**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”

**“Побудова простих запитів”**

Варіант 32

ВиконалаІП-21 Скрипець Ольга Олександрівна

ПеревірилаМарченко О. І.

**Київ 2023**

**ЛАБОРАТОРНА РОБОТА №5**

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

**Мета роботи:**

* 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 значень, котрі повертаються, 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 лабораторної роботи № 5**

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.

**Варіант 32:**

**32) “** Тролейбусне депо **“**

Програмне забезпечення «Тролейбусне депо» . База даних містить інформацію: відомості про водіїв (табельний номер; ПІБ ; дата , час початку і закінчення роботи на маршруті), відомості про тролейбуси (номер ; марка ; кількість пасажирських місць; дата останнього технічного огляду), відомості про маршрути (номер ; час початку і кінця роботи тролейбусів ; початковий пункт ; кінцевий пункт ; список зупинок ; тривалість маршруту). Кожен водій працює на одному тролейбусі і на одному маршруті . На одному маршруті працює кілька водіїв. Кількість пасажирських місць залежить тільки від марки тролейбуса.

**Виконання:**

**1) Збережені процедури:**

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

-- створення тимчасової таблиці через процедуру

DELIMITER //

CREATE PROCEDURE CreateTempTable ()

BEGIN

CREATE TABLE temp\_table(

tempID INT PRIMARY KEY AUTO\_INCREMENT,

temp\_line VARCHAR(30)

);

END //

DELIMITER ;

**b. запит з використанням умовної конструкції IF;**

-- якщо у розкладі субота, то буде хороший настрій

DELIMITER //

CREATE PROCEDURE ifProcedure()

BEGIN

SELECT scheduleID, start\_work,

IF(weekday = 'Sunday', 'Good day', 'Lasy day') AS moode

FROM schedule;

END //

DELIMITER ;

**c. запит з використанням циклу WHILE;**

-- процедура додає нових водіїв з прізвищем "Podoliak" та різними ідентифікаторами водія до таблиці

DROP PROCEDURE IF EXISTS whileProcedure;

DELIMITER //

CREATE PROCEDURE whileProcedure()

BEGIN

DECLARE dr\_ID INT;

SET dr\_ID = 21;

WHILE dr\_ID < 25 DO

insert into driver (DriverID, surname) values (dr\_ID, CONCAT('Podoliak', dr\_ID));

SET dr\_ID = dr\_ID+1;

END WHILE;

END //

DELIMITER ;

CALL whileProcedure;

**d. створення процедури без параметрів;**

-- процедура рахує водіїв

DELIMITER //

CREATE PROCEDURE calcDrivers()

BEGIN

SELECT COUNT(DriverID) FROM driver;

END //

DELIMITER ;

CALL calcDrivers;

**e. створення процедури з вхідним параметром;**

-- видає назву зупинки за номером

DELIMITER //

CREATE PROCEDURE GetBusStopNameByNumber(IN stopID INT)

BEGIN

DECLARE stopName VARCHAR(40);

SELECT bus\_stop\_name INTO stopName

FROM bus\_stop

WHERE bus\_stopID = stopID;

SELECT CONCAT('Bus stop name: ', stopName) AS message

END //

DELIMITER ;

CALL GetBusStopNameByNumber(2);

**f. створення процедури з вхідним параметром та RETURN;**

-- рахує кількість зупинок

DELIMITER //

CREATE PROCEDURE CountBusStopWithReturn(OUT count\_stop INT)

BEGIN

SET count\_stop = (SELECT COUNT(bus\_stopID) FROM bus\_stop);

END //

DELIMITER ;

DROP PROCEDURE IF EXISTS CountBusStopWithReturn;

CALL CountBusStopWithReturn(@count\_stop);

SELECT @count\_stop;

**g. створення процедури оновлення даних в деякій таблиці БД;**

-- перейменування вказаної зупинки

DELIMITER //

CREATE PROCEDURE UpdateBusStopName(IN id INT, IN newName VARCHAR(45))

BEGIN

UPDATE bus\_stop SET bus\_stop\_name = newName WHERE bus\_stopID = id;

END //

DELIMITER ;

CALL UpdateBusStopName(1,"Шулявка");

**h. створення процедури, в котрій робиться вибірка даних.**

-- показує шлях тролейбуса

DELIMITER //

CREATE PROCEDURE wayOfTrolleybus(id VARCHAR(5))

BEGIN

SELECT \* FROM route WHERE route\_number IN(

SELECT DISTINCT d.route\_number FROM driver d, trolleybus t WHERE t.troll\_number=d.troll\_number AND t.troll\_number = id);

END //

DELIMITER ;

CALL wayOfTrolleybus('T13');

DROP PROCEDURE IF EXISTS wayOfTrolleybus;

**2) Функції:**

**a. створити функцію, котра повертає деяке скалярне значення;**

-- знаходить загальну кількість пасажирів на маршруті

DELIMITER //

CREATE FUNCTION GetTotalPassengers(routeNumber VARCHAR(20)) RETURNS INT

DETERMINISTIC

BEGIN

DECLARE totalPassengers INT;

SELECT SUM(number\_of\_passengers)

INTO totalPassengers

FROM trolleybus

WHERE troll\_number IN (SELECT troll\_number FROM driver WHERE route\_number = routeNumber);

RETURN totalPassengers;

END //

DELIMITER ;

SELECT GetTotalPassengers('Route13') AS TotalPassengers;

**b. створити функцію, котра повертає таблицю з динамічним набором стовпців;**

DELIMITER //

CREATE PROCEDURE DynamicTable()

BEGIN

SET @sql = CONCAT('SELECT \* FROM ', 'route');

