**ЛАБОРАТОРНАЯ РАБОТА №3**

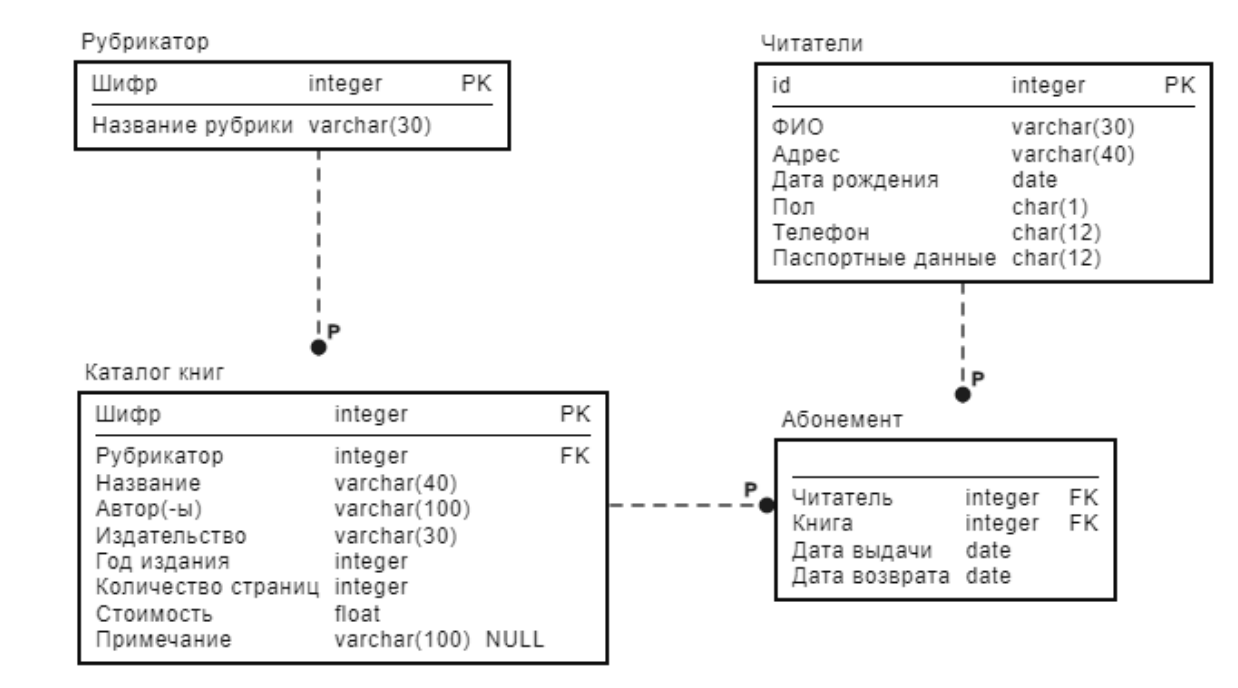
**«Язык SQL. Генераторы. Функции. Триггеры»**

**3.1 Цель работы**

Выработать у обучающихся практические навыки по работе с реляционными базами данных, ознакомить с принципом работы генераторов, функций и триггеров.

**3.2 Индивидуальный вариант**

Вариант лабораторной работы представлен на картинке 3.1.

Рисунок 3.1 – Вариант лабораторной работы

**3.3 Ход выполнения работы**

3.3.1 После выполнения запросов на создание таблицы, были созданы генераторы для всех полей идентификаторов. В листинге 3.2 показан запрос на создание генератора для таблицы дисциплин, а в листинге 3.3 запрос на заполнение этой таблицы данными с использованием генератора.

Листинг 3.2 – Создание генератора для таблицы дисциплин

create sequence discipline\_id start 1;

Листинг 3.3 – Заполнение таблицы дисциплин данными

INSERT INTO discipline (id, name) VALUES (nextval('discipline\_id'), 'Инженерная графика');

INSERT INTO discipline (id, name) VALUES (nextval('discipline\_id'), 'Теория механизмов и машин');

INSERT INTO discipline (id, name) VALUES (nextval('discipline\_id'), 'Материаловедение');

INSERT INTO discipline (id, name) VALUES (nextval('discipline\_id'), 'Электротехника');

