Санкт-Петербургский государственный политехнический университет

Институт информационных технологий и управления

Кафедра компьютерных систем и программных технологий

Отчёт

по лабораторной работе № 4

“Хранимые процедуры SQL”

по дисциплине «Базы данных»

Работу выполнил студент гр.43501/13

Михайлов Д.М. \_\_\_\_\_

Преподаватель:

Моисеев М.Ю. \_\_\_\_\_

Санкт-Петербург

2013

1. **Цель работы**

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

1. **Программа работы**

Познакомиться с возможностями реализации обработки данных на стороне сервера БД с помощью триггеров и хранимых процедур.

1. Создать процедуру, которая добавляет тому клиенту, который написал больше всего ревью, бесплатный тур в тот курорт, в котором клиент ещё не был. Продемонстрировать её корректную работу.
2. Создать процедуру, которая выводит всех клиентов, квадрат числа билетов которых меньше куба числа отелей в которых они были. Продемонстрировать её корректную работу.
3. **Выполнение программы работы**

**3.1. Процедура реализует, помимо своей основной задачи, вывод данных для отображения правильности её выполнения. Вначале процедура ищет клиента с большим количеством ревью (или выше стоящего по id в случае одинаковых значений) и выводит его id, а также запоминает значение id в переменной. Затем она выводит все курорты, в которых был найденный клиент. Далее процедура во вложенном цикле выполняет поиск курорта, в котором клиент ещё не был, в случае нахождения добавляет путевку в этот курорт и выводит его id.**

/\*Тому, кто написал больше всего ревью, бесплатный тур в тот курорт, \*/

/\* в котором он ещё не был\*/

connect 'C:\Users\Danya\Documents\Database\tour\_named\tour\_named.fdb' user 'SYSDBA' password 'masterkey';

-- delete from CARD where card\_id = :max\_card\_id;

-- delete from BOOKING where card\_booking\_id = :max\_card\_id;

-- drop generator maxcard;

CREATE GENERATOR maxcard;

create or alter procedure Maxrev

returns (cl integer, num integer, cl\_resorts integer, free\_resort integer)

as

declare variable cl\_id integer; /\* client id \*/

declare variable cl\_num integer; /\* number of clients reviewed \*/

declare variable cl\_resort\_id integer; /\* client's resorts \*/

declare variable resort\_id integer; /\* resort id , one not repeating resort for client \*/

declare variable insert\_tour integer;

declare variable insert\_ri integer;

declare variable resort\_found integer;

declare variable max\_card\_id integer;

declare variable last\_resort integer;

begin

cl\_id = 0;

cl\_num = 0;

cl\_resort\_id = 0;

resort\_id = 0;

resort\_counter = 0;

insert\_tour = 0;

insert\_ri = 0;

max\_card\_id = 0;

resort\_found = 0;

-------------------------- finding client with max review and showing him

for select first 1 client.p\_id, client.name, client.surname, count(tour.t\_id) as times

from client, review, tour

where client.p\_id = review.p\_review\_id and review.t\_review\_id = tour.t\_id

group by client.p\_id, client.name, client.surname

order by times desc

into :cl\_id,:cl\_num

do begin

cl = :cl\_id;

num = :cl\_num;

suspend;

end

-------------------------- choosing all cl\_id's resorts and showing them

for select resort.r\_id

from client, card, booking, room\_info, hotel, resort

where p\_id = :cl\_id and p\_id = p\_card\_id -- client is one so cl\_id is the only

and card\_id = card\_booking\_id and ri\_booking\_id = ri\_id

and h\_ri\_id = h\_id and r\_hotel\_id = r\_id

into :cl\_resort\_id

do begin

cl\_resorts = :cl\_resort\_id;

suspend;

end

cl\_resort\_id = 0;

-------------------------- for cl\_id finding not repeating resort and showing it

for select first 200 resort.r\_id, tour.t\_id, room\_info.ri\_id -- resort and its data

from client, card, tour, booking, room\_info, hotel, resort

where t\_id = t\_card\_id and card\_id = card\_booking\_id

and ri\_booking\_id = ri\_id and h\_ri\_id = h\_id

and r\_hotel\_id = r\_id -- and :resort\_found = '0'

into :resort\_id,:insert\_tour,:insert\_ri

do begin

if(resort\_found='0') then begin

resort\_found = 1;

for select resort.r\_id

from client, card, booking, room\_info, hotel, resort

where p\_id = :cl\_id and p\_id = p\_card\_id -- client is one so cl\_id is the only

and card\_id = card\_booking\_id and ri\_booking\_id = ri\_id

and h\_ri\_id = h\_id and r\_hotel\_id = r\_id

into :cl\_resort\_id

do begin

if(:resort\_id=:cl\_resort\_id) then

resort\_found = 0;

end

if(:resort\_found = '1') then begin

max\_card\_id = gen\_id(maxcard,(SELECT MAX(card\_id) FROM card) + 1);

INSERT INTO CARD VALUES (:max\_card\_id,:cl\_id, :insert\_tour, 0); -- we know tour

