

Санкт-Петербургский государственный политехнический  
университет Петра Великого  
**Кафедра компьютерных систем и программных технологий**

**Отчет по лабораторной работе**

**Дисциплина:** Базы данных

**Тема:** SQL-программирование: хранимые процедуры

Выполнил  
студент гр. 43501/3

\_\_\_\_\_ Д. А. Зобков  
(подпись)

Преподаватель

\_\_\_\_\_ А. В. Мяснов  
(подпись)

“\_\_” \_\_\_\_\_ 2016 г.

Санкт-Петербург  
2016 г.

## 1 Цель работы

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

## 2 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 системе контроля версий;
4. Продемонстрировать результаты преподавателю.

## 3 Ход работы

- 3.1 Реализовать процедуру выдачи награды за самый популярный по продажам альбом: за заданный период вычислять самый продаваемый альбом и выдавать ему награду заданного типа

```
1 CREATE OR ALTER PROCEDURE sp1 (  
2     START_DATE DATE,  
3     FINISH_DATE DATE,  
4     AWARD_ID INTEGER  
5 )  
6 AS  
7 DECLARE VARIABLE TOP_ALBUM INTEGER;  
8 DECLARE VARIABLE ID INTEGER;  
9 BEGIN  
10     /* Находим самый продаваемый альбом */  
11     SELECT FIRST 1 ALB_IN_CD.ALB_ID AS ALB FROM SELLING  
12         JOIN ALB_IN_CD ON SELLING.CD_ID = ALB_IN_CD.CD_ID  
13         WHERE SOLD_DATE BETWEEN :START_DATE AND :FINISH_DATE  
14         GROUP BY ALB  
15         ORDER BY SUM(SOLD) DESC  
16         INTO :TOP_ALBUM;  
17     /* Вычисляем количество записей в таблице */  
18     SELECT COUNT(*) FROM (SELECT * FROM ALB_AWARDS) INTO :ID;  
19     /* Инкрементируем первичный ключ */  
20     ID = ID + 1;  
21     /* Добавляем запись */  
22     INSERT INTO ALB_AWARDS VALUES (:ID, :TOP_ALBUM, :AWARD_ID);  
23 END  
24 -- Завершение транзакции  
25 COMMIT;
```

### 3.2 Реализовать процедуру вычисления агрегатов — сумму продаж по дням, неделям, месяцам, годам, городам, странам. Для хранения вычисленных значений необходимо создать отдельную таблицу.

Для хранения вычисленных значений создадим по таблице для каждого агрегата:

```
1 CREATE TABLE SOLD_DAY_SUMMARY (
2     SUMMARY_DATE DATE NOT NULL,
3     SOLD_SUM     INTEGER NOT NULL
4 );
5
6 CREATE TABLE SOLD_WEEK_SUMMARY (
7     START_DATE DATE NOT NULL,
8     FINISH_DATE DATE NOT NULL,
9     SOLD_SUM   INTEGER NOT NULL
10 );
11
12 CREATE TABLE SOLD_MONTH_SUMMARY (
13     START_DATE DATE NOT NULL,
14     FINISH_DATE DATE NOT NULL,
15     SOLD_SUM   INTEGER NOT NULL
16 );
17
18 CREATE TABLE SOLD_YEAR_SUMMARY (
19     START_DATE DATE NOT NULL,
20     FINISH_DATE DATE NOT NULL,
21     SOLD_SUM   INTEGER NOT NULL
22 );
23
24 CREATE TABLE SOLD_CITY_SUMMARY (
25     CITY_ID INTEGER NOT NULL,
26     SOLD_SUM INTEGER NOT NULL
27 );
28
29 CREATE TABLE SOLD_COUNTRY_SUMMARY (
30     COUNTRY_ID INTEGER NOT NULL,
31     SOLD_SUM   INTEGER NOT NULL
32 );
33
34 ALTER TABLE SOLD_CITY_SUMMARY
35     ADD FOREIGN KEY (CITY_ID) REFERENCES CITY_LIST (CITY_ID);
36 ALTER TABLE SOLD_COUNTRY_SUMMARY
37     ADD FOREIGN KEY (COUNTRY_ID) REFERENCES COUNTRY_LIST (COUNTRY_ID);
38 -- Завершение транзакции
39 COMMIT;
```

