Программируемый клиент ORACLE

Технология Pro C/C++

Два вида программного интерфейса

- Предкомпилятор
- CALL- интерфейс

Достоинства и недостатки подходов

СПРАВОЧНИК ВЫЗОВОВ

Call –интерфейс требует детального знания процедур и функций:

- названия;
- количество пар-ров;
- тип параметров;
- знание кодов возврата;
- обработка исключительных ситуаций;
- и т.п.

Время и трудозатраты на разработку - возрастает

ПРЕДКОМПИЛЯТОР

Использовать проще:

- наглядность;
- понятность;
- структурированность

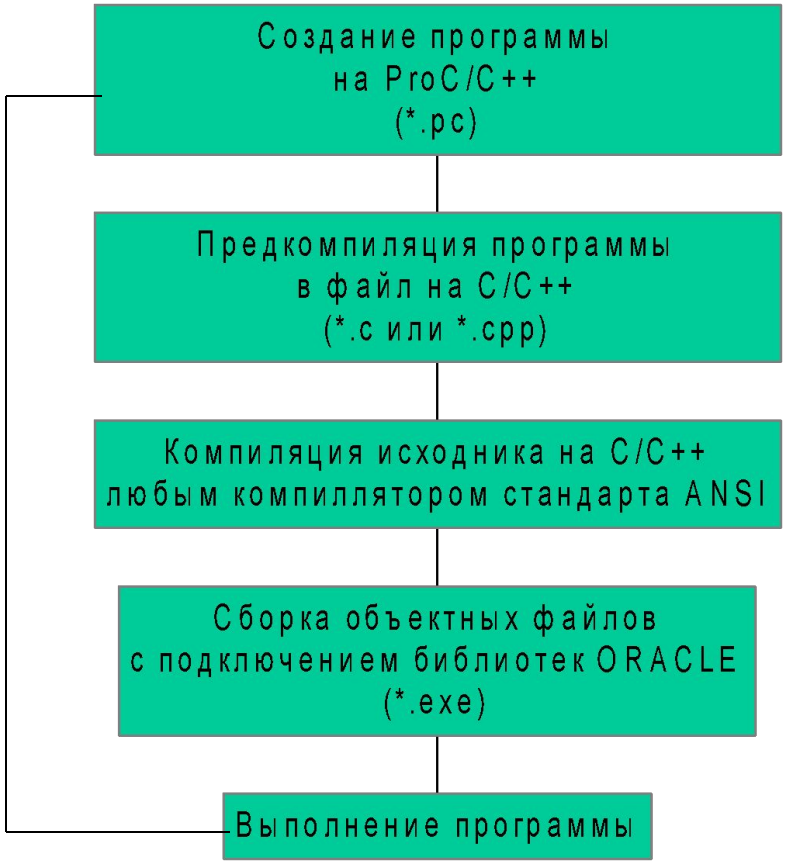
Однако, мобильность программ – меньше, так как требуется предкомпилятор Oracle, позволяющий строить код исходной программы на C/C++, Fortran, Ada, Cobol, Pascal и др.

Возможности предкомпиляции

- Один вызов к Oracle автоматически преобразуется в несколько вызовов процедур(функций).
- Одна программа может применяться для работы с разными БД
- Несколько программ могут быть отдельно предкомпилированы и совместно выполнены (собраны).

Этапы разработки приложений

отладка



Правила и соглашения технологии Pro C/C++

- В программу на Pro C/C++ может быть включен любой оператор SQL
- Перед всеми операторами SQL ставится префикс **EXEC SQL**
- (для некоторых уникальных конструкций EXEC ORACLE).
- Операторы SQL делятся на **декларативные** и **выполняемые.**
- После исполнения выполняемых операторов коды возврата заносятся в SQLCA
- Декларативные операторы не изменяют SQLCA

Структура программы на Pro C/C++

Секция DECLARE

- Единственная
- Может иметь локальную или глобальную видимость

EXEC SQL BEGIN DECLARE SECTION;

Описание переменных обмена данными

EXEC SQL END DECLARE SECTION;

Пример

```
EXEC SQL BEGIN DECLARE SECTION;
Int n,no;
Char name[11];
EXEC SQL END DECLARE SECTION;
Void main(){
……

EXEC SQL select id, fio into :n,:name
                from personal
                where num_o = :no;
```

Для отличия переменных от атрибутов в SQL запросах используется символ ‘:’, перед именем переменной.