INSERT INTO discipline (id, name) VALUES (nextval('discipline\_id'), 'Термодинамика');

INSERT INTO discipline (id, name) VALUES (nextval('discipline\_id'), 'Гидравлика');

INSERT INTO discipline (id, name) VALUES (nextval('discipline\_id'), 'Сопротивление материалов');

INSERT INTO discipline (id, name) VALUES (nextval('discipline\_id'), 'Автоматизация производственных процессов');

INSERT INTO discipline (id, name) VALUES (nextval('discipline\_id'), 'Мехатроника');

INSERT INTO discipline (id, name) VALUES (nextval('discipline\_id'), 'Металлообработка');

INSERT INTO discipline (id, name) VALUES (nextval('discipline\_id'), 'Программирование');

INSERT INTO discipline (id, name) VALUES (nextval('discipline\_id'), 'Компьютерные сети');

INSERT INTO discipline (id, name) VALUES (nextval('discipline\_id'), 'Системы управления');

INSERT INTO discipline (id, name) VALUES (nextval('discipline\_id'), 'Оптика');

INSERT INTO discipline (id, name) VALUES (nextval('discipline\_id'), 'Нанотехнологии');

То же самое было проделано с таблицей, содержащей информацию о кафедрах. В листинге 3.4 продемонстрирован запрос на создание генератора для этой таблицы, а в листинге 3.5 отражена вставка данных.

Листинг 3.4 – Создание генератора для таблицы кафедр

create sequence department\_id start 1;

Листинг 3.5 – Заполнение таблицы кафедр данными

INSERT INTO department (id, name, head) VALUES (nextval('department\_id'), 'Кафедра информатики', 'Иванов И.И.');

INSERT INTO department (id, name, head) VALUES (nextval('department\_id'), 'Кафедра математики', 'Петров П.П.');

INSERT INTO department (id, name, head) VALUES (nextval('department\_id'), 'Кафедра физики', 'Сидоров С.С.');

INSERT INTO department (id, name, head) VALUES (nextval('department\_id'), 'Кафедра химии', 'Андреев А.А.');

INSERT INTO department (id, name, head) VALUES (nextval('department\_id'), 'Кафедра биологии', 'Кузнецов К.К.');

INSERT INTO department (id, name, head) VALUES (nextval('department\_id'), 'Кафедра экономики', 'Морозов М.М.');

INSERT INTO department (id, name, head) VALUES (nextval('department\_id'), 'Кафедра социологии', 'Новиков Н.Н.');

INSERT INTO department (id, name, head) VALUES (nextval('department\_id'), 'Кафедра психологии', 'Федоров Ф.Ф.');

INSERT INTO department (id, name, head) VALUES (nextval('department\_id'), 'Кафедра философии', 'Васильев В.В.');

INSERT INTO department (id, name, head) VALUES (nextval('department\_id'), 'Кафедра истории', 'Григорьев Г.Г.');

INSERT INTO department (id, name, head) VALUES (nextval('department\_id'), 'Кафедра искусств', 'Дмитриев Д.Д.');

INSERT INTO department (id, name, head) VALUES (nextval('department\_id'), 'Кафедра географии', 'Жуков Ж.Ж.');

INSERT INTO department (id, name, head) VALUES (nextval('department\_id'), 'Кафедра литературы', 'Ильин И.И.');

INSERT INTO department (id, name, head) VALUES (nextval('department\_id'), 'Кафедра инженерии', 'Королев К.К.');

INSERT INTO department (id, name, head) VALUES (nextval('department\_id'), 'Кафедра механики', 'Лебедев Л.Л.');

Затем был создан генератор для создания идентификаторов таблицы преподавателей, код для этого запроса показан в листинге 3.6, а заполнение данными этой таблице в листинге 3.7.

