实验 嵌入式pgSQL

主要内容

嵌入式pgSQL的介绍

ECPG环境配置

连接数据库

PGC文件的执行

- 预编译
- 编译与执行

PGC文件的编写

- 不涉及游标
- 涉及游标

主要内容

嵌入式pgSQL的介绍

ECPG环境配置

连接数据库

PGC文件的执行

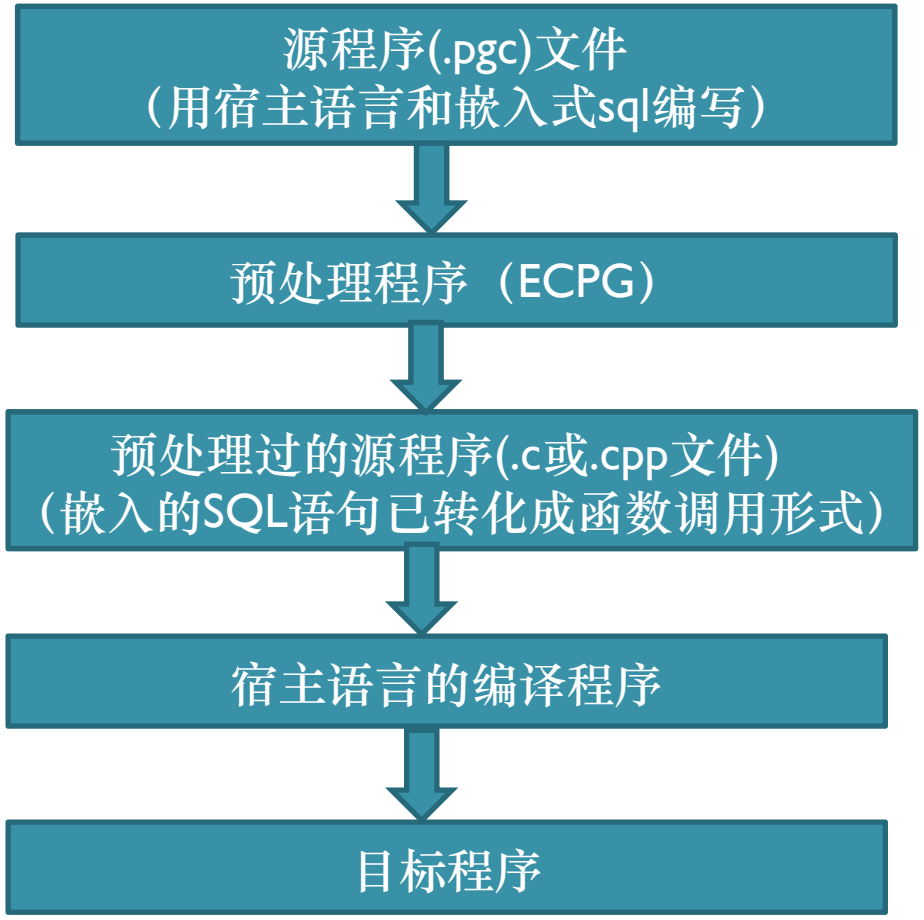
- 预编译
- 编译与执行

PGC文件的编写

- 不涉及游标
- 涉及游标

嵌入式pgSQL的介绍

pgSQL可以作为独立的数据语言以交互的方式使用，还可以作为子语言嵌入在宿主语言中使用。



主要内容

嵌入式pgSQL的介绍

ECPG环境配置

连接数据库

PGC文件的执行

- 预编译
- 编译与执行

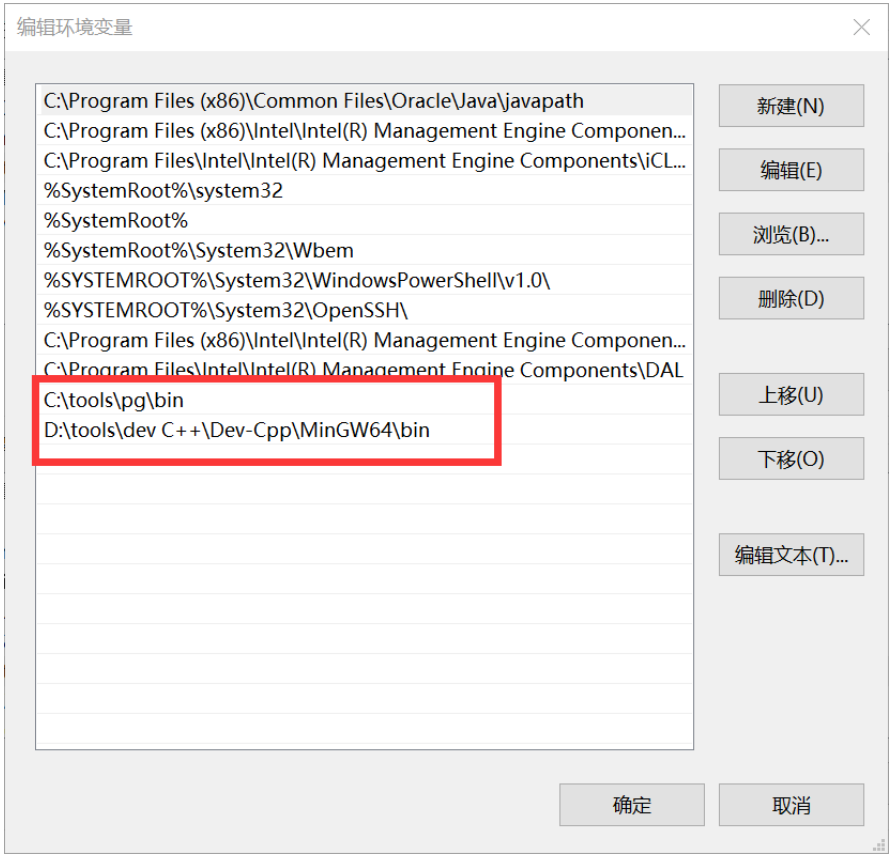
PGC文件的编写

- 不涉及游标
- 涉及游标

ECPG环境配置

ECPG是postgreSQL数据库内置的标准客户端编程接口，
用于将SQL嵌入以C编程语言编写的程序中。

将 ...dev C++\Dev-Cpp\MinGW64\bin 放到系统变量的path路径下。
将 …pg\bin 放到系统变量path路径下。



主要内容

嵌入式pgSQL的介绍

ECPG环境配置

连接数据库

PGC文件的执行

- 预编译
- 编译与执行

PGC文件的编写

- 不涉及游标
- 涉及游标

连接数据库

新建lab7.pgc文件

连接数据库：

EXEC SQL CONNECT TO *target* [AS *connection-name*][*USER user-name*];

为区分pgSQL语句和宿主语言，在pgSQL语句加：
前缀标识 ： EXEC SQL
结束标志 ： 分号“；”

target指定方法：

dbname[@hostname][:port]

连接本地exam数据库，端口为5432，密码为123456

exec sql connect to
“exam @127.0.0.1:5432” user “postgres” using “123456”

若不是连接本地服务器，则将127.0.0.1 替换成服务器地址。

Pgc文件如下：

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

int main(){
