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 7

Виконав(ла) ІП-15 *Дацьо Іван*
(шифр, прізвище, ім'я, по батькові)

Перевірив *Ліщук Олександр Васильович*
(прізвище, ім'я, по батькові)

Київ 2022

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 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:

Постановка задачі лабораторної робота № 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.

Індивідуальне завдання, Варіант 7:

Програмне забезпечення «Школа». Загальноосвітня школа, в якій навчаються учні, має номер, назву, адресу, ПІБ директора. У школах є певна кількість класів, котрі мають назву, класного керівника, список учнів, певний перелік предметів. Предмети викладаються вчителями, причому один вчитель може викладати декілька предметів, а однакові предмети можуть викладати різні вчителі. Предмети викладаються згідно з розкладом у кабінетах, котрі мають номер, назву, відповідне обладнання та розкладом класів.

Предмети мають назву, кількість годин вивчення, список навчальних посібників.

Виконання роботи:

1. Збережені процедури:

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

```
CREATE TEMPORARY TABLE ##school_close (
principal_id INT,
name VARCHAR(255),
distnance INT
);
```

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

```
BEGIN
 FOR rec IN SELECT * FROM school
 LOOP
  IF rec.schoolId IN (SELECT classIs FROM Classes) then
   raise notice '% HAS a class!', rec.name;
  ELSE
      raise notice '% DOESNT HAVE a class!', rec.name;
  END IF;
 END LOOP;
```

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

```
DECLARE @counter INT = 0
WHILE @counter < 5
BEGIN
  UPDATE school SET name = 'New one' WHERE distance
= '5'
```

```
  SET @counter = @counter + 1
END
```

Пункт d. **створення процедури без параметрів;**

```
CREATE PROCEDURE update school_close
AS
BEGIN
  UPDATE school_close SET  name = 'New new one' WHERE distance = '100'
END
```

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

```
CREATE    PROCEDURE    update_school_close
(@name_of VARCHAR(255))
AS
BEGIN
   UPDATE  school_close  SET  name=  @name
WHERE distance = '20'
END
```

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

```
CREATE  PROCEDURE  update_school_close  (@name_of
VARCHAR(255), @updated INT OUTPUT)
AS
BEGIN
   UPDATE  school_close  SET  name  =  @name_of  WHERE
distance = '5'
   SET @updated = @@ROWCOUNT
   RETURN @updated
END
```

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

```
CREATE PROCEDURE Task1
@SchoolNumber INT
```

```
AS
IF @SchoolNumber >= (SELECT AVG(School.Number) FROM School)
BEGIN
return @SchoolNumber;
END
ELSE
BEGIN
RETURN 666 + 1;
END
EXEC Task1 15
```

2. Функції:

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

```
CREATE PROCEDURE update_school_close_list (@sc
VARCHAR(20), @updated INT OUTPUT)
AS
BEGIN
   UPDATE orders SET order_status = @status WHERE
order_status = 'pending'
   SET @updated = @@ROWCOUNT
   RETURN @updated
END
```

Пункт b. **створити функцію, котра повертає таблицю з динамічним набором стовпців;** ***Пункт c.*** створити функцію, котра повертає таблицю заданої структури

```
CREATE FUNCTION B(@ClassId int)
RETURNS TABLE
AS
```

RETURN SELECT SchoolClass.Name, SchoolClass.SchoolId

FROM dbo.SchoolClass

WHERE Id = @ClassId

3. Робота з курсорами::

Пункт a. **створити курсор;**

Пункт b. **відкрити курсор;**

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

```
DECLARE school_cursor CURSOR FOR SELECT * FROM School;
OPEN school_cursor;
DECLARE @school_id INT, @school_name VARCHAR(255);


OPEN employee_cursor;


FETCH NEXT FROM school_cursor INTO @school_id, @school
_name;


WHILE @@FETCH_STATUS = 0
BEGIN
    PRINT @school_id + ' - ' + @school_name;
    FETCH NEXT
```

4. Робота з тригерами:

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

```
CREATE TRIGGER b_4
ON SchoolClass
FOR DELETE
```

```
AS
BEGIN
  insert into HISTORY(Message)
  SELECT 'DELETE' + deleted.Name + 'WITH NUMBER' +
deleted.Id
  FROM deleted
END
```

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

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

```
CREATE TRIGGER TrInsertSchool
on School
FOR INSERT
AS
BEGIN
  Declare @id int
  SELECT @Id = Id from inserted
  IF (SELECT SchoolClass.Name FROM SchoolClass WHERE
SchoolClass.SchoolId = @id) LIKE 'N%'
      ROLLBACK;
END
```