Санкт-Петербургский государственный политехнический университет

Институт информационных технологий и управления

Кафедра компьютерных систем и программных технологий

Отчет по лабораторной работе

По дисциплине «Базы данных»

«Хранимые процедуры»

Работу выполнил студент группы № 43501/4 Смирнов Л.А.

Работу принял преподаватель Мяснов А.А.

Санкт-Петербург

2015

Цели работы

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

Программа работы

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

Выполнение работы

ЗАДАЧА 1: Рассчитать отношение (в %) количества призывников к общему количеству новых призывников по годам и изменение этого показателя по отношению к предыдущему году.

Для начала создадим запрос для вывода по годам количества новых призывников :

```
select EXTRACT(YEAR FROM p1.registr_date), count(p1.id_file) from personal_file as p1

 where p1.registr_date > '01.01.2000'  group by EXTRACT(YEAR FROM p1.registr_date)
```

EXTRACT	COUNT
2 000	506
2 001	5 031
2 002	4 947
2 003	4 942
2 004	4 961
2 005	4 967
2 006	5 032
2 007	4 925
2 008	4 937
2 009	5 014
2 010	4 918
2 011	4 947
2 012	4 982
2 013	5 034
2 014	4 941

Видим статистику по каждому году. Напишем хранимую процедуру, которая будет обрабатывать эти данные и подсчитывать отношение (в процентах) показателя новых призывников по отношению к предыдущему году:

```
create or alter procedure PROCENT_ON_YEAR
returns ( param1 integer,
          percent float)
as
declare variable value_1 float;
declare variable value_2 float;
declare variable value_3 float;
declare variable value_4 float;
BEGIN
  FOR select EXTRACT(YEAR FROM p1.registr_date), count(p1.id_file) from personal_file as p1
     where p1.registr_date > '01.01.2000'  group by EXTRACT(YEAR FROM p1.registr_date)
     INTO :param1, :value_1
  DO
  BEGIN

    if (:param1 > 2000) then
    begin
       value_2 = value_1;
       value_4 = (value_2 / value_3)*100;
       if (value_4 > 100) then
       begin
         percent = value_4 - 100;
       end
       value_4 = (value_2 / value_3)*100;
       if (value_4 > 100) then
       begin
         percent = value_4 - 100;
       end
       else
       begin
         percent = (100 - value_4)*(-1) ;
       end
       value_3 = value_2;
       suspend;
    end
    else
    begin
        value_3 = value_1;
    end
  END
END
```

Запустим процедуру и оценим результат:

NUM_YEAR	PERCENT
2 001	894,269
2 002	-1,670
2 003	-0,101
2 004	0,384
2 005	0,121
2 006	1,309
2 007	-2,126
2 008	0,244
2 009	1,560
2 010	-1,915
2 011	0,590
2 012	0,707
2 013	1,044
2 014	-1,847
2 015	2,388
2 016	-94,030

Убедимся, что выходные данные верны. Проверим статистику увеличения процента с 2003 на 2004 год. Persent = 100% - (4961 / 4942)*100% = 0.384%. Получается, что набор призывников увеличился на 0.384% по сравнению с прошлым годом. Это свидетельствует о правильности работы процедуры.

Execute time = 18ms

ЗАДАЧА 2: За выбранный период времени вывести отношение (в %) военнослужащих заключивших контракты после прохождения службы по призыву.

Чтобы удостовериться в правильности работы процедуры введем в нашу базу некоторые оговоренности:

- В таблице KIND_ACTIVITY вид деятельности с **id = 499** – служба по контракту

- В таблице KIND_ACTIVITY вид деятельности с **id = 498** – срочная служба

Создадим процедуру по запросу, представленному ниже хранимую процедуру.

Запрос выводит количество военнослужащих которые уже прошли срочную службу за весь период времени и количество людей, которые после срочной службы, заключившие контракт в заданный период времени (01.01.2010 --- 01.01.2017)

```
select count(p1.id_file),
(
    select count(p1.id_file) from activities as p1
where p1.activity_date > '01.01.2010' and p1.activity_date < '01.01.2017'
and p1.id_activity = 499
) from activities as p1 where  p1.id_activity = 498;
```

Заменим явнозаданные данные входными параметрами. Процедура представлена ниже:

```
create or alter procedure KONTRAKTNIKI(
            DATE_FROM date,
            DATE_TO date)
returns (
    procent float)
as
declare variable kontract float;
declare variable srochnye float;
begin
    for
  select count(p1.id_file),
(
    select count(p1.id_file) from activities as p1
where p1.activity_date > :date_from and p1.activity_date < :date_to
and p1.id_activity = 499
) from activities as p1  where p1.id_activity = 498
    into :srochnye, :kontract
    do
    begin
    procent = (kontract / srochnye)*100;
    end

end
```

Для того чтобы хоть примерно знать порядок цифр людей, прошедших срочную службу (id=498), и людей, которые за весь период времени заключили контракты (id=499), напишем запрос. Его результат представлен ниже:

498	14 035
499	5 026

Проведем тестирование данной процедуры:

1) Зададим период с 01.01.2007 до 01.01.2015

   Результат работы: PROCENT = 19,0523681640625

2) период с 01.01.2009 до 01.01.2012

   Результат работы: PROCENT = 7,74492359161377

   Execute time = 16ms

Выводы:
В данной работе мы познакомились с хранимыми процедурами. Этот инструмент очень похож на подпрограммы на языках программирования: есть входные параметры, выходные и какое-либо вычисление. Хранимые процедуры являются объектами БД, чтобы улучшить производительность и постоянство выполнения повторяемых задач. Создать хранимую процедуру в IBExpert можно двумя способами: 1) написанный запрос в новой вкладке sql-editor можно автоматически перевести в хранимую процедуру; 2) начинать создавать ее с нуля. IBExpert помогает пользователю, редактор процедуры может выполняться в “lazy mode” (когда вспомогательные данные вводятся в вкладках “Входные параметры”, “Выходные параметры” и т.п.

Проведем некоторое резюме по поводу использования хранимых процедур:
Преимущества:
1) ХП могут уменьшить сетевой трафик. Вместо того, чтобы посылать сотни операторов по сети, пользователи могут выполнить сложную операцию, посылая единственный оператор, который сокращает количество запросов, проходящих между клиентом и сервером.
2) Обеспечивает механизм защиты. Если клиент может получить доступ к данным только через хранимые процедуры, никто не сможет стереть данные через команду DELETE
3) Хранимые процедуры выполняются намного быстрее относительно запросов над базой данных из приложения.

Недостатки:

1) Повышение нагрузки на сервер баз данных в связи с тем, что большая часть работы выполняется на серверной части, а меньшая - на клиентской.
2) По отзывам программистов БД сложность вызывает миграция с одной СУБД на другую, что может привести к проблемам.
3) Дополнительные «телодвижения» для изучения синтаксиса