Embedded SQL in PostgreSQL

Dr. Michael Meskes, michael.meskes@credativ.de

7. Dezember 2010

Michael Meskes

- Seit 1993 Freie Software
- Seit 1994 Linux
- Seit 1995 Debian GNU/Linux
- Seit 1998 PostgreSQL

Michael Meskes

- 1992 - 1996 Promotion
- 1996 - 1998 Projektleiter
- 1998 - 2000 Niederlassungsleiter
- seit 2000 Geschäftsführer

credativ

Embedded SQL

Embedded SQL ist eine Spracherweiterung von SQL, mit der es möglich ist, SQL-Anweisungen innerhalb einer strukturierten oder objektorientierten Programmiersprache [...] auszuführen.

Quelle: Wikipedia

Konzepte von Embedded SQL

- SQL-Standard
- Normales C-Programm mit eingebettetem SQL
- Präkompiler wandelt SQL-Code in Library-Aufrufe um
- Prüft Syntax des SQL-Code beim Kompilieren
- Erleichtert Datenaustausch zwischen C und SQL

Präkompiler - ECPG

- Kontrolle der SQL-Syntax
- Identisch mit SQL-Syntax des Backends
- Umwandlung von SQL nach C

Libraries

- libecpg - SQL Umsetzung
- libpgtypes - SQL Datentypen
- libecpg_compat - Kompatibilität

Programme übersetzen

```
ecpg prog1.pgc
# (erzeugt prog1.c)

cc -c -I/usr/include/postgresql prog1.c
# (erzeugt prog1.o)

cc -o prog prog1.o ... -lecpg
# (erzeugt prog)
```

Beispiel der C-Syntax

```
#include <stdlib.h>
#include <stdio.h>

int main(int argc, char *argv[])
{
    ...
    puts("Hallo");
    ...
    EXEC SQL UPDATE tabelle SET a = a + 1;
    ...
}
```

Übersetzt

```
/* Processed by ecpg (4.6.0) */
/* These include files are added by the preprocessor */
#include <ecpglib.h>
#include <ecpgerrno.h>
#include <sqlca.h>
/* End of automatic include section */

#line 1 "a.pgc"
#include <stdlib.h>
#include <stdio.h>

int main(int argc, char *argv[])
{
    ...
    puts("Hallo");
    ...
    { ECPGdo(__LINE__, 0, 1, NULL, 0, ECPGst_normal,
        "update tabelle set a = a + 1", ECPGt_EOIT, ECPGt_EORT);}
#line 9 "a.pgc"
    ...
}
```

Verbinden

```
EXEC SQL CONNECT TO dbname AS connname USER user;
```

Zum Beispiel:

```
EXEC SQL CONNECT
    TO tcp:postgresql://sql.beispiel.de:5432/mydb
    AS myconn USER peter IDENTIFIED BY 'geheim';
```

Verbindung beenden

```
EXEC SQL DISCONNECT;  -- aktuelle Verbindung
EXEC SQL DISCONNECT connname;
```

Autocommit

- Precompiler-Option
- In der Standardkonfiguration wird kein automatisches `COMMIT` ausgeführt.
- Mit Option wie im Backend.

Befehle mit Hostvariablen (1)

```
EXEC SQL BEGIN DECLARE SECTION;
int v1;
char v2[100];
EXEC SQL END DECLARE SECTION;

v1 = 42;
strcpy(v2, "text");

EXEC SQL INSERT INTO test VALUES (:v1, :v2);
EXEC SQL COMMIT;
```

Befehle mit Hostvariablen (2)

```
EXEC SQL BEGIN DECLARE SECTION;
int v1, ind1;
VARCHAR v2;
EXEC SQL END DECLARE SECTION;

...

EXEC SQL DECLARE foo CURSOR FOR SELECT a, b FROM test;

...

do {
    ...
    EXEC SQL FETCH NEXT FROM foo INTO :v1:ind1, :v2;
    ...
} while (...);
```

Übersetzt

```
...
#line 2 "a.pgc"
 int v1 , ind1 ;
#line 3 "a.pgc"
  struct varchar_v2_1  { int len; char arr[ 40 ]; }  v2 ;
/* exec sql end declare section */
#line 4 "a.pgc"
...
/* declare foo cursor for select a , b from test */
#line 6 "a.pgc"
{ ECPGdo(__LINE__, 0, 1, NULL, 0, ECPGst_normal, "declare foo cursor for select a , b from test", ... }
#line 7 "a.pgc"
...
do {
    ...
    { ECPGdo(__LINE__, 0, 1, NULL, 0, ECPGst_normal, "fetch next from foo", ECPGt_EOIT,
        ECPGt_int,&(v1),(long)1,(long)1,sizeof(int),
        ECPGt_int,&(ind1),(long)1,(long)1,sizeof(int),
        ECPGt_varchar,&(v2),(long)40,(long)1,sizeof(struct varchar_v2_1),
        ECPGt_NO_INDICATOR, NULL , 0L, 0L, 0L, ECPGt_EORT);}
#line 11 "a.pgc"
    ...
} while (...);
```

Fehlerbehandlung: Callbacks

```
EXEC SQL WHENEVER SQLERROR STOP;

EXEC SQL WHENEVER SQLWARNING SQLPRINT;

EXEC SQL WHENEVER SQLERROR CALL func();

EXEC SQL WHENEVER NOT FOUND DO BREAK;
```

Migrationen

- EXEC SQL CONNECT
- SQL-Migration
- Variablen
- Anpassungen am Präkompiler möglich
- Kompatibilitätsschichten

Geschichte

- Linus Tolke
- Import nach Postgres Anfang 1998
- Kompatibilität 2003

Aktuelle Versionen

- Präkompiler 4.6.0
- Library 6.2
- Typlibrary 3.1
- Kompatibilitätslibrary 3.2

Weiterentwicklung

- Kontinuierlich
- Performance, z.B. Auto-Prepare
- Standardfeatures, z.B. SQLDA
- Kompatibilität!

Wir brauchen SIE!

Anregungen?
Feedback?
Wünsche?