Санкт-Петербургский политехнический университет Петра Великого

Институт компьютерных наук и технологий

Кафедра компьютерных систем и программных технологий

**Отчёт о лабораторной работе №5**

**Дисциплина:** Базы данных

**Тема:** Хранимые процедуры

Выполнил студент гр. 43501/1 Калугина М.О.

Руководитель Мяснов А.В.

“ ” 2016 г.

Санкт -Петербург

2016

1. **Цели работы**

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

# Программа работы

* Изучить возможности языка PSQL
* Создать две хранимые процедуры в соответствии с индивидуальным заданием.

1. **Ход работы**

Хранимая процедура (ХП) — это программный модуль, который может быть вызван с клиента, из другой процедуры, функции, выполнимого блока или триггера. Хранимые процедуры, хранимые функции, исполняемые блоки и триггеры пишутся на процедурном языке SQL (PSQL).

Большинство операторов SQL доступно и в PSQL, иногда с ограничениями или расширениями. Заметными исключениями являются DDL и операторы управления транзакциями.

Хранимые процедуры могут принимать и возвращать множество параметров.

1. **Индивидуальное задание**
   1. **Определить клиентов, у которых средний промежуток времени между заказами менее заданного.**

Процедура numerate выдает поле с нумерацией строки (num), с id клиента (cly) и с датой заказа (dat). Также происходит сортировка данных по id клиента по возрастанию, потом по дате заказа по возрастанию.

**SET** TERM **^** **;**

**CREATE** **or** **alter** **PROCEDURE** numerate

**returns** **(**num **int,** cly **int,** dat **date)**

**as**

**declare** **variable** I **int;**

**begin**

I **=** 0**;**

**for** **select** **:**I**,** id\_client**,** date\_ofo

**from** orders

**group** **by** date\_ofo**,** id\_client

**order** **by** id\_client**,** date\_ofo

**into** **:**num**,:**cly**,** **:**dat

**do** **begin**

I **=** I **+** 1**;**

suspend**;**

**end**

**end**

**^**

Процедура find\_interval находит интервал в днях между двумя заказами одного клиента и выводит id клиента (client), интервал (interv), дата предыдущего заказа (interv1) и дату следующего заказа (interv2).

**CREATE** **or** **alter** **PROCEDURE** find\_interval

**returns** **(** client **int,** interv **int,** interv1 **date,** interv2 **date)**

**as**

**begin**

**for** **select** o1**.**cly**,** **(**o2**.**dat **-** o1**.**dat**)** **as** interv**,** o1**.**dat **as** dat1**,** o2**.**dat **as** dat2

**from** numerate **as** o1**,** numerate **as** o2

**where** o2**.**num **=** o1**.**num **+** 1

**and** o1**.**cly **=** o2**.**cly

**order** **by** o1**.**dat**,** o2**.**dat

**into** **:**client**,** **:**interv**,** **:**interv1**,** **:**interv2

**do** **begin**

suspend**;**

**end**

**end**

**^**

Процедура find\_clients ищет и выдает средний интервал между заказами клиента (avg\_interv), который больше заданного.

**CREATE** **or** **alter** **PROCEDURE** find\_clients **(**interv **int)**

**returns** **(** client **int,** avg\_interv **int)**

**as**

**begin**

**for** **select** client**,** avg\_interval **from**

**(**

**select** client**,** **avg(**interv**)** **as** avg\_interval

**from** find\_interval

**group** **by** client

**)**

**where** avg\_interval **>** **:**interv

**into** **:**client**,:**avg\_interv

**do** **begin**

suspend**;**

**end**

**end**

**^**

**SET** TERM **;** **^**

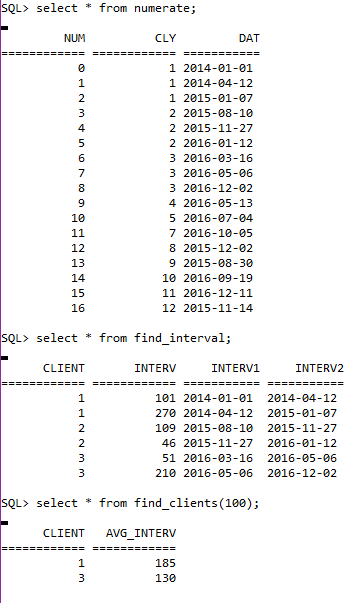


Рис.1. Результат выполнения процедур

* 1. **За заданный промежуток времени рассчитать минимальную, среднюю и максимальную сумму заказа за каждую неделю, месяц и год.**

Процедура stat ищет в промежутке между двумя заданными датами минимальную, среднюю и максимальную сумму заказа за каждую неделю, месяц и год и выдает строку с наименованием недели, месяца или года (str), номер недели, месяца или года (num), минимальная (minimum), средняя (average) и максимальная (maximum) сумма заказа.