        exec sql whenever sqlwarning sqlprint;
        exec sql whenever sqlerror sqlprint;
        exec sql connect to "exam@127.0.0.1:5432" user
"postgres" using "123456";
exec sql disconnect;
return 0;
}
```

练习

- 编写.pgc文件，连接自己的数据库。

主要内容

嵌入式pgSQL的介绍

ECPG环境配置

连接数据库

PGC文件的执行

- 预编译
- 编译与执行

PGC文件的编写

- 不涉及游标
- 涉及游标

预编译

通过ECPG将.pgc 文件预编译成 .c文件。

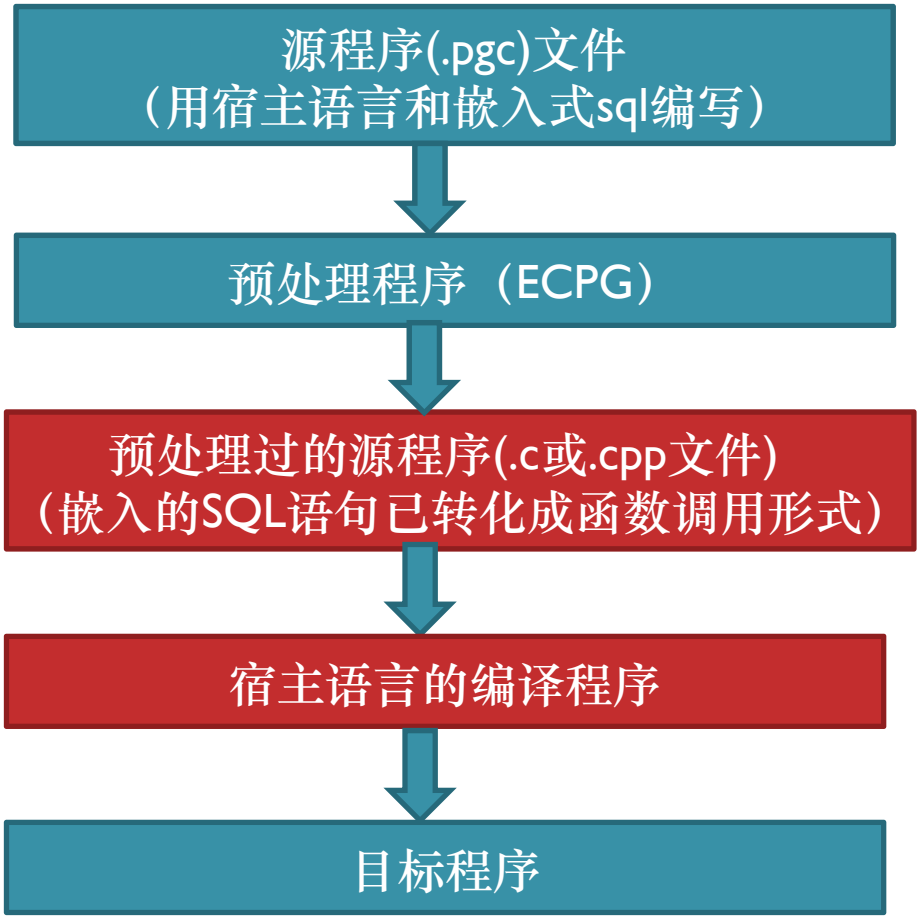
使用DOS命令行的方式:

ecpg –t –c –I [pg include路径] –o [c文件路径][pgc 文件路径]

例:

ecpg –t –c –I D:\tools\pg\include -o D:\text. c D:\text.pgc

编译与执行


源程序(.pgc)文件
（用宿主语言和嵌入式sql编写）
预处理程序（ECPG）
预处理过的源程序(.c或.cpp文件)
（嵌入的SQL语句已转化成函数调用形式）
宿主语言的编译程序
目标程序

编译与执行

将.c 文件编译成.exe 文件。

使用DOS命令行的方式：

gcc -I [pg的include路径] -Wall -g [c文件路径] -L [pg的lib路径] -lecpg -lpq -lpgtypes -o [exe输出路径]

执行生成的exe文件：

直接在dos下输入文件名即可。

练习

- 编写命令编译.pgc文件 -> 编译.c文件 -> 执行.exe文件

主要内容

嵌入式pgSQL的介绍

ECPG环境配置

连接数据库

PGC文件的执行

- 预编译
- 编译与执行

PGC文件的编写

- 不涉及游标
- 涉及游标

不涉及游标

以下三种情况不使用游标：

① 对于DDL/DPL语言
② DML中的insert、delete、update语句
③ DQL中select语句查询结果只有一个元组。

直接在sql语句中加入前缀标识exec sql 和结束标识分号“；”即可。

在examinee表中插入数据。