PREPARE stmt FROM @sql;

EXECUTE stmt;

DEALLOCATE PREPARE stmt;

END//

DELIMITER ;

CALL DynamicTable;

**c. створити функцію, котра повертає таблицю заданої структури.**

DROP PROCEDURE IF EXISTS driver\_and\_trolleybus;

DROP TABLE IF EXISTS driver\_trolleybus;

DELIMITER //

CREATE PROCEDURE driver\_and\_trolleybus()

BEGIN

CREATE TEMPORARY TABLE driver\_trolleybus (full\_name VARCHAR(90), brand VARCHAR(45));

INSERT INTO driver\_trolleybus SELECT CONCAT(d.first\_name,' ',d.surname,' ',d.last\_name) AS driver, c.car\_brand\_name

FROM car\_brand c, driver d, trolleybus t

WHERE c.car\_brandID = t.car\_brandID;

END //

DELIMITER ;

CALL driver\_and\_trolleybus();

SELECT \* FROM driver\_trolleybus;

**3) Робота з курсорами:**

**a. створити курсор;**

DELIMITER //

CREATE PROCEDURE CreateDriverCursor()

BEGIN

DECLARE done INT DEFAULT FALSE;

DECLARE driverName VARCHAR(150);

DECLARE carBrandName VARCHAR(30);

DECLARE driverCursor CURSOR FOR

SELECT CONCAT(d.first\_name, ' ', d.last\_name, ' ', d.surname) AS driver, c.car\_brand\_name AS carBrand

FROM car\_brand c, driver d, trolleybus t

WHERE c.car\_brandID = t.car\_brandID;

DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = TRUE;

END //

DELIMITER ;

CALL CreateDriverCursor;

**b. відкрити курсор;**

DROP PROCEDURE IF EXISTS OpenDriverCursor;

DELIMITER //

CREATE PROCEDURE OpenDriverCursor()

BEGIN

DECLARE done INT DEFAULT FALSE;

DECLARE driverName VARCHAR(150);

DECLARE carBrandName VARCHAR(30);

DECLARE driverCursor CURSOR FOR

SELECT CONCAT(d.first\_name, ' ', d.last\_name, ' ', d.surname) AS driver, c.car\_brand\_name AS carBrand

FROM car\_brand c, driver d, trolleybus t

WHERE c.car\_brandID = t.car\_brandID;

DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = TRUE;

OPEN driverCursor;

CLOSE driverCursor;

END //

DELIMITER ;

CALL OpenDriverCursor;

**c. вибірка даних, робота з курсорами.**

DROP PROCEDURE IF EXISTS SelectDataFromCursor;

DELIMITER //

CREATE PROCEDURE SelectDataFromCursor()

BEGIN

DECLARE done INT DEFAULT FALSE;

DECLARE driverName VARCHAR(150);

DECLARE carBrandName VARCHAR(30);

DECLARE driverCursor CURSOR FOR

SELECT CONCAT(d.first\_name, ' ', d.last\_name, ' ', d.surname) AS driver, c.car\_brand\_name AS carBrand

FROM car\_brand c, driver d, trolleybus t

WHERE c.car\_brandID = t.car\_brandID;

DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = TRUE;

OPEN driverCursor;

FETCH driverCursor INTO driverName, carBrandName;

CLOSE driverCursor;

END //

DELIMITER ;

CALL SelectDataFromCursor;

**4) Робота з тригерами:**

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

DELIMITER //

CREATE TRIGGER driver\_after\_delete

AFTER DELETE ON driver

FOR EACH ROW

BEGIN

INSERT INTO audit\_log(DriverID, deleted\_at)

VALUES (OLD.DriverID, NOW());

END; //

DELIMITER ;

DELETE FROM driver WHERE DriverID = 24;

CREATE TABLE audit\_log(

id INT PRIMARY KEY AUTO\_INCREMENT,

DriverID INT,

deleted\_at DATETIME

);

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

DELIMITER //

CREATE TRIGGER driver\_after\_update

AFTER UPDATE ON driver

FOR EACH ROW

BEGIN

INSERT INTO audit\_log(DriverID, updated\_at, action)

VALUES (NEW.DriverID, NOW(), 'UPDATE');

END; //

DELIMITER ;

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

DELIMITER //

CREATE TRIGGER driver\_after\_insert

AFTER INSERT ON driver

FOR EACH ROW

BEGIN

INSERT INTO audit\_log(DriverID, inserted\_at, action)

VALUES (NEW.DriverID, NOW(), 'INSERT');

END; //

DELIMITER ;

**ВИСНОВОК**

В ході виконання лабораторної роботи було вивчено основи програмування в мові SQL, зокрема збережені процедури, функції, курсори та тригери. Мета роботи включала розгляд побудови ідентифікаторів, змінних та типів даних, а також вивчення синтаксису та семантики команд для створення, модифікації та видалення збережених об'єктів.

У роботі було створено та виконано запити для створення тимчасової таблиці через змінну типу TABLE, використання умовної конструкції IF та циклу WHILE в збережених процедурах. Також було реалізовано створення процедур без та з параметрами, включаючи процедури з використанням RETURN для повернення значень. В контексті функцій були створені скалярні та табличні функції, а також функція з заданою структурою таблиці.

Робота з курсорами включала створення курсорів, їх відкриття та вибірку даних з використанням цих курсорів. У контексті тригерів було створено тригери, які реагують на події видалення, модифікації та додавання даних.

Отже, лабораторна робота дозволила ознайомитися з ключовими концепціями програмування в мові SQL та набути практичних навичок роботи зі збереженими процедурами, функціями, курсорами та тригерами в контексті конкретної бази даних.