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

Ход работы:

Вариант 6

Вариант 6 - Студенты

1 №зачетки ФИО группа

специальнос

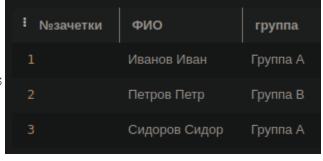
2 Группа факультет ты

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 .

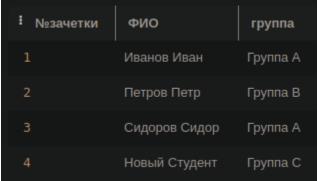
Таблица Зачетки



Процедура:

CREATE OR REPLACE PROCEDURE добавить_зачетку(IN номер_зачетки INTEGER, IN полное имя TEXT, IN группа TEXT)

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 С');



Функция:

CREATE OR REPLACE FUNCTION количество_студентов(в_группе TEXT) RETURNS INTEGER

RETURNS INTEGER

AS
\$\$

DECLARE

 cтуденты_в_группе INTEGER;

BEGIN

 SELECT COUNT(*) INTO студенты_в_группе FROM

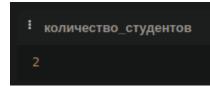
Зачетки WHERE группа = в_группе;

 RETURN студенты_в_группе;

END;

\$\$

LANGUAGE plpgsql;



```
SELECT количество_студентов ('Группа A');
```

Триггер:

```
CREATE OR REPLACE FUNCTION проверка_уникальности_зачетки()
RETURNS TRIGGER
AS
$$
BEGIN
    IF EXISTS (SELECT 1 FROM Зачетки WHERE №зачетки = NEW.№зачетки) ТНЕN
        RAISE EXCEPTION 'Попытка вставки дублирующегося номера зачетки: %', NEW.
№зачетки;
    END IF;
    RETURN NEW;
END:
$$
LANGUAGE plpqsql;
                                                Help: db error: ERROR: Попытка вставки
                                                дублирующегося номера зачетки: 1
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 А');
```

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');
```

| ! Группа | факультет | специальность |
|-----------------|-------------|-----------------|
| Группа А | Факультет 1 | Специальность X |
| Группа В | Факультет 2 | Специальность Ү |
| Группа С | Факультет 1 | Специальность Z |

Процедура:

```
CREATE OR REPLACE PROCEDURE добавить_группу(
в_группу ТЕХТ,
в_факультет ТЕХТ,
в_специальность ТЕХТ
)
LANGUAGE SQL
AS $$
INSERT INTO "Группа" (Группа, факультет,
специальность)
VALUES (в_группу, в_факультет,
в_специальность);
$$;
```

| : Группа | факультет | специальность |
|----------|-------------|-----------------|
| Группа А | Факультет 1 | Специальность X |
| Группа В | Факультет 2 | Специальность Ү |
| Группа С | Факультет 1 | Специальность Z |
| Группа D | Факультет 2 | Специальность W |

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:

| і группа | факультет | специальность |
|-----------------|-------------|-----------------|
| Группа А | Факультет 1 | Специальность X |

```
$$ LANGUAGE plpqsql;
SELECT * FROM получить_информацию_о_группе('Группа A');
Триггер:
CREATE OR REPLACE FUNCTION проверка_уникальности_факультета()
RETURNS TRIGGER
AS $$
                                                Help: db error: ERROR: Группа с таким
BEGIN
                                                факультетом уже существует
    IF EXISTS (SELECT 1 FROM "Группа"
               WHERE факультет =
NEW.факультет
                 AND Γρуппа <> NEW.Γρуппа) THEN
        RAISE EXCEPTION 'Группа с таким факультетом уже существует';
    END IF;
    RETURN NEW;
$$ 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 А', 'Факультет 1', 'Специальность Х');
```

Таблица Студенты:

| CREATE TABLE Студенты (ФИО ТЕХТ PRIMARY KEY, | : ФИО | год_рождения | адрес | телефон |
|---|---------------|--------------|---------|--------------|
| | Иванов Иван | 1998-03-15 | Адрес 1 | 123-456-7890 |
| год_рождения DATE, aдрес TEXT, | Петров Петр | 1999-07-20 | Адрес 2 | 987-654-3210 |
| телефон TEXT); | Сидоров Сидор | 1997-11-10 | Адрес 3 | 555-123-4567 |
| 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 (

| | : ФИО | год_рождения | адрес | телефон |
|-----|---------------|--------------|-------------------------|--------------|
| RΕ | Петров Петр | 1999-07-20 | Адрес 2 | 987-654-3210 |
| | Сидоров Сидор | 1997-11-10 | Адрес 3 | 555-123-4567 |
|) B | Иванов Иван | 1998-03-15 | Новый Адрес для Иванова | 123-456-7890 |

Иван', 'Новый Адрес для Иванова');

```
Функция:
```

```
CREATE OR REPLACE FUNCTION получить_информацию (
    IN p_ΦMO TEXT
) RETURNS TABLE (
    год_рождения DATE,
    адрес TEXT,
    телефон ТЕХТ
                                                                              телефон
                                                    год рождения
                                                                    адрес
)
AS $$
                                                  1999-07-20
                                                                             987-654-3210
                                                                   Адрес 2
BEGIN
    RETURN QUERY SELECT s.год_рождения,
s.адрес, s.телефон
                 FROM Студенты s
                 WHERE s.\PhiMO = p_\PhiMO;
END:
$$ LANGUAGE plpgsql;
SELECT * FROM получить_информацию ('Петров Петр');
Триггер:
CREATE OR REPLACE
                         Help: db error: ERROR: Студент с ФИО Петров
FUNCTION
                         Петр уже существует
проверить_уникальность_ФИО()
RETURNS TRIGGER AS $$
BEGIN
    IF TG_OP = 'INSERT' OR TG_OP = 'UPDATE' THEN
        IF (SELECT COUNT(*) FROM CTYMENTЫ WHERE ФИО = NEW.ФИО) > 0 THEN
            RAISE EXCEPTION 'Студент с ФИО % уже существует', NEW.ФИО;
    END IF;
    RETURN NEW;
END;
$$ 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 триггеров разных видов и фильтров в базах данных.