

Отчёт по лабораторной работе

Дисциплина: Базы данных

Тема: Хранимые процедуры

Выполнил студент гр. 43501/1

А.О. Перешеин

(подпись)

Руководитель

А.В. Мяснов

(подпись)

“__” _____ 2015 г.

Санкт-Петербург

2015

1. Цели работы

Знакомство с возможностями реализации более сложной обработки данных на стороне сервера с помощью хранимых процедур.

2. Программа работы

- Изучить возможности языка PSQL
- Создать две хранимые процедуры в соответствии с **индивидуальным заданием**, полученным у преподавателя
- Выложить скрипт с созданными сущностями в svn
- Продемонстрировать результаты преподавателю

3. Реализация хранимых процедур

Структура хранимой процедуры (ХП) имеет следующий вид:

```
CREATE PROCEDURE procname
  [(inparam> [, <iinparam> ...])]
  [RETURNS (<ioutparam> [, <ioutparam> ...])]
AS
  [<ideclarations>]
BEGIN
  [<iPSQL statements>]
END

<iinparam>      ::= <iparam_decl> [{= | DEFAULT} value]
<ioutparam>    ::= <iparam_decl>
```

ХП может иметь входные, выходные и локальные переменные. Возможно использование в теле процедуры следующих конструкций ветвления и циклов:

```
if (not (a <> b)) then
  MyVariable = 'Equal';
else
  MyVariable = 'Not equal';

case <iexpression>
  when <iexp1> then <iresult1>
  when <iexp2> then <iresult2>
  ...
  [else <idefaultresult>]
end

while ( Counter > 12 ) do
```

Также возможно записывать результаты запросов на языке SQL-DML в переменные:

```
FOR <iselect-stmt>
  INTO <ivar> [, <ivar> ...]
  [AS CURSOR name]
DO
  <ipsql-stmt>

<iselect-stmt> ::= A valid SELECT statement.
<ivar>         ::= A PSQL variable name, optionally preceded by ":"
<ipsql-stmt>  ::= A single statement or a block of PSQL code.
```

- Все диски, которые были проданы более заданного количества раз, наградить заданной наградой.

Перед созданием хранимой процедуры создадим атрибут award в таблице cds:

```
alter table cds add award varchar(15);
select * from cds;
```

CODE	RELEASE_YEAR	SOLD	STATUS	AWARD
=====	=====	=====	=====	=====
679137227	1990	100000000	3	<null>
781676650	2000	100000000	3	<null>
329751487	1986	200000000	3	<null>
105520746	2013	1700000	3	<null>
714679272	1996	600000	1	<null>

Содержание хранимой процедуры:

```
create or alter procedure LAB5_P1 (
  AWARD varchar(15) not null,
  SOLD_COPY integer not null)
as
begin
  update cds set cds.award = :award where cds.code in
    (select cds.code from cds where cds.sold > :sold_copy);
end
```

Выполнение XII:

```
execute procedure lab5_p1('MTV award', 99999);
execute procedure lab5_p1('grammy', 9999999);
select * from cds;
```

CODE	RELEASE_YEAR	SOLD	STATUS	AWARD
=====	=====	=====	=====	=====
679137227	1990	100000000	3	grammy
781676650	2000	100000000	3	grammy
329751487	1986	200000000	3	grammy
105520746	2013	1700000	3	MTV award
714679272	1996	600000	1	MTV award

В данном случае мы наградили наградой “grammy” диски, которые были проданы в количестве больше 9999999 копий и наградой “MTV award” для дисков с объемом продаж больше 99999 копий.

- Все альбомы заданного жанра заданного артиста причислить к еще одному заданному жанру.

