dbext for Vim

David Fishburn

http://www.vim.org/scripts/script.php?script_id=356

dbext

- Database extension
- Database agnositc
- Allows you to execute SQL and query databases without having to leave your editor
- Speed up development by having tools at your finger tips via commands, maps and menu items

Features

- Execute any SQL statement and show the results (if any)
- Describe Tables, Procedures
  - Useful for variable definition
- Generate Table, Procedure, View and Column lists
  - Object lists for quick lookups
  - Column lists for SELECT statements

OMNI Completion

- Integrates with Vim's built in OMNI completion (or Intellisense)
- Completes based on SQL syntax
  - Vim by default has 13 different SQL syntax files
- If dbext is available completes dynamically from your database
  - Tables, procedures, views, columns
  - Assists writing new SQL statements

Database Types

- Sybase SQL Anywhere (ASA)
- Sybase SQL Anywhere UltraLite (ULTRALITE)
- Sybase Adaptive Server Enterprise (ASE)
- Oracle (ORA)
- Oracle Rdb (RDB)
- Microsoft Sql Server (SQLSRV)
- IBM DB2 (DB2)

- MySQL (MYSQL)
- PostgreSQL (PGSQL)
- SQLite (SQLITE)
- Firebird (FIREBIRD)
- Ingres (INGRES)
- Interbase (INTERBASE)

- The following are only available with the Perl DBI extension plug.
- It requires a Perl enabled Vim (i.e. echo has('perl'))
- Perl DBI (DBI)
- Perl DBI::ODBC (ODBC)

Native Types

- All the non-Perl types access the database by shelling out and running the standard binaries
  - isql, psql, mysql, dbisql, sqlplus, osql
  - Means the database client software must be installed on your machine
- Results are displayed in a Vim buffer

Perl DBI and ODBC

- Perl offers many advantages over the native types
  - Loaded in memory and maintains a connection
    - Faster between uses and can commit and rollback
  - The output is formatted a better
  - Only the Perl module must be installed not the native client which may not be available on all platforms
  - I use this almost exclusively

Profiles

- Each buffer can be connected to a different database
- Connection information can be saved in your .vimrc
- Modelines can be set in each file
- Autocmds can be used to set profiles based on directory information
- Default profiles can be specified

Profile Examples

- let g:dbext_default_profile_ASA_SMM = 'type=ODBC:user=:passwd=:dsnname=smm'
- let g:dbext_default_profile_ASA = 'type=ASA:user=dba:passwd=sql'
- let g:dbext_default_profile_mysql_local = 'type=MYSQL:user=root:passwd=:dbname=mysql:extra=-t'

Modeline

- A comment at the beginning or end of your file allows you to customize various settings
- // dbext:profile=ASA_SMM

Autocmds

- augroup smm
- au!
- autocmd bufread */\(HostedProdEnv\|SMM\)/* DBSetOption profile='ASA_SMM'
- autocmd bufread */\(HostedProdEnv\|SMM\)/* let g:dbext_default_variable_def_regex = DB_listOption('variable_def_regex')|
- \ let b:dbext_variable_def_regex = g:dbext_default_variable_def_regex . ',\<\(p_\|lv_\)\w\+\>'|
- augroup end

Default Profile

- If you usually connect to the same database you can setup a default profile if the first time you use dbext connection information was not already specified
  - let g:dbext_default_profile = 'ASA_SMM'

General Commands

- Execute arbitrary SQL
- Select from table (under cursor)
- List tables
- List columns (for pasting)
- Describe Objects
- Reuse variables

FileType Support

- Each filetype, SQL, Perl, C, Java, ... has standard string formats
- Variables can be embedded within the strings
- Concatenation can be used to build a SQL statement
  - “Select 1, 2 ” + “ from myTable”
- Prepared statements can also be used
  - “Select 1, 2 from myTable where col1 = $myVar”

FileType Rules

- Based on the filetype dbext can determine how to parse a string, remove any concatenation and find variables
- Prompt the user for the variables
- Execute the statement
- Display the results

Perl FileType

- Prompt for $mycol1 and $cols[1]

```
my $sql    = "SELECT c1, c2
              FROM table1
             WHERE c1 = $mycol1
              AND c2 = " . $cols[1];
```

Java FileType

- Prompt for both ?s and “user”

```
String query = "SELECT c1, c2 " +
               "  FROM table1 " +
               " WHERE c3 = ? " +
               "   AND c4 = ? " +
               "   AND c5 = " + user +
               "   AND c6 > 4 ";
```

Help

- I need some help implementing:
  - DisconnectAll
  - Reconnect after timeout interval
- Unfortunately, it takes me a long time to write Perl, but you guys should whip this off like breathing!