Міністерство освіти і науки України

Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського»

Факультет інформатики та обчислювальної техніки Кафедра інформатики та програмної інженерії

Звіт

з лабораторної роботи №5 з дисципліни

«Бази даних»

# «Основи програмування з використанням мови SQL. Збережені процедури. Курсори. Створення, програмування та керування тригерами.»

Варіант 19

Виконав студент ІП-13 Нещерет Віталій Олександрович

(шифр, прізвище, ім'я, по батькові)

Перевірив Марченко Олена Іванівна

(прізвище, ім'я, по батькові)

Київ 2022

**Лабораторна робота №5**

**Варіант 19**

**Тема:** Основи програмування з використанням мови SQL. Збережені процедури. Курсори. Створення, програмування та керування тригерами.

**Мета:**

* Вивчити правила побудови ідентифікаторів, правила визначення змінних та типів. Визначити правила роботи з циклами та умовними конструкціями, роботу зі змінними типу Table.
* Вивчити синтаксис та семантику функцій та збережених процедур, способів їх ідентифікації, методів визначення та специфікації параметрів та повертаємих значень, виклик функцій та збережених процедур.
* Застосування команд для створення, зміни та видалення як скалярних, так і табличних функцій, збережених процедур.
* Вивчити призначення та типи курсорів, синтаксис та семантику команд мови SQL для створення курсорів, вибірки даних з курсорів, зміни даних із застосуванням курсорів.
* Вивчити призначення та типи тригерів, умов їх активації, синтаксису та семантики для їх створення, модифікації, перейменування, програмування та видалення.

**Постановка задачі**

При виконанні лабораторної роботи необхідно виконати наступні дії:

1) Збережені процедури:

a. запит для створення тимчасової таблиці через змінну типу TABLE;

b. запит з використанням умовної конструкції IF;

c. запит з використанням циклу WHILE;

d. створення процедури без параметрів;

e. створення процедури з вхідним параметром;

f. створення процедури з вхідним параметром та RETURN;

g. створення процедури оновлення даних в деякій таблиці БД;

h. створення процедури, в котрій робиться вибірка даних.

2) Функції:

a. створити функцію, котра повертає деяке скалярне значення;

b. створити функцію, котра повертає таблицю з динамічним набором

стовпців;

c. створити функцію, котра повертає таблицю заданої структури.

3) Робота з курсорами:

a. створити курсор;

b. відкрити курсор;

c. вибірка даних, робота з курсорами.

4) Робота з тригерами:

a. створити тригер, котрий буде спрацьовувати при видаленні даних;

b. створити тригер, котрий буде спрацьовувати при модифікації даних;

c. створити тригер, котрий буде спрацьовувати при додаванні даних.

**Виконання завдання**

**Створені запити:**

USE exchequer;  
  
DROP PROCEDURE IF EXISTS *create\_table*;  
CREATE PROCEDURE *create\_table*()  
BEGIN  
 DROP TABLE IF EXISTS temp\_table;  
 CREATE TEMPORARY TABLE temp\_table AS  
 SELECT \*  
 FROM `кошторис`  
 LIMIT 15;  
END;  
CALL *create\_table*();  
  
DROP PROCEDURE IF EXISTS *procedure\_with\_if*;  
CREATE PROCEDURE *procedure\_with\_if*()  
BEGIN  
 SELECT `кошторис id`,  
 `ліміт витрат`,  
 *IF*(`ліміт витрат` > 5000000, 'Великий бюджет',  
 'Маленький бюджет')  
 AS 'розмір бюджету'  
 FROM `кошторис`  
 ORDER BY `розмір бюджету`;  
END;  
CALL *procedure\_with\_if*();  
  
DROP PROCEDURE IF EXISTS *procedure\_with\_while*;  
CREATE PROCEDURE *procedure\_with\_while*()  
BEGIN  
 DECLARE counter INT DEFAULT 5;  
  
 WHILE counter > 0 DO  
 SELECT counter;  
 SET counter = counter - 1;  
 END WHILE;  
END;  
CALL *procedure\_with\_while*();  
  
DROP PROCEDURE IF EXISTS *procedure\_with\_while*;  
CREATE PROCEDURE *procedure\_with\_while*()  
BEGIN  
 DECLARE counter INT DEFAULT 5;  
  
 WHILE counter > 0 DO  
 SELECT counter;  
 SET counter = counter - 1;  
 END WHILE;  
END;  
CALL *procedure\_with\_while*();  
  
DROP PROCEDURE IF EXISTS *procedure\_without\_params*;  
CREATE PROCEDURE *procedure\_without\_params*()  
BEGIN  
 SELECT \*  
 FROM кекв;  
END;  
CALL *procedure\_without\_params*();  
  
