
   $lexeme=substr($lexeme,1,length($lexeme)-2);
   $lexeme=~ s/,/\n/g;
   if (length($lexeme)==0) {
     $lexeme = $stopword;
   }
   print "$lexeme\n";
 }
 $sth->finish();
 close(INFILE);
 $dbh->disconnect();
```

Now you can generate a working ispell-stopwords-file thus:

```
~/sw4is.pl /var/lib/pgsql/data/contrib/german.stop|sort|\
uniq>/var/lib/pgsql/data/contrib/german.stop.ispell
```

Update your ispell-configuration to use this new stopwords-file:

```
UPDATE public.pg_ts_dict
SET dict_initoption =
'DictFile="/usr/lib/ispell/german.med",AffFile="/usr/lib/ispell/german.aff",Stop
File="/var/lib/pgsql/data/contrib/german.stop.ispell"'
WHERE dict_name = 'de_ispell';
```

VI. Testing tsearch2

Connect to your database and run some query like this:

```
my_unicode_db=# select * from ts_debug('PostgreSQL ist weitgehend konform mit dem SQL92/SQL99-
Standard, d.h. alle in dem Standard geforderten Funktionen stehen zur Verfügung und verhalten sich
so, wie vom Standard gefordert; dies ist bei manchen kommerziellen sowie nichtkommerziellen SQL-
Datenbanken bisweilen nicht gegeben.');
```

ts_name	tok_type	description	token	dict_name	tsvector
default_german	lword	Latin word	PostgreSQL	{de_ispell,de}	'postgresql'
default_german	lword	Latin word	ist	{de_ispell,de}	
default_german	lword	Latin word	weitgehend	{de_ispell,de}	'weitgehend'
default_german	lword	Latin word	konform	{de_ispell,de}	'konform'
default_german	lword	Latin word	mit	{de_ispell,de}	
default_german	lword	Latin word	dem	{de_ispell,de}	
default_german	file	File or path name	SQL92/SQL99-Standard	{simple}	'sql92/sql99-standard'
default_german	host	Host	d.h	{simple}	'd.h'
default_german	lword	Latin word	alle	{de_ispell,de}	
default_german	lword	Latin word	in	{de_ispell,de}	
default_german	lword	Latin word	dem	{de_ispell,de}	
default_german	lword	Latin word	Standard	{de_ispell,de}	'standard'

default_german	lword	Latin word	geforderten	{de_ispell,de}	'gefordert'
default_german	lword	Latin word	Funktionen	{de_ispell,de}	'funktionen'
default_german	lword	Latin word	stehen	{de_ispell,de}	'stehen'
default_german	lword	Latin word	zur	{de_ispell,de}	
default_german	word	Word	Verfügung	{de_ispell,de}	'verfügung'
default_german	lword	Latin word	und	{de_ispell,de}	
default_german	lword	Latin word	verhalten	{de_ispell,de}	'halten' 'verhalten'
default_german	lword	Latin word	sich	{de_ispell,de}	
default_german	lword	Latin word	so	{de_ispell,de}	
default_german	lword	Latin word	wie	{de_ispell,de}	
default_german	lword	Latin word	vom	{de_ispell,de}	
default_german	lword	Latin word	Standard	{de_ispell,de}	'standard'
default_german	lword	Latin word	gefordert	{de_ispell,de}	'fordern' 'gefordert'
default_german	lword	Latin word	dies	{de_ispell,de}	
default_german	lword	Latin word	ist	{de_ispell,de}	
default_german	lword	Latin word	bei	{de_ispell,de}	
default_german	lword	Latin word	manchen	{de_ispell,de}	
default_german	lword	Latin word	kommerziellen	{de_ispell,de}	'kommerziell'
default_german	lword	Latin word	sowie	{de_ispell,de}	
default_german	lword	Latin word	nichtkommerziellen	{de_ispell,de}	'nichtkommerziell'
default_german	lhword	Latin hyphenated word	SQL-Datenbanken	{de_ispell,de}	'sql' 'datenbanken' 'sql-datenbank'
default_german	lpart_hword	Latin part of hyphenated word	SQL	{de_ispell,de}	'sql'
default_german	lpart_hword	Latin part of hyphenated word	Datenbanken	{de_ispell,de}	'datenbanken'
default_german	lword	Latin word	bisweilen	{de_ispell,de}	'bisweilen'
default_german	lword	Latin word	nicht	{de_ispell,de}	
default_german	lword	Latin word	gegeben	{de_ispell,de}	'geben' 'gegeben'

ts_debug will allow you to find out exactly what's happening with your input if you'd e.g. use to_tsvector to convert it to lexemes.

One more thing: Compound-Words like “nichtkommerziellen” are correctly stemmed, but they are not broken up into their compounds. There's actually a patch available for this at http://www.sai.msu.su/~megera/postgres/gist/tsearch/V2/, but I haven't yet gotten around to trying that one out.

If you need help regarding the actual indexing of data in your database, please refer to the official tsearch2-docs.

Markus Wollny <markus.wollny@email.de> - 07/12/04

VII. Changes/Remarks

03/12/04: Oleg Bartunov came up with this shell script as a replacement for the Perl-script from V.:

```
cat stop-words-file-for-ispell |\
awk " { print \"select lexize('de','\"\$1\"');\" }" |\
  psql aa -P 'tuples_only'|\
  grep -v '^$' |\
  tr -d ' {}' > stop-words-file-for-stemmer

here 'aa' is a database where tsearch2 is installed without stop-words
```

07/12/04: Added hint regarding recoding of german.med to UTF-8 and replacing the special characters in this dictionary in order to work with tsearch2; changed pg_ts_cfgmap for 'word' from '{simple}' to '{de_ispell,de}'. Thanks to Peter Alberer for this hint.