```
exec sql insert into examinee (eeid,eename,eeage)
                values(‘218811011014’,‘林小溪’,20);
```

- 对于select 语句，还需要增加into子句，指出值应该送到相应的共享变量。

```
exec sql select eename,grade
into :tt, :yy from examinee
where eeid=:mm;
```

定义的共享变量

通过变量将结果打印出来。

定义变量

```
EXEC SQL BEGIN DECLARE SECTION;
  int  mm;
  char tt[20]="abcd";
  int  yy;
EXEC SQL END DECLARE SECTION;
```

解决中文显示乱码的方法:

Dos下输入:

chcp 65001

查询eeid='218811011013'的考生姓名和年龄。

```
EXEC SQL BEGIN DECLARE SECTION;
  char id[12]="218811011013";
  char name[20];
  int age;
EXEC SQL END DECLARE SECTION;


EXEC SQL SELECT eename, eeage
         INTO  :name, :age
         FROM examinee
        WHERE eeid=:id;
EXEC SQL DISCONNECT;
```

涉及游标

DML中select语句查询结果中有多个元组。

查询examinee 表的所有学生姓名和成绩。

```
exec sql declare cursor for
  …
exec sql open quescr;
  …
exec sql close quescr;
```

查询考试成绩大于80分的考试信息

```
exec sql begin declare section;
        char seeid[20],seid[20];
        int sachieve;
        int req_achieve;
exec sql end declare section;
req_achieve = 80;
exec sql declare quescr2 cursor for
        select eeid, eid, achieve
        from eeexam
        where achieve > :req_achieve;
exec sql open quescr2;
while(1){exec sql fetch from quescr2
        into :seeid, :seid, :sachieve;
        if(No_more_tuples)break;
        printf("%s,%s,%d\n", seeid, seid, sachieve);
}
exec sql close quescr2;
exec sql disconnect;
}
```

卷游标的定义、推进和移动

EXEC SQL DECLARE queryc [NO]SCROLL CURSOR

exec sql fetch next from <游标名>into<变量名>

exec sql fetch prior from <游标名>into<变量名>

exec sql fetch first from <游标名>into<变量名>

exec sql fetch last from <游标名>into<变量名>

exec sql fetch relative -2 from <游标名>into<变量名>

exec sql fetch absolute 3 from <游标名> into <变量名>

exec move forward 2 from <游标名>

exec move backward 2 from <游标名>

使用卷游标查询

```
exec sql fetch absolute -2 from quescr2
                    into :seeid, :seid, :sachieve;
if(!No_more_tuples){
                    printf("%s,%s,%d\n", seeid, seid, sachieve);
          }
```

动态SQL语句

- 实际应用中SQL语句只能在实际运行时才能完全确定。

更新XX考生的信息，将其学院改为XX学院。

```
exec sql begin declare section;
char *tt = "update examinee set eedepa = ? where eename = ?;";
char *ttc = "commit;";
char name[20];
char depa[20];
exec sql end declare section;
printf("请输入你要修改的考生姓名:");
scanf("%s",name);
printf("考生姓名：%s\n",name);
printf("请输入修改后的院系名:");
scanf("%s",depa);
printf("修改后的院系名称:%s\n",depa);

exec sql prepare mmtt from :tt;
exec sql execute mmtt using :depa, :name;
exec sql execute immediate :ttc;
exec sql disconnect;
```