Министерство образования Республики Беларусь

Учреждение образования

«Брестский Государственный технический университет»

Кафедра ИИТ

**Лабораторная работа №4**

По дисциплине «Проектирование баз знаний»

Тема: «Хранимые процедуры, функции и триггеры в MS SQL Server 2005/2008.»

**Выполнил:**

Студент 3 курса

Группы ИИ-21

Кирилович А. А.

**Проверил:**

Савонюк В.А.

Брест 2023

**Цель работы:** Изучение и решение заданий по реализации хранимых процедур, функций и триггеровразных видов и фильтров в базах данных.

**Ход работы:**

**Вариант 6**

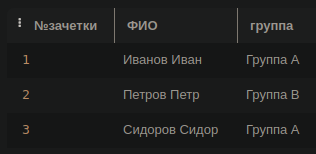
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Вариант 6 | **-** | Студенты |  |  |
| 1 | **№зачетки** | ФИО | группа |  |
| 2 | **Группа** | факультет | специальность |  |
| 3 | **ФИО** | год рождения | адрес | телефон |

Задание.

1. Изучить материал, приведенный в “К лабораторной работе 4.doc”.

2. На основании логической модели (в соответствии с вариантом Лабораторной работы №1) создать и проверить на работоспособность хранимые процедуры, функции и триггеры для каждой из таблиц БД: реализовать хранимые процедуры, функции и триггеры разных видов для каждой таблицы .

**Таблица Зачетки**

CREATE TABLE Зачетки (

№зачетки INTEGER PRIMARY KEY,

ФИО TEXT,

группа TEXT

);

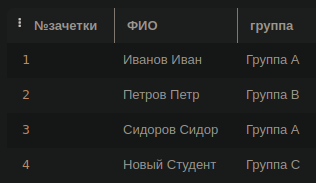
INSERT INTO Зачетки (№зачетки, ФИО, группа) VALUES

(1, 'Иванов Иван', 'Группа A'),

(2, 'Петров Петр', 'Группа B'),

(3, 'Сидоров Сидор', 'Группа A');

**Процедура:**

CREATE OR REPLACE PROCEDURE добавить\_зачетку(IN номер\_зачетки INTEGER, IN полное\_имя TEXT, IN группа TEXT)

AS

$$

BEGIN

INSERT INTO Зачетки (№зачетки, ФИО, группа) VALUES (номер\_зачетки, полное\_имя, группа);

END;

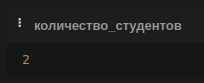
$$

LANGUAGE plpgsql;

CALL добавить\_зачетку(4, 'Новый Студент', 'Группа C');

**Функция:**

CREATE OR REPLACE FUNCTION количество\_студентов(в\_группе TEXT)

RETURNS INTEGER

AS

$$

DECLARE

студенты\_в\_группе INTEGER;

BEGIN

SELECT COUNT(\*) INTO студенты\_в\_группе FROM Зачетки WHERE группа = в\_группе;

RETURN студенты\_в\_группе;

END;

$$

LANGUAGE plpgsql;

SELECT количество\_студентов('Группа A');

**Триггер:**

CREATE OR REPLACE FUNCTION проверка\_уникальности\_зачетки()

RETURNS TRIGGER

AS

$$

BEGIN

IF EXISTS (SELECT 1 FROM Зачетки WHERE №зачетки = NEW.№зачетки) THEN

RAISE EXCEPTION 'Попытка вставки дублирующегося номера зачетки: %', NEW.№зачетки;

END IF;

RETURN NEW;

END;

$$

LANGUAGE plpgsql;

CREATE TRIGGER уникальность\_зачетки

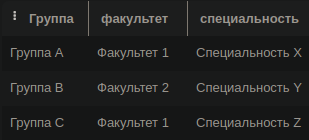
BEFORE INSERT ON Зачетки

FOR EACH ROW

EXECUTE FUNCTION проверка\_уникальности\_зачетки();

INSERT INTO Зачетки (№зачетки, ФИО, группа) VALUES (1, 'Иванов Иван', 'Группа A');

**Таблица Группа**



CREATE TABLE Группа (

Группа TEXT PRIMARY KEY,

факультет TEXT,

специальность TEXT

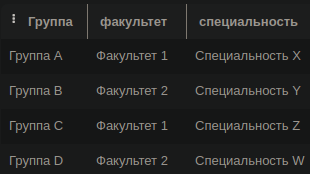
);

INSERT INTO Группа (Группа, факультет, специальность) VALUES

('Группа A', 'Факультет 1', 'Специальность X'),

('Группа B', 'Факультет 2', 'Специальность Y'),

('Группа C', 'Факультет 1', 'Специальность Z');

**Процедура:**

CREATE OR REPLACE PROCEDURE добавить\_группу(

в\_группу TEXT,

в\_факультет TEXT,

в\_специальность TEXT

)