Листинг 3.6 – Создание генератора для таблицы преподавателей

create sequence teacher\_id start 1;

Листинг 3.7 – Заполнение таблицы преподавателей данными

INSERT INTO teacher VALUES (nextval('teacher\_id'), 'Иванов Иван Иванович', 'M', '1975-05-21', 1, 'Профессор', 'Доктор наук', 'Физико-математические науки');

INSERT INTO teacher VALUES (nextval('teacher\_id'), 'Петров Петр Петрович', 'M', '1980-03-15', 2, 'Доцент', 'Кандидат наук', 'Химические науки');

INSERT INTO teacher VALUES (nextval('teacher\_id'), 'Сидорова Анна Сергеевна', 'F', '1982-11-02', 3, 'Ассистент', 'Магистр', 'Биологические науки');

INSERT INTO teacher VALUES (nextval('teacher\_id'), 'Кузнецов Николай Алексеевич', 'M', '1970-08-19', 4, 'Старший преподаватель', 'Кандидат наук', 'Экономические науки');

INSERT INTO teacher VALUES (nextval('teacher\_id'), 'Андреева Марина Ивановна', 'F', '1978-12-23', 5, 'Доцент', 'Кандидат наук', 'Юридические науки');

INSERT INTO teacher VALUES (nextval('teacher\_id'), 'Морозова Елена Викторовна', 'F', '1985-06-11', 6, 'Ассистент', 'Магистр', 'Социологические науки');

INSERT INTO teacher VALUES (nextval('teacher\_id'), 'Новиков Сергей Дмитриевич', 'M', '1977-04-28', 7, 'Профессор', 'Доктор наук', 'Психологические науки');

INSERT INTO teacher VALUES (nextval('teacher\_id'), 'Федорова Анна Петровна', 'F', '1983-09-14', 8, 'Старший преподаватель', 'Кандидат наук', 'Философские науки');

INSERT INTO teacher VALUES (nextval('teacher\_id'), 'Васильев Андрей Николаевич', 'M', '1981-02-17', 9, 'Доцент', 'Кандидат наук', 'Исторические науки');

INSERT INTO teacher VALUES (nextval('teacher\_id'), 'Григорьева Елена Алексеевна', 'F', '1987-07-08', 10, 'Ассистент', 'Магистр', 'Искусствоведение');

INSERT INTO teacher VALUES (nextval('teacher\_id'), 'Дмитриев Алексей Сергеевич', 'M', '1979-01-05', 11, 'Старший преподаватель', 'Кандидат наук', 'Географические науки');

INSERT INTO teacher VALUES (nextval('teacher\_id'), 'Жуков Николай Иванович', 'M', '1984-11-19', 12, 'Доцент', 'Кандидат наук', 'Литературоведение');

INSERT INTO teacher VALUES (nextval('teacher\_id'), 'Ильина Мария Сергеевна', 'F', '1982-03-21', 13, 'Ассистент', 'Магистр', 'Инженерные науки');

INSERT INTO teacher VALUES (nextval('teacher\_id'), 'Королев Владимир Алексеевич', 'M', '1976-10-14', 14, 'Профессор', 'Доктор наук', 'Механика');

INSERT INTO teacher VALUES (nextval('teacher\_id'), 'Лебедева Татьяна Ивановна', 'F', '1980-05-06', 15, 'Доцент', 'Кандидат наук', 'Компьютерные науки');

Последним был создан генератор для идентификатора поля таблицы расписания, запрос для его создания показан в листинге 3.8, а вставка 15 записей в эту таблицу в листинге 3.9.

Листинг 3.6 – Создание генератора для таблицы преподавателей

create sequence timetable\_id start 1;

Листинг 3.7 – Заполнение таблицы преподавателей данными

INSERT INTO timetable VALUES (nextval('timetable\_id'), 1, '101', 'Лекция', 1, 'пн', '101');

INSERT INTO timetable VALUES (nextval('timetable\_id'), 2, '102', 'Семинар', 2, 'вт', '202');

INSERT INTO timetable VALUES (nextval('timetable\_id'), 3, '103', 'Лабораторная', 3, 'ср', '303');

INSERT INTO timetable VALUES (nextval('timetable\_id'), 4, '104', 'Лекция', 4, 'чт', '404');

INSERT INTO timetable VALUES (nextval('timetable\_id'), 5, '105', 'Семинар', 5, 'пт', '505');

INSERT INTO timetable VALUES (nextval('timetable\_id'), 6, '106', 'Лабораторная', 6, 'сб', '606');

INSERT INTO timetable VALUES (nextval('timetable\_id'), 7, '107', 'Лекция', 7, 'пн', '107');

INSERT INTO timetable VALUES (nextval('timetable\_id'), 8, '108', 'Семинар', 8, 'вт', '208');