INSERT INTO BOOKING VALUES (:max\_card\_id, :insert\_ri ,'11-DEC-2013', 1);

free\_resort = :resort\_id;

suspend;

end

end

end

end

Выполнение хранимой процедуры select \* from maxrev

**Для тестирования функциональности используем саму процедуру:**

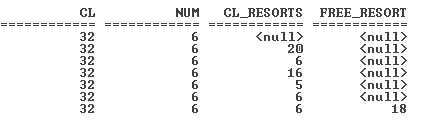


Рис.1

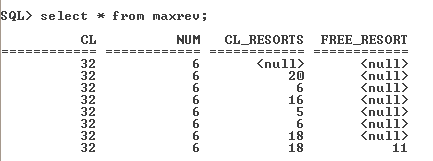


Рис.2

На Рис 1 и 2 показаны результаты работы хранимой процедуры при выполнении вышеописанных операций. Первая строка выводит id клиента и число отзывов, далее выводятся 5 и более id курортов, далее – курорт, относящийся к добавляемой путевке. Не обновленные данные выводятся повторно, но таким образом можно избежать повторения тех же операций в запросах, требуемых для вывода этих же данных.

При достижении максимального количества курортов у клиента путевок с новыми курортами не добавляется.

Для уникальности значения id для пользователя, выполняющего процедуру, а также для обеспечения закрытия доступа другим пользователям к записи с уникальным значением id во время добавления данных, присущих только новой вводимой путевке, используется sql генератор.

**3.2. Вывести всех клиентов, квадрат числа билетов которых меньше куба числа отелей, в которых они были.**

**Процедура во внешнем цикле выполняет поиск соответствий клиент – число билетов и во вложенном – клиент – число отелей, и при выполнении условия выводит найденные данные.**

**Для обеспечения вычета одинаковых отелей используется оператор distinct в операторе select.**

/\*Все клиенты, квадрат числа билетов которых меньше куба числа отелей в которых они были\*/

connect 'C:\Users\Danya\Documents\Database\tour\_named\tour\_named.fdb' user 'SYSDBA' password 'masterkey';

create or alter procedure tickets\_hotels

returns (cl integer, tickets integer, hotels integer)

as

declare variable t\_cl\_id integer; /\* client id \*/

declare variable tk\_num integer;

declare variable h\_cl\_id integer;

declare variable h\_num integer;

begin

t\_cl\_id = 0;

tk\_num = 0;

h\_cl\_id = 0;

h\_num = 0;

-------------------------- finding client - tickets

for select client.p\_id, count(tk\_id) as num\_tickets

from client, card, tk\_connect, tickets

where client.p\_id = card.card\_id and t\_card\_id = t\_tk\_connect\_id

and tk\_tk\_connect\_id = tk\_id

group by client.p\_id

into :t\_cl\_id,:tk\_num

do begin

-------------------------- finding client - hotels

for select client.p\_id, count(distinct h\_id) as num\_hotels -- non repeating hotels

from client, card, booking, room\_info, hotel

where client.p\_id = :t\_cl\_id and client.p\_id = card.card\_id -- inner cycle

and card\_id = card\_booking\_id and ri\_booking\_id = ri\_id

and h\_ri\_id = h\_id

group by client.p\_id

into :h\_cl\_id,:h\_num

do begin

if(power(:tk\_num,2) < power(:h\_num,3)) then begin

cl = :t\_cl\_id;

tickets = :tk\_num;

hotels = :h\_num;

suspend;

end

end

end

end

**Для тестирования функциональности используем выходные данные процедуры:**

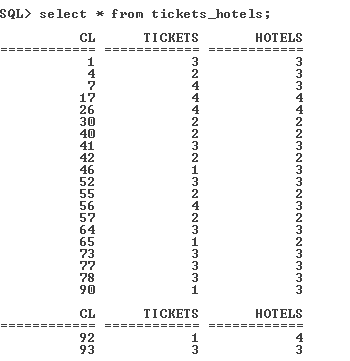


Рис.3

На рис.3 билеты^2<отели^3.

1. **Выводы**

Хранимая процедура — это объект базы данных, представляющий собой набор SQL-инструкций, который компилируется один раз и хранится на сервере.

Рассмотрим достоинства и недостатки хранимых процедур:

Использование хранимых процедур имеет следующие преимущества:

- Удобство использования. Написав единожды хранимую процедуру, больше не потребуется выполнять сложные запросы.

- Скорость исполнения. Так как ХП хранятся на сервере и уже откомпилированы, то их выполнение будет всегда быстрее отправки идентичного по задачам запроса.

- возможность автоматической проверки триггерами DML корректности данных, определенные значения которых необходимо хранить в соответствующей таблице;

- Создание гибкой системы управления данными;

- Используется более развернутое манипулирование данными, включающее сочетание описания действий с неким языком программирования (поддержка локальных переменных, условных операторов, циклов, вызовов процедур).

Недостатком может быть невозможность создания виртуальных таблиц view, что приводит ко вложенным циклам и необходимости перебора большего количества данных.

Также не может быть вызван оператор select для вывода данных в табличном виде в консоль, как это можно сделать в запросах.