LANGUAGE SQL

AS $$

INSERT INTO "Группа" (Группа, факультет, специальность)

VALUES (в\_группу, в\_факультет, в\_специальность);

$$;

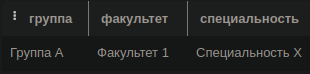
CALL добавить\_группу('Группа D', 'Факультет 2', 'Специальность W');

**Функция:**

CREATE OR REPLACE FUNCTION получить\_информацию\_о\_группе(название\_группы TEXT)

RETURNS TABLE (группа TEXT, факультет TEXT, специальность TEXT)

AS $$

BEGIN

RETURN QUERY SELECT \* FROM "Группа" WHERE Группа = название\_группы;

END;

$$ LANGUAGE plpgsql;

SELECT \* FROM получить\_информацию\_о\_группе('Группа A');

**Триггер:**

CREATE OR REPLACE FUNCTION проверка\_уникальности\_факультета()

RETURNS TRIGGER

AS $$

BEGIN

IF EXISTS (SELECT 1 FROM "Группа"

WHERE факультет = NEW.факультет

AND Группа <> NEW.Группа) THEN

RAISE EXCEPTION 'Группа с таким факультетом уже существует';

END IF;

RETURN NEW;

END;

$$ LANGUAGE plpgsql;

CREATE TRIGGER триггер\_уникальности\_факультета

BEFORE INSERT OR UPDATE

ON "Группа"

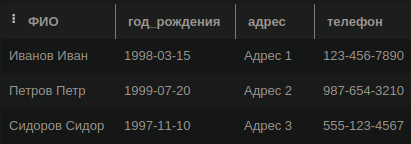
FOR EACH ROW

EXECUTE FUNCTION проверка\_уникальности\_факультета();

INSERT INTO Группа (Группа, факультет, специальность) VALUES

('Группа A', 'Факультет 1', 'Специальность X');

**Таблица Студенты:**

CREATE TABLE Студенты (

ФИО TEXT PRIMARY KEY,

год\_рождения DATE,

адрес TEXT,

телефон TEXT

);

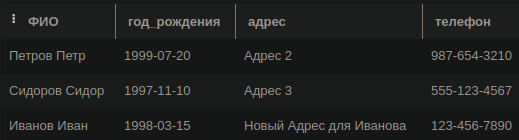
INSERT INTO Студенты (ФИО, год\_рождения, адрес, телефон) VALUES

('Иванов Иван', '1998-03-15', 'Адрес 1', '123-456-7890'),

('Петров Петр', '1999-07-20', 'Адрес 2', '987-654-3210'),

('Сидоров Сидор', '1997-11-10', 'Адрес 3', '555-123-4567');

**Процедура:**

CREATE OR REPLACE PROCEDURE обновить\_адрес(

IN p\_ФИО TEXT,

IN p\_новый\_адрес TEXT

)

AS $$

BEGIN

UPDATE Студенты SET адрес = p\_новый\_адрес WHERE ФИО = p\_ФИО;

END;

$$ LANGUAGE plpgsql;

CALL обновить\_адрес('Иванов Иван', 'Новый Адрес для Иванова');

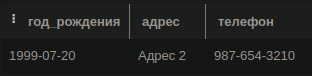
**Функция:**

CREATE OR REPLACE FUNCTION получить\_информацию(

IN p\_ФИО TEXT

) RETURNS TABLE (

год\_рождения DATE,

адрес TEXT,

телефон TEXT

)

AS $$

BEGIN

RETURN QUERY SELECT s.год\_рождения, s.адрес, s.телефон

FROM Студенты s

WHERE s.ФИО = p\_ФИО;

END;

$$ LANGUAGE plpgsql;

SELECT \* FROM получить\_информацию('Петров Петр');

**Триггер:**

CREATE OR REPLACE FUNCTION проверить\_уникальность\_ФИО()

RETURNS TRIGGER AS $$

BEGIN

IF TG\_OP = 'INSERT' OR TG\_OP = 'UPDATE' THEN

IF (SELECT COUNT(\*) FROM Студенты WHERE ФИО = NEW.ФИО) > 0 THEN

RAISE EXCEPTION 'Студент с ФИО % уже существует', NEW.ФИО;

END IF;

END IF;

RETURN NEW;

END;

$$ LANGUAGE plpgsql;

CREATE TRIGGER триггер\_уникальности\_ФИО

BEFORE INSERT OR UPDATE ON Студенты

FOR EACH ROW

EXECUTE FUNCTION проверить\_уникальность\_ФИО();

INSERT INTO Студенты (ФИО, год\_рождения, адрес, телефон) VALUES

('Петров Петр', '1999-07-20', 'Адрес 2', '987-654-3210')

**Вывод:** Изучил и решил задания по реализации хранимых процедур, функций и триггеровразных видов и фильтров в базах данных.