INSERT INTO timetable VALUES (nextval('timetable\_id'), 9, '109', 'Лабораторная', 9, 'ср', '309');

INSERT INTO timetable VALUES (nextval('timetable\_id'), 10, '110', 'Лекция', 10, 'чт', '410');

INSERT INTO timetable VALUES (nextval('timetable\_id'), 11, '111', 'Семинар', 11, 'пт', '511');

INSERT INTO timetable VALUES (nextval('timetable\_id'), 12, '112', 'Лабораторная', 12, 'сб', '612');

INSERT INTO timetable VALUES (nextval('timetable\_id'), 13, '113', 'Лекция', 13, 'пн', '102');

INSERT INTO timetable VALUES (nextval('timetable\_id'), 14, '114', 'Семинар', 14, 'вт', '203');

INSERT INTO timetable VALUES (nextval('timetable\_id'), 15, '115', 'Лабораторная', 15, 'ср', '304');

3.3.2 Далее были написаны функции по заданию. В начале была написана функция, которая рассчитывает конец занятия по его типу и начальному времени.

Листинг 3.8 – Код функции, которая рассчитывает конец занятия

CREATE OR REPLACE FUNCTION end\_of\_lesson(start\_time TIME, lesson\_type VARCHAR)

RETURNS TIME

LANGUAGE plpgsql

AS

$$

BEGIN

IF lesson\_type = 'Лекция' OR lesson\_type = 'Семинар' THEN

RETURN start\_time + INTERVAL '1 hour 20 minutes';

ELSIF lesson\_type = 'Лабораторная' THEN

RETURN start\_time + INTERVAL '1 hour 30 minutes';

ELSE

RETURN NULL;

END IF;

END;

$$;

Следующая функция была написана для того, чтобы сделать отчет о некорректных данных в расписании для преподавателей кафедры путем добавления данных в специальную таблицу «Отчет». Расписание считается некорректным, если у одного преподавателя более 4-х занятий в день или если время разных занятий у одного преподавателя совпадает или пересекается. Код этой функции содержится в листинге 3.9.

Листинг 3.9 –Создание ограничения таблицы Типы\_растений

create function make\_report() returns void

language plpgsql

as

$$

declare

rec record;

lessons integer;

first\_lesson\_end time;

next\_lesson\_end time;

begin

for rec in (

select tch.fio, t1.teacher, t1.weekday, t1.lesson\_type as first\_lesson\_type,

t1.lesson\_time as first\_lesson, t2.lesson\_type as next\_lesson\_type, t2.lesson\_time as next\_lesson

from timetable t1

left join timetable t2 on t1.teacher = t2.teacher and t1.weekday = t2.weekday and t1.lesson\_time < t2.lesson\_time

join teacher tch on tch.id = t1.teacher

where not exists (

select 1 from timetable t3

where t3.teacher = t1.teacher and t3.weekday = t1.weekday and t3.lesson\_time > t1.lesson\_time and t3.lesson\_time < t2.lesson\_time

)

order by t1.teacher, t1.lesson\_time

) loop

select count(\*) into lessons

from timetable

where teacher = rec.teacher and weekday = rec.weekday;

first\_lesson\_end := end\_of\_lesson(rec.first\_lesson, rec.first\_lesson\_type);

next\_lesson\_end := end\_of\_lesson(rec.next\_lesson, rec.next\_lesson\_type);

if lessons > 4

or (first\_lesson\_end is not null and rec.next\_lesson is not null and first\_lesson\_end > rec.next\_lesson) then

insert into report (fio, weekday, total\_lessons, first\_lesson\_start, next\_lesson\_start)

values (rec.fio, rec.weekday, lessons, rec.first\_lesson, rec.next\_lesson);

else

insert into report (fio, weekday, total\_lessons, first\_lesson\_start, next\_lesson\_start)

values (rec.fio, rec.weekday, null, null, null);

end if;

end loop;

end;

$$;

alter function make\_report() owner to postgres;

Заполненная таблица с расписанием показана на рисунке 3.2.