Для выполнения заданной хранимой процедуры создадим атрибут subgenre в таблице album:

```
alter table albums add subgenre varchar(25);
select alb_id, album, genre, subgenre from albums;
```

ALB_ID	ALBUM	...	GENRE	SUBGENRE
=====	=====	==	=====	=====
1	Cowboys from Hell		1	<null>
2	Tales From The Thousand Lakes		1	<null>
3	Black Winter Day		1	<null>
4	Reign In Blood		1	<null>

5	Icon	2	<null>
6	Crimson	1	<null>

Содержание XII:

```
CREATE OR ALTER procedure LAB5_P2 (
    GENRE type of column GENRES."GEN_ID" not null,
    SUBGENRE varchar(25),
    ARTIST type of column ARTISTS.ART_ID not null)
as
begin
    update albums set albums.subgenre = :subgenre
    where albums.alb_id in
    (select albums.alb_id from artists, cds, cds_alb, albums, art_alb
    where artists.art_id = art_alb.art_id and cds_alb.code = cds.code and
    art_alb.alb_id = albums.alb_id and cds_alb.alb_id = albums.alb_id and
    albums.genre = :genre and artists.art_id = :artist);
end
```

Выполнение XII:

```
execute procedure lab5_p2(1, 'thrash metal', 1);
execute procedure lab5_p2(1, 'death metal', 2);
execute procedure lab5_p2(1, 'thrash metal', 3);
execute procedure lab5_p2(2, 'rap', 4);
execute procedure lab5_p2(1, 'death metal', 5);
select alb_id, album, genre, subgenre from albums;
```

ALB_ID	ALBUM	...	GENRE	SUBGENRE
=====	=====	==	=====	-----
1	Cowboys from Hell		1	thrash metal
2	Tales From The Thousand Lakes		1	death metal
3	Black Winter Day		1	death metal
4	Reign In Blood		1	thrash metal
5	Icon		2	rap
6	Crimson		1	death metal

В данном варианте мы заполнили атрибут subgenre для всех имеющихся в БД альбомов, исходя из «основного» жанра альбома.

- Во все плейлисты, содержащие более заданного количества песен заданного автора добавить треки с заданного диска.

Алгоритм работы XII следующий:

Сначала получаем текущее название плейлиста требуемого исполнителя, просматривая подряд все доступные плейлисты, с помощью конструкции for select <...> into :s. и если для текущего плейлиста выполняется заданное условие (см. постановку задания), то переходим к записи в этот плейлист треков с заданного диска (количество треков, подлежащих записи, определяется с помощью for select <...> into :i, а сама запись выполняется через update or insert). Затем переходим к следующему плейлисту и так далее.

Содержание XII:

```
create or alter procedure LAB5_P3 (
    CD type of column CDS.CODE not null,
    ARTIST type of column ARTISTS.ART_ID not null,
    NUM_OF_TRACKS integer not null)
as
```

```

declare variable S varchar(25);
declare variable I integer;
begin
    for select playlists.name from playlists, pl_tr, tracks, tracklists,
        albums, artists, art_alb, cds_alb, cds where
        artists.art_id = :artist and
        artists.art_id = art_alb.art_id and cds_alb.code = cds.code and
        art_alb.alb_id = albums.alb_id and cds_alb.alb_id = albums.alb_id and
        albums.alb_id = tracklists.alb_id and tracks.tr_id = tracklists.tr_id and
        pl_tr.tr_id = tracks.tr_id and pl_tr.name = playlists.name
        group by playlists.name having count(*) > :num_of_tracks into :s
    do begin
        for select tracks.tr_id from
            tracks, tracklists, albums, cds_alb, cds
            where cds_alb.code = cds.code and cds_alb.alb_id = albums.alb_id and
            albums.alb_id = tracklists.alb_id and tracks.tr_id = tracklists.tr_id and
            cds.code = :cd into :i
        do begin
            update or insert into pl_tr values(:s, :i);
        end
    end
end