DROP PROCEDURE IF EXISTS *procedure\_with\_params*;  
CREATE PROCEDURE *procedure\_with\_params*(  
 IN max\_sum INT  
)  
BEGIN  
 SELECT \*  
 FROM кошторис  
 WHERE `витрачена сума` < max\_sum;  
END;  
CALL *procedure\_with\_params*(1000000);  
  
DROP PROCEDURE IF EXISTS *procedure\_with\_in\_and\_out*;  
CREATE PROCEDURE *procedure\_with\_in\_and\_out*(  
 IN max\_sum INT,  
 OUT res INT  
)  
BEGIN  
 SELECT *COUNT*(\*)  
 FROM кошторис  
 WHERE `витрачена сума` < max\_sum  
 INTO res;  
 SELECT res;  
END;  
CALL *procedure\_with\_in\_and\_out*(1000000, @res);  
  
DROP PROCEDURE IF EXISTS *update\_table*;  
CREATE PROCEDURE *update\_table*()  
BEGIN  
 UPDATE `кошторис`  
 SET `витрачена сума` = `витрачена сума` + 100000  
 WHERE `витрачена сума` < 1000000;  
END;  
CALL *update\_table*();  
  
DROP PROCEDURE IF EXISTS *select\_procedure*;  
CREATE PROCEDURE *select\_procedure*()  
BEGIN  
 SELECT \*  
 FROM `кошторис`  
 WHERE `витрачена сума` < 1000000;  
END;  
CALL *select\_procedure*();  
  
DROP FUNCTION IF EXISTS *function\_scalar*;  
CREATE FUNCTION *function\_scalar*() RETURNS INT  
READS SQL DATA  
BEGIN  
 DECLARE res INT;  
 SELECT *COUNT*(\*)  
 FROM `кошторис`  
 WHERE `витрачена сума` < 1000000  
 INTO res;  
 RETURN res;  
END;  
SELECT *function\_scalar*();  
  
-- Function in MySQL cannot return a table. So, I use stored procedure for this task.  
DROP PROCEDURE IF EXISTS *procedure\_table\_with\_dynamic\_columns*;  
CREATE PROCEDURE *procedure\_table\_with\_dynamic\_columns*(  
 IN magic\_number INT  
)  
BEGIN  
 IF magic\_number = 1 THEN  
 SELECT `кекв id`, `витрачена сума`  
 FROM `кекв`;  
 ELSE  
 SELECT \*  
 FROM `кекв`;  
 END IF;  
END;  
CALL *procedure\_table\_with\_dynamic\_columns*(1);  
  
DROP PROCEDURE IF EXISTS *work\_with\_cursor*;  
CREATE PROCEDURE *work\_with\_cursor*(  
)  
BEGIN  
 DECLARE done INT DEFAULT 0;  
 DECLARE name VARCHAR(255);  
 DECLARE cur CURSOR FOR  
 SELECT назва  
 FROM `комерційний банк`;  
 DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = 1;  
 OPEN cur;  
 read\_loop: LOOP  
 FETCH cur INTO name;  
 IF done = 1 THEN  
 LEAVE read\_loop;  
 END IF;  
 SELECT name;  
 END LOOP;  
END;  
CALL *work\_with\_cursor*();  
  
DROP TRIGGER IF EXISTS trigger\_delete;  
CREATE TRIGGER trigger\_delete BEFORE DELETE ON `кошторис` FOR EACH ROW BEGIN  
 IF OLD.`витрачена сума` > 1000000 THEN  
 SIGNAL SQLSTATE '45000'  
 SET MESSAGE\_TEXT = 'Ви не можете видалити запис, в якому витрачена сума більша за 1000000';  
 END IF;  
END;  
  
DROP TRIGGER IF EXISTS trigger\_update;  
CREATE TRIGGER trigger\_update BEFORE UPDATE ON `кошторис` FOR EACH ROW BEGIN  
 IF NEW.`витрачена сума` > 1000000 THEN  
 SET NEW.`витрачена сума` = 1000000;  
 END IF;  
END;  
  
DROP TRIGGER IF EXISTS trigger\_insert;  
CREATE TRIGGER trigger\_insert BEFORE INSERT ON `кошторис` FOR EACH ROW BEGIN  
 IF NEW.`витрачена сума` > 1000000 THEN  
 SET NEW.`витрачена сума` = 1000000;  
 END IF;  
END;

**Висновок:**

На даній лабораторній роботі я попрактикувався у створенні та використанні збережених процедур та функцій на мові SQL. Вивчив можливості створення процедур з вхідними та вихідними параметрами. Також, навчився працювати з курсорами та тригерами.