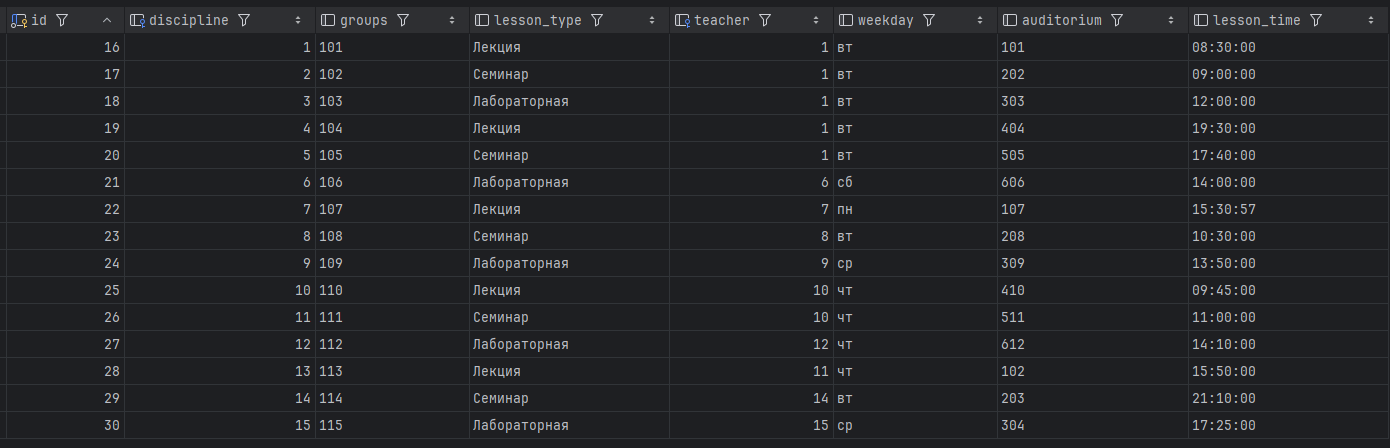


Рисунок 3.2 – Заполненная таблица расписания

На рисунке 3.3 показана заполненная таблица с отчетом, где указаны все ошибки найденные функции.

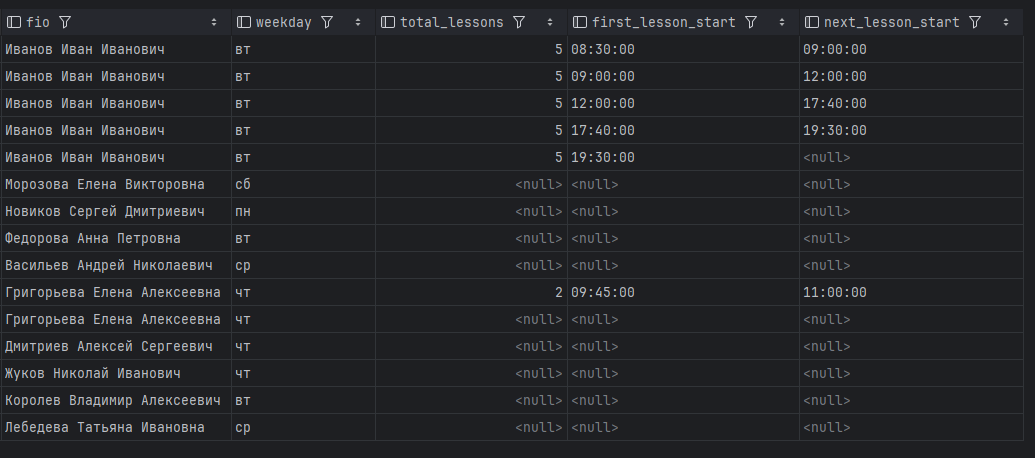


Рисунок 3.3 – Заполненная таблица расписания

3.3.3 Далее в ходе лабораторной работы были написаны 2 триггера. Первый триггер проверяет при добавлении значения полей расписания: тип занятий, день недели, аудиторию, занятия не могут начинаться раньше 9.00 и заканчиваться позднее, чем в 21.00. Код триггерной функции показан в листинге 3.10, а в листинге 3.11 код самого триггера.

