Санкт-Петербургский политехнический университет Петра Великого

Институт компьютерных наук и технологий

Кафедра компьютерных систем и программных технологий

**Дисциплина**

Базы данных

**Отчёт по лабораторной работе**

# SQL-программирование: хранимые процедуры

Работу выполнил студент группы № 43501/1 Трофимова Н.С.

Работу принял преподаватель \_\_\_\_\_\_\_\_\_\_\_\_\_\_ Мяснов А.В.

Санкт-Петербург

2016

Хранимая процедура является программой, хранящейся в области метаданных базы данных и выполняющейся на стороне сервера. К хранимой процедуре могут обращаться хранимые процедуры (в том числе и сама к себе), триггеры и клиентские программы. Если хранимая процедура вызывает саму себя, то такая хранимая процедура называется рекурсивной.

Существуют два вида хранимых процедур — выполняемые хранимые процедуры (executable stored procedures) и селективные процедуры (selectable stored procedures).

Выполняемые хранимые процедуры, осуществляют обработку данных, находящихся в базе данных. Эти процедуры могут получать входные параметры и возвращать одиночный набор выходных (RETURNS) параметров. Такие процедуры выполняются с помощью оператора EXECUTE PROCEDURE.

Селективные хранимые процедуры обычно осуществляют выборку данных из базы данных и возвращают при этом произвольное количество строк. Такие процедуры позволяют получать довольно сложные наборы данных, которые зачастую

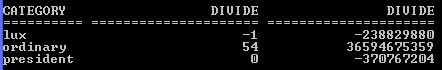
невозможно или весьма затруднительно получить с помощью обычных DSQL SELECT

запросов. Обычно такие процедуры выполняют циклический процесс извлечения данных,

возможно преобразуя их, прежде чем заполнить выходные переменные (параметры) новыми данными на каждой итерации цикла. Оператор SUSPEND, обычно расположенный в конце каждой итерации, заполняет буфер и ожидает пока вызывающая сторона не выберет (fetch) строку.

1. По каждому типу комнат вывести статистику загрузки за год (отношение занятых дней клиентами к общему числу) и приходов денег от клиентов поквартально.

select rooms.category, sum(contracts.lastday-contracts.firstday)/365, sum(contracts.paidcost)/4 from rooms,contracts where (rooms.roomid=contracts.roomid and contracts.firstday>'01.01.2016' and contracts.lastday<'31.12.2016') group by rooms.category;



create or alter procedure CLIENTSTAT (

FDY date,

LDY date)

returns (

CATID varchar(10),

BUSYDAYS float,

MIDDLECOST float)

as

begin

select rooms.category, sum(datediff(day,contracts.firstday+contracts.firsttime,contracts.lastday+contracts.lasttime))/365, sum(contracts.paidcost)/4 from rooms,contracts where contracts.firstday>:fdy and contracts.lastday<:ldy group by category into :catid, :busydays, :middlecost;

end^

SET TERM ; ^

/\* Following GRANT statements are generated automatically \*/

GRANT SELECT ON ROOMS TO PROCEDURE CLIENTSTAT;

GRANT SELECT ON CONTRACTS TO PROCEDURE CLIENTSTAT;

/\* Existing privileges on this procedure \*/

GRANT EXECUTE ON PROCEDURE CLIENTSTAT TO SYSDBA;

1. Для заданного размера интервала (в днях) найти наиболее загруженный и наименее загруженный период в году.

Получение количества действующих контрактов по датам:

select count(contractid) from contracts where firstday between '01.01.2016' and '31.12.2016' and lastday between '01.01.2016' and '31.12.2016' group by firstday;

Процедура:

SET TERM ^ ;

create or alter procedure SEEDATES (

LDY date,

FDY date)

returns (

MAXDATE date,

MINDATE date

MININT integer,

MAXINT integer,)

as

begin

select first 1 firstday, (firstday-:fdy)/15 from contracts where firstday between :fdy and :ldy and lastday between :fdy and :ldy group by firstday order by count(contractid) into :mindate,:minint;

select first 1 firstday, (firstday-:fdy)/15 from contracts where firstday between :fdy and :ldy and lastday between :fdy and :ldy group by firstday order by count(contractid) desc into :maxdate,:maxint; suspend;

end^

SET TERM ; ^

**Выводы:**

Я получила опыт написания хранимых процедур, меняющих данные в таблице, производящих поиск записей и вычисления.

Хранимые процедуры имеют следующие преимущества:

1. *Модульность:* Приложения, работающие с одной и той же базой данных, могут

использовать одну и ту же хранимую процедуру, тем самым уменьшив размер кода

приложения и устранив дублирование кода.

2. *Упрощение поддержки приложений:* При изменении хранимой процедуры, изменения

отражаются сразу во всех приложениях, без необходимости их перекомпиляции.

3. *Увеличение производительности:* Поскольку хранимые процедуры выполняются на

стороне сервера, а не клиента, то это уменьшает сетевой трафик, что повышает

производительность.