

Задание 4. Создайте хранимые процедуры:

1. Для повышения стипендии отличникам на 10%

Сначала создадим функцию, которая поможет нам посчитать этот средний балл для каждого студента, и занесем его в столбец `average_grade`:

```
CREATE OR REPLACE FUNCTION calculate_average_grade()
RETURNS void AS $$
DECLARE
    student_id INTEGER;
    avg_grade NUMERIC;
BEGIN
    FOR student_id IN (SELECT id_student FROM lab.student)
    LOOP
        SELECT AVG(grade) INTO avg_grade
        FROM lab.attestation
        WHERE id_is_studying = (SELECT id_is_studying FROM lab.is_studying WHERE id_student =
student_id);

        UPDATE lab.student
        SET average_grade = avg_grade
        WHERE id_student = student_id;
    END LOOP;
END;
$$ LANGUAGE plpgsql;

SELECT calculate_average_grade();
```

Видим актуальные данные:

	<code>id_student</code> [PK] integer	<code>full_name</code> character varying	<code>average_grade</code> numeric	<code>scholarship</code> numeric
1	1	Иван Иванов	4.25	[null]
2	2	Мария Смирнова	2.5	[null]
3	3	Георгий Константинов	3	[null]
4	4	Дмитрий Рубежкин	5	[null]

Затем приступаем к функции увеличения стипендии:

```
CREATE OR REPLACE PROCEDURE increase_scholarship()
AS $$
DECLARE
    avg_grade numeric;
BEGIN
    IF NOT EXISTS (SELECT 1 FROM information_schema.columns
                    WHERE table_name = 'student' AND column_name = 'average_grade')
    THEN
        ALTER TABLE lab.student ADD COLUMN average_grade numeric;
    END IF;

    IF NOT EXISTS (SELECT 1 FROM information_schema.columns
```

```

        WHERE table_name = 'student' AND column_name = 'scholarship')

    THEN
        ALTER TABLE lab.student ADD COLUMN scholarship numeric;
    END IF;
SELECT
    UPDATE lab.student
    SET scholarship = type_scholarship.amount_sch * 1.1
    FROM lab.type_scholarship, lab.receiving_sch
    WHERE lab.receiving_sch.id_sch = lab.type_scholarship.id_sch
    AND lab.student.average_grade >= 4.5;

    SELECT AVG(grade) INTO avg_grade
    FROM lab.attestation;

END;
$$ LANGUAGE plpgsql;

CALL increase_scholarship();

```

Query Query History

1 SELECT * FROM lab.student

Data Output Messages Graph Visualiser x Notifications

	id_student [PK] integer	full_name character varying	average_grade numeric	scholarship numeric
1	1	Иван Иванов	4.25	[null]
2	2	Мария Смирнова	2.5	[null]
3	3	Георгий Константинов	3	[null]
4	4	Дмитрий Рубежкин	5	11000.0

Новая стипендия для отличника Дмитрия Рубежкина – 11000 рублей.

2. Для перевода студентов на следующий курс.

```

CREATE OR REPLACE PROCEDURE add_year()
AS $$
DECLARE
BEGIN
    UPDATE lab.l_group
    SET year = year + 1;
END;
$$ LANGUAGE plpgsql;

CALL add_year();

```

Смотрим таблицу и видим, что первокурсники действительно повзрослели, и вся группа теперь 2 курс.

Query

Query History

1

CREATE OR REPLACE PROCEDURE add_year()

2

AS \$\$

3

DECLARE

4

BEGIN

5

UPDATE lab.l_group

6

SET year = year + 1;

7

END;

8

\$\$ LANGUAGE plpgsql;

9

10

CALL add_year();

11

12

SELECT * FROM lab.l_group;

Data Output

Messages

Graph Visualiser

×

Notifications

	id_group [PK] character varying	from date	to date	amount_students integer	year integer	group_number character varying	id_plan character varying
1	Group1	2021-09-01	[null]	30	2	G1	Plan1
2	Group2	2021-09-01	[null]	25	2	G2	Plan1

3. Для изменения оценки при успешной пересдаче экзамена.

```
CREATE OR REPLACE PROCEDURE update_grade(student_id VARCHAR, id_subject VARCHAR,
new_grade INTEGER)
AS $$
BEGIN
    UPDATE lab.attestation
    SET grade = new_grade,
        number_attempt = number_attempt + 1
    WHERE id_is_studying = student_id AND id_program_disc = id_subject;
END;
$$ LANGUAGE plpgsql;
```

Data Output

Messages

Graph Visualiser

×

Notifications

	att_date [PK] date	grade real	number_attempt integer	id_program_disc character varying	id_is_studying character varying	id_teacher integer
1	2023-05-01	4.5	1	PD1	IS1	1
2	2023-05-02	3	1	PD1	IS2	2
3	2023-05-06	4	1	PD2	IS1	3
4	2023-05-07	2	1	PD2	IS2	4
5	2023-06-25	4	2	PD1	IS3	1
6	2023-05-29	2	2	PD1	IS3	1
7	2023-05-27	5	1	PD1	IS4	2

Наблюдаем за троечником IS2.

После процедуры:

	att_date [PK] date	grade real	number_attempt integer	id_program_disc character varying	id_is_studying character varying	id_teacher integer
1	2023-05-01	4.5	1	PD1	IS1	1
2	2023-05-06	4	1	PD2	IS1	3
3	2023-05-07	2	1	PD2	IS2	4
4	2023-06-25	4	2	PD1	IS3	1
5	2023-05-29	2	2	PD1	IS3	1
6	2023-05-27	5	1	PD1	IS4	2
7	2023-05-02	5	2	PD1	IS2	2

Мы UPDATEнули номер попытки и оценку учащегося.

Задание 5. Триггеры

Задаем следующий триггер:

Триггер для автоматического перевода студентов на следующий курс.

Скриншот до срабатывания триггера.

	id_group [PK] character varying	from date	to date	amount_students integer	year integer	group_number character varying	id_plan character varying
1	Group1	2021-09-01	[null]	30	2	G1	Plan1
2	Group2	2021-09-01	[null]	25	2	G2	Plan1

```
CREATE OR REPLACE FUNCTION update_year_trigger()
RETURNS TRIGGER AS $$
BEGIN
    IF NEW.year <= 4 THEN
        NEW.year = NEW.year + 1;
    END IF;
    RETURN NEW;
END;
$$ LANGUAGE plpgsql;
```








```
CREATE TRIGGER update_year_trigger
AFTER INSERT OR UPDATE ON lab.l_group
FOR EACH ROW
EXECUTE FUNCTION update_year_trigger();
```

Делаем UPDATE нашей таблицы:

```
UPDATE lab.l_group
```

```
SET year = year;
```

Получаем следующее:

	id_group [PK] character varying 	from date 	to date 	amount_students integer 	year integer 	group_number character varying 	id_plan character varying 
1	Group1	2021-09-01	[null]	30	3	G1	Plan1
2	Group2	2021-09-01	[null]	25	3	G2	Plan1

Вывод:

В ходе лабораторной работы №3 мы изучили работу с процедурами и функциями, а также задали триггер для нашей базы данных, улучшающий работу с ней.