Реализованная процедура:

```
1 /* Основная процедура */
2 CREATE OR ALTER PROCEDURE sp2 (
3     OPERATION VARCHAR(7),          -- Вид диапазона
4     SUMMARY_DATE DATE              -- Дата для пересчета
5 ) AS
```

```

6  DECLARE VARIABLE START_DATE DATE;      -- Дата начала диапазона
  DECLARE VARIABLE FINISH_DATE DATE;      -- Дата конца диапазона
8  DECLARE VARIABLE DADD DATE;            -- Результат DATEADD()
  DECLARE VARIABLE EXT INT;               -- Результат EXTRACT()
10 DECLARE VARIABLE ID INT;               -- ID городов/стран
  DECLARE VARIABLE SOLD_COUNT INT;        -- Количество проданных дисков за
    диапазон
12 DECLARE VARIABLE QUERY VARCHAR(100); -- Строка с запросом
BEGIN
14  IF (:OPERATION = 'DAY') THEN
    BEGIN
16      /* Находим количество продаж */
      SELECT SUM(SOLD) FROM SELLING
18      WHERE SOLD_DATE=:SUMMARY_DATE
        INTO :SOLD_COUNT;

20      /* Добавляем новую запись либо обновляем старую */
22      UPDATE OR INSERT INTO SOLD_DAY_SUMMARY
        VALUES (:SUMMARY_DATE, :SOLD_COUNT)
24      MATCHING (SUMMARY_DATE);
    END

26  ELSE IF (:OPERATION = 'WEEK' OR :OPERATION = 'MONTH' OR :OPERATION =
    'YEAR') THEN
28  BEGIN
    /* Находим начало и конец диапазона */
30    IF (:OPERATION = 'WEEK') THEN BEGIN
      EXT = EXTRACT(WEEKDAY FROM :SUMMARY_DATE);
32      IF (EXT = 0) THEN EXT = 7; -- Воскресенье

      START_DATE = :SUMMARY_DATE+1-EXT;
      FINISH_DATE = :SUMMARY_DATE+7-EXT;
36    END
    ELSE IF (:OPERATION = 'MONTH') THEN BEGIN
38      EXT = EXTRACT(DAY FROM :SUMMARY_DATE);
      DADD = DATEADD(MONTH, 1, :SUMMARY_DATE);

40      START_DATE = :SUMMARY_DATE-EXT+1;
      FINISH_DATE = DADD-EXTRACT(DAY FROM DADD);
42    END
    ELSE IF (:OPERATION = 'YEAR') THEN BEGIN
44      EXT = EXTRACT(YEARDAY FROM :SUMMARY_DATE);
      DADD = DATEADD(YEAR, 1, :SUMMARY_DATE);

46      START_DATE = :SUMMARY_DATE-EXT;
      FINISH_DATE = DADD-EXTRACT(YEARDAY FROM DADD)-1;
50    END

52    /* Находим количество продаж */
    SELECT SUM(SOLD) FROM SELLING
54    WHERE SOLD_DATE BETWEEN :START_DATE AND :FINISH_DATE
      INTO :SOLD_COUNT;

56    /* Создаем запрос */

```

```

58     QUERY = 'UPDATE OR INSERT INTO SOLD_' || :OPERATION || '_SUMMARY '
        || 'VALUES (?, ?, ?) '
60     || 'MATCHING (START_DATE, FINISH_DATE)';

62     /* Выполнение динамического запроса */
    EXECUTE STATEMENT (QUERY) (:START_DATE, :FINISH_DATE, :SOLD_COUNT);
64 END

66 ELSE IF (:OPERATION = 'CITY') THEN
    BEGIN
68         /* Находим кол-во продаж (диапазон не учитывается) */
        FOR SELECT CITY_ID, SUM(SOLD)
70            FROM SELLING
            GROUP BY CITY_ID
72            INTO :ID, :SOLD_COUNT
        DO BEGIN
74            UPDATE OR INSERT INTO SOLD_CITY_SUMMARY
                VALUES (:ID, :SOLD_COUNT)
76            MATCHING (CITY_ID);
            END
78        END

80 ELSE IF (:OPERATION = 'COUNTRY') THEN
    BEGIN
82         /* Находим кол-во продаж (диапазон не учитывается) */
        FOR SELECT CITIES_IN_COUNTRY.COUNTRY_ID AS COUNTRY,
84            SUM(SELLING.SOLD) FROM SELLING
            JOIN CITIES_IN_COUNTRY
86            ON CITIES_IN_COUNTRY.CITY_ID = SELLING.CITY_ID
            GROUP BY COUNTRY
88            INTO :ID, :SOLD_COUNT
        DO BEGIN
90            UPDATE OR INSERT INTO SOLD_COUNTRY_SUMMARY
                VALUES (:ID, :SOLD_COUNT)
92            MATCHING (COUNTRY_ID);
            END
94        END
    END

96 -- Завершение транзакции
    COMMIT;

```

## 4 Выводы

В ходе выполнения данной работы было проведено ознакомление с хранимыми процедурами, хранящимися в базе данных и напоминающие функции из языков программирования. Хранимые процедуры позволяют упростить выполнение однотипных операций сложной выборки данных из базы, например, в случае отличия только в константах.

Использование хранимых процедур повышает безопасность, позволяя давать пользователю доступ только к ним и изолировать от самой базы данных, уменьшает количество запросов к серверу, вследствие чего повышается скорость работы.

К недостаткам хранимых процедур можно отнести отсутствие контроля целостности кода при изменении структуры БД, вследствие чего поддержка такой БД более затратна, „размазывание“ логики между клиентским приложением и БД, непереносимость между различными диалектами СУБД.