Листинг 3.10 – Триггерная функция проверки полей таблицы расписания

create or replace function timetable\_trigger\_fnc()

returns trigger as

$$

declare

end\_time time;

begin

if not ((new.lessdon\_type = 'Лекция') or (new.lessdon\_type = 'Семинар')

or (new.lessdon\_type = 'Лабораторная')) then

raise exception 'not allowable lesson type';

end if;

if (new.lessdon\_type not in ('пн', 'вт', 'ср', 'чт', 'пт', 'сб')) then

raise exception 'not allowable weekday type';

end if;

if (new.auditorium not like '???') then

raise exception 'not allowable auditorium value';

end if;

end\_time = end\_of\_lesson(new.lesson\_time, new.lesson\_type);

if (new.lesson\_time < '9:00' or end\_time > '21:00') then

raise exception 'not allowable lesson time';

end if;

return new;

end

$$

language plpgsql;

Листинг 3.11 – Код триггера для таблицы расписаний

create trigger timetable\_insert\_trigger

before insert

on timetable

for each row

execute procedure timetable\_trigger\_fnc();

На рисунках 3.4 – 3.7 показано то, как выглядят все ошибки при попытке вставить неверные данные в таблицу расписаний.

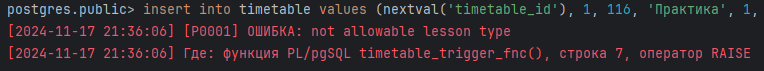


Рисунок 3.4 – Ошибка при добавлении неправильного типа занятия

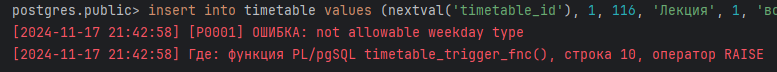


Рисунок 3.5 – Ошибка при добавлении неправильного дня недели

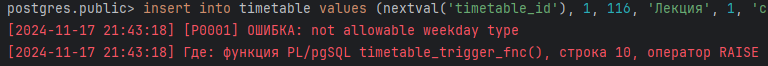


Рисунок 3.6 – Ошибка при добавлении неправильного номера кабинета

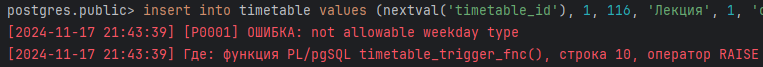


Рисунок 3.7 – Ошибка при добавлении неправильного времени занятия

Второй триггер копирует изменения таблицы с преподавателями, датой изменения и пользователем, который эти изменения ввел. Код триггерной функции показан в листинге 3.12, а в листинге 3.13 код самого триггера.

Листинг 3.12 – Триггерная функция вставки измененных строк в архив

create or replace function archive\_teachers\_fnc()

returns trigger as

$$

begin

insert into teacher\_archive values (new.id, new.fio, new.gender, new.birth, new.department, new.post, new.academic\_title, new.academic\_degree, current\_date, current\_user);

end

$$

language plpgsql;

Листинг 3.11 – Код триггера для вставки измененных строк в архив

create trigger timetable\_insert\_trigger

after insert

on teacher

for each row

execute procedure archive\_teachers\_fnc();

На рисунке 3.8 показано то, как выглядит таблица архива после внесения изменений в таблице преподавателей.

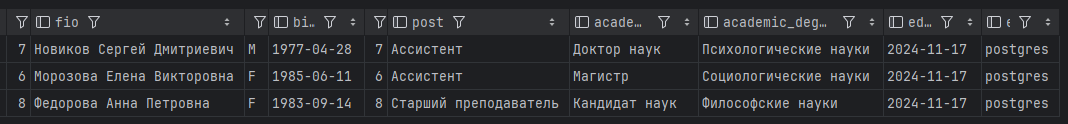


Рисунок 3.8 – Результат работы триггера сохранения изменений в архив

**Выводы**

В ходе выполнения работы были получены знания о генераторах, способах работы с ними и примеры практического использования. Также были рассмотерны функции в СУБД PostgreSQL, варианты их задания вызова и использования. Помимо этого были рассмотрены триггеры и триггерные функции, изучены вопросы касающиеся их создания и применения. Далее была создана база данных деканата по предложенному варианту. Для создания полей-идентификаторов использовались генераторы. После вставки данных в таблицы было создано несколько функций: одна проверяет вхоит ли время занятия допустимый диапазон, а вторая выводит ошибки в расписании. Также было создано два триггера для базы данных: первый проверяет все значения полей при вставке данных в таблицу расписания, а второй сохраняет изменения в таблице преподавателей в архив. В конце был написан отчет по лабораторной работе.