Для строк переменной длины в секции DECLARE используется специальный тип VARCHAR.

Пример
VARCHAR j[40];

Автоматически порождается
```
Struct {
        unsigned short int len;
        unsigned char arr[40];
} j;
```

Можно использовать в программе
j.len – фактическая длина стринга
j.arr – указатель на массив содержимого

INCLUDE SQLCA

- EXEC SQL INCLUDE SQLCA;

Область SQLCA содержит:

- флаги предупреждений;
- информацию о событиях;
- коды ошибок;
- текст диагностики ошибок;
- и др. служебную информацию.

Пример

```
…..
If( sqlca.sqlcode == 0 ) continue; //успешное завершение sql
  else if( sqlca.sqlcode < 0 ) cout << “Ошибка выполнения”;
      else cout << “ Выбрана последняя строка”;
```

CONNECT

Соединение с ORACLE Server

EXEC SQL CONNECT :oralogpass;

Пример

```
EXEC SQL BEGIN DECLARE SECTION;
Varchar oralogpass[40];
EXEC SQL END DECLARE SECTION;
…….
Strcpy(oralogpass.arr,”o01/o01”);
oralogpass.len=strlen(oralogpass.arr);

EXEC SQL CONNECT :oralogpass;
…….
```

Тело программы

- Стандартные операторы языка C/C++ ;
- Операторы SQL (select,delete,insert,update);
- Контроль распределенной обработки данных

EXEC SQL DECLARE <имя> STATEMENT;

- Связь и управление распределенной БД

EXEC SQL DECLARE <имя> DATABASE;

- Изменение стандартных установок

EXEC ORACLE OPTION (option=значение);

Множественный и единичный выбор с использованием массивов м указателей.

Оператор	массив(указатель)	простая переменная
Select заголовок	можно	можно
Select .. Where	нельзя	можно
Insert.. Values()	можно	можно
Update .. Set.. where	можно однов.	можно однов.

Примеры

```
EXEC SQL BEGIN DECLARE SECTION;
char name[100][100];
int n[100];
Float sal[100];
EXEC SQL END DECLARE SECTION;
……..
EXEC SQL SELECT  fio,id into :name,:n FROM personal; // ok!
EXEC SQL SELECT  fio,id into :name,:n FROM personal
                  WHERE many = :sal; // неверный оператор
EXEC SQL INSERT INTO personal(fio,id,many)
                VALUES(:name, :n, :sal); // ok!
EXEC SQL UPDATE SET many = :sal WHERE id = :n; //ok!
EXEC SQL UPDATE SET many = :sal WHERE id = :n[0]; // no!
```

Оператор COMMIT WORK завершает транзакцию

EXEC SQL COMMIT WORK;

Оператор COMMIT WORK RELEASE осуществляет DISCONNECT

EXEC SQL COMMIT WORK RELEASE;

Пример программы, которая позволяет задать (с клавиатуры) номера сотрудников и их оклады, а затем добавляет эти данные в таблицу

```
#include <stdio.h>
#include <string.h>
EXEC SQL BEGIN DECLARE SECTION;
varchar log_pass[40];
int loop,n[100];
float many[100];
EXEC SQL END DECLARE SECTION;
EXEC SQL INCLUDE sqlca.h;
main() {
  int i,ret;
  strcpy(log_pass.arr,”o01/o01@stud”);
  log_pass.len = strlen(log_pass.arr);
EXEC SQL CONNECT :log_pass;
for(i=0;i<100;i++) {
    ret=scanf(“%d %f\n”, &n[i],&many[i]);
    if(ret == EOF|| ret ==0|| n[i]==0) break;
}
   loop=i;
   EXEC SQL FOR :loop
                    INSERT INTO sotrud (Tnum,okl)
                    VALUES(:n,:many);
  EXEC SQL COMMIT WORK RELEASE;
}
```