```

Исходная таблица с плейлистами:

```
SQL> select * from pl_tr;
```

NAME	TR_ID
home	21
home	22
home	23
home	24
home	25
home	26
home	27
home	32
home	33
home	34
list1	1
list1	2
list2	3
list2	4
list2	5
mp3	35
mp3	36
mp3	37
mp3	38
mp3	50
mp3	51
mp3	52
list4	13

Выполнение процедуры для случая, когда в плейлист 1-го автора, содержащий больше 2-х записей добавляются композиции с диска 679137227 (содержит 1 альбом). В итоге должно измениться содержание плейлиста list2:

```
execute procedure lab5_p3(679137227, 1, 2);
select * from pl_tr;
```

NAME	TR_ID
home	21
home	22
home	23
home	24
home	25
home	26
home	27
home	32
home	33
home	34
list1	1
list1	2
list2	3
list2	4
list2	5
mp3	35
mp3	36
mp3	37
mp3	38
mp3	50
mp3	51
mp3	52
list4	13

home	21
home	22
home	23
home	24
home	25
home	26
home	27
home	32
home	33
home	34
list1	1
list1	2
list2	3
list2	4
list2	5
mp3	35
mp3	36
mp3	37
mp3	38
mp3	50

NAME	TR_ID
=====	=====
mp3	51
mp3	52
list4	13
list2	1
list2	2
list2	6
list2	7
list2	8
list2	9
list2	10
list2	11
list2	12

Выполнение процедуры для случая, когда в плейлист 1-го автора, содержащий больше 1-ой записи добавляются композиции с диска 781676650 (содержит 2 альбома). В итоге должно измениться содержание плейлистов list1 и list2:

```
SQL> execute procedure lab5_p3(781676650, 1, 1);
SQL> select * from pl_tr;
```

NAME	TR_ID
=====	=====
home	21
home	22
home	23
home	24
home	25
home	26
home	27
home	32
home	33
home	34
list1	1
list1	2
list2	3
list2	4
list2	5
mp3	35
mp3	36
mp3	37

mp3	38
mp3	50

NAME	TR_ID
=====	=====
mp3	51
mp3	52
list4	13
list1	21
list1	22
list1	23
list1	24
list1	25
list1	26
list1	27
list1	28
list1	29
list1	30
list1	32
list1	33
list1	34
list2	21
list2	22
list2	23
list2	24

NAME	TR_ID
=====	=====
list2	25
list2	26
list2	27
list2	28
list2	29
list2	30
list2	32
list2	33
list2	34

Вариант, когда имеющиеся плейлисты не удовлетворяют условию изменения:

```
execute procedure lab5_p3(105520746, 4, 10);
select * from pl_tr;
```

NAME	TR_ID
=====	=====
home	21
home	22
home	23
home	24
home	25
home	26
home	27
home	32
home	33
home	34
list1	1
list1	2
list2	3
list2	4
list2	5
mp3	35
mp3	36
mp3	37

mp3	38
mp3	50

NAME	TR_ID
=====	=====
mp3	51
mp3	52
list4	13

4. Выводы

В ходе выполнения работы познакомились с основами языка PSQL, как подмножеством языка SQL для процедур, триггеров и блоков. Хранимые процедуры – это ряд операций над данными, которые реализуют общую для всех пользователей логику и не связаны с пользовательским интерфейсом. В связи с этими особенностями хранимые процедуры обычно выносятся на сервер, а пользователь только обращается к ним, при этом экономя время на ввод запроса и проверку его правильности. Также посредством ХП осуществляется контроль целостности данных.

В работе на примере создания 3-х ХП произведено использование входных, локальных переменных, с записью в последние результатов запроса `select`. С помощью конструкции `for select <...> do <...> into <...>` организован последовательный «проход» по данным таблиц с их непосредственным анализом (с точки зрения удовлетворения поставленного условия).

Используемые конструкции позволили реализовать достаточно сложные запросы, которые не могли быть выполнены посредством вложенных запросов. А использование входных параметров процедур, позволило внести ограничения для данных, подлежащих корректировке.