**SET** TERM **^** **;**

**CREATE** **or** **alter** **PROCEDURE** stat **(**dat1 **date,** dat2 **date)**

**returns** **(**str **varchar(**5**),** num **smallint,** minimum **int,** average **int,** maximum **int)**

**as**

**declare** **variable** tmpdat **date;**

**begin**

num **=** 0**;**

tmpdat **=** **:**dat1**;**

str **=** 'week'**;**

**while** **(:**tmpdat **<** **:**dat2**)** **do** **begin**

**select** **min(**summ**)** **as** minimum**,** **avg(**summ**)** **as** average**,** **max(**summ**)** **as** maximum

**from** orders

**where** date\_ofo **between** **:**dat1 **and** **:**dat2

**and** **EXTRACT(**week **FROM** date\_ofo**)** **=** **EXTRACT(**week **FROM** **:**tmpdat**)**

**and** **EXTRACT(month** **FROM** date\_ofo**)** **=** **EXTRACT(month** **FROM** **:**tmpdat**)**

**and** **EXTRACT(year** **FROM** date\_ofo**)** **=** **EXTRACT(year** **FROM** **:**tmpdat**)**

**into** **:**minimum**,:**average**,:**maximum**;**

num **=** **:**num **+** 1**;**

tmpdat **=** **:**tmpdat **+** 7**;**

suspend**;**

**end**

num **=** 0**;**

tmpdat **=** **:**dat1**;**

str **=** 'month'**;**

**while** **(:**tmpdat **<** **:**dat2**)** **do** **begin**

**select** **min(**summ**)** **as** minimum**,** **avg(**summ**)** **as** average**,** **max(**summ**)** **as** maximum

**from** orders

**where** date\_ofo **between** **:**dat1 **and** **:**dat2

**and** **EXTRACT(month** **FROM** date\_ofo**)** **=** **EXTRACT(month** **FROM** **:**tmpdat**)**

**and** **EXTRACT(year** **FROM** date\_ofo**)** **=** **EXTRACT(year** **FROM** **:**tmpdat**)**

**into** **:**minimum**,:**average**,:**maximum**;**

num **=** **:**num **+** 1**;**

tmpdat **=** **:**tmpdat **+** 30**;**

suspend**;**

**end**

num **=** 0**;**

tmpdat **=** **:**dat1**;**

str **=** 'year'**;**

**while** **(:**tmpdat **<** **:**dat2**)** **do** **begin**

**select** **min(**summ**)** **as** minimum**,** **avg(**summ**)** **as** average**,** **max(**summ**)** **as** maximum

**from** orders

**where** date\_ofo **between** **:**dat1 **and** **:**dat2

**and** **EXTRACT(year** **FROM** date\_ofo**)** **=** **EXTRACT(year** **FROM** **:**tmpdat**)**

**into** **:**minimum**,:**average**,:**maximum**;**

num **=** **:**num **+** 1**;**

tmpdat **=** **:**tmpdat **+** 365**;**

suspend**;**

**end**

**end**

**^**

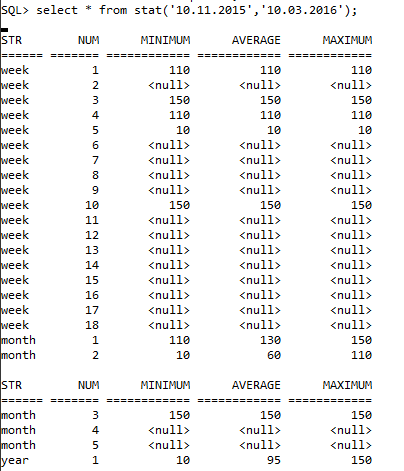


Рис.2. Результат выполнения скрипта

1. **Вывод**

С помощью хранимых процедур мы можем вызывать фрагмент кода в любой момент, что обеспечивает нам модульность. Становится проще использовать код повторно. С помощью параметров, передаваемые в процедуры, мы можем данные, полученные в одной процедуре, использовать в другой.

Так как часто используемые операции выполняются на сервере, это существенно уменьшает количество данных, которые передаются по сети данных.

Недостаток такой базы данных это сложный перенос проекта из одной СУБД в другую, так как хранимые процедуры зависят от типа и версии СУБД.