ФГБОУ ВО
«Заполярный государственный университет им. Н.М. Федоровского»

Кафедра __ИСиТ__
Специальность __ИС-21__

ОТЧЕТ
о выполнении лабораторной работы

Выполнил:

Сидельников М.Э., ИС-21

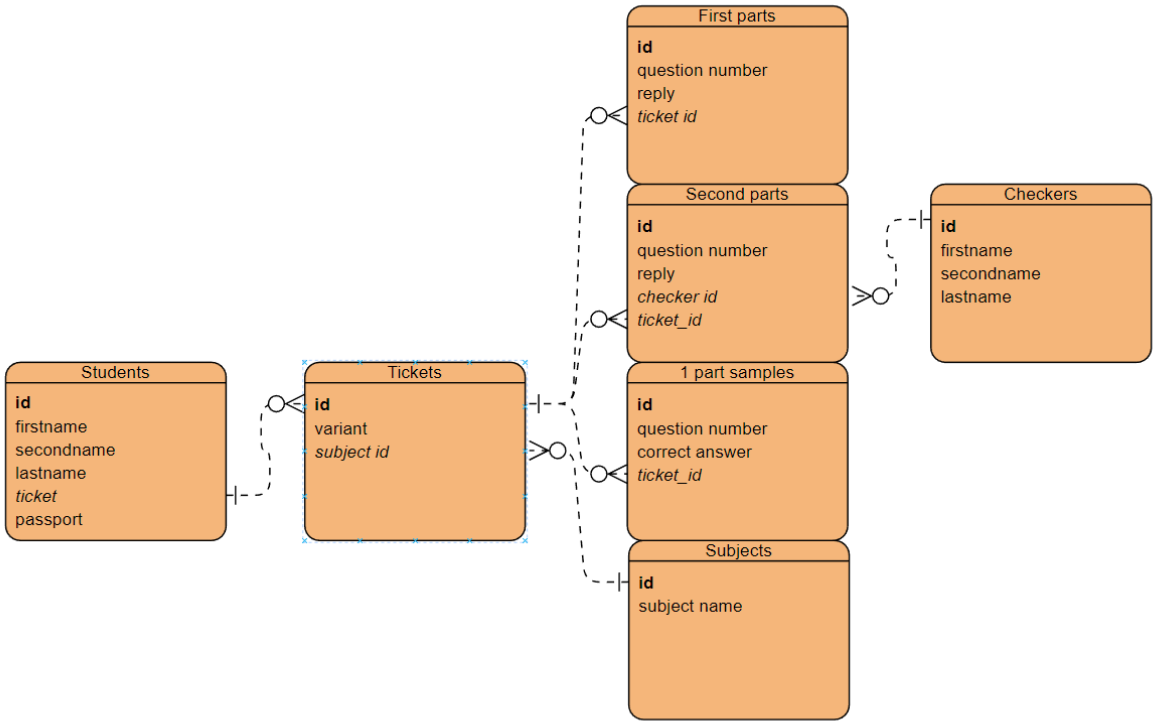
Дата:
« 25 » февраля 2023 г.

Норильск 2023

Вариант 15

Задание: спроектировать реляционную базу данных, отражающую процесс обработки результатов ЕГЭ.

Модель базы данных (Entity Relationship Diagram)



Сущность students содержит в себе информацию об ученике, сдающем экзамен. Она обладает уникальным полем passport (значения в нём не могут повторяться). Также она связана с сущностью tickets посредством связи «один ко многим».

Сущность tickets представляет собой билет конкретного ученика. У неё есть поле subject_id, представляющее собой ссылку на поле соответствующей сущности subjects («один ко многим»).

Сущность subjects есть список всех предметов, доступных для сдачи.

Сущность first_parts – первая часть билетов экзаменуемых. Она обладает полем ticket_id, ссылающемся на поле соответствующей сущности tickets («один ко многим»).

Сущность first_part_samples является ответами/ключом к первой части. Она обладает полем ticket_id, ссылающемся на поле соответствующей сущности tickets («один ко многим»).

Сущность second_parts – вторая часть билетов экзаменуемых. Она обладает полем ticket_id, ссылающемся на поле соответствующей сущности tickets («один ко многим»). Также она обладает полем checker_id, ссылающемся на поле соответствующей сущности checkers («один ко многим»).

Сущность checkers есть список проверяющих – людей, которые проверяют вторую часть билета.

Код создания БД

```
create table checkers
(
	checker_id serial not NULL,
	firstname text not NULL,
	secondname text not NULL,
	lastname text not NULL,
	constraint checkers_pkey primary key (checker_id)
);

create table subjects
(
	subject_id serial not NULL,
	subject_name text not NULL,
	constraint subjects_pkey primary key (subject_id)
);

create table tickets
(
	ticket_id serial not NULL,
	variant integer not NULL,
	check (variant > 0),
	subject_id integer not NULL,
	check (subject_id > 0),
	constraint tickets_pkey primary key (ticket_id),
	constraint subject_id_fkey foreign key (subject_id)
		references subjects (subject_id)
);

create table first_part_samples
(
	first_part_sample_id serial not NULL,
	ticket_id integer not NULL,
	check (ticket_id > 0),
	question_number integer not NULL,
	check (question_number > 0),
	correct_answer smallint not NULL,
	check (correct_answer > 0),
```

```
	constraint first_part_samples_pkey primary key (first_part_sample_id),
	constraint ticket_id foreign key (ticket_id)
		references tickets (ticket_id)
);

create table first_parts
(
	first_part_id serial not NULL,
	question_number integer not NULL,
	check (question_number > 0),
	reply smallint not NULL,
	check (reply > 0),
	ticket_id integer not NULL,
	check (ticket_id > 0),
	constraint first_parts_pkey primary key (first_part_id),
	constraint ticket_id_fkey foreign key (ticket_id)
		references tickets (ticket_id)
);

create table second_parts
(
	second_part_id serial not NULL,
	question_number integer not NULL,
	check (question_number > 0),
	reply text not NULL,
	checker_id integer not NULL,
	check (checker_id > 0),
	ticket_id integer not NULL,
	check (ticket_id > 0),
	constraint second_parts_pkey primary key (second_part_id),
	constraint cheker_id_fkey foreign key (checker_id)
		references checkers (checker_id),
	constraint ticket_id_fkey foreign key (ticket_id)
		references tickets (ticket_id)
);

create table students
(
```

```
        student_id serial not NULL,
        firstname text not NULL,
        secondname text not NULL,
        lastname text not NULL,
        passport integer not NULL,
        check (passport > 0),
        ticket_id integer not NULL,
        unique (passport),
        constraint students_pkey primary key (student_id),
        constraint ticket_id_fkey foreign key (ticket_id)
                references tickets (ticket_id)
);

insert into checkers values
(default, 'Михаил', 'Петрович', 'Жмышенко'),
(default, 'Илья', 'Мавкарович', 'Попов'),
(default, 'Жанна', 'Алексеевна', 'Кулиш');

insert into subjects values
(default, 'Бэброведение'),
(default, 'Мемология'),
(default, 'Великий и могучий'),
(default, 'Матеша');

insert into tickets values
(default, 1, 1),
(default, 3, 3),
(default, 2, 2),
(default, 4, 1);

insert into first_parts values
(default, 1, 1, 1),
(default, 2, 3, 1),
(default, 3, 2, 1),
(default, 4, 1, 1);

insert into second_parts values
(default, 1, 'Абоба', 1, 1),
```

(default, 2, 'А я на пудже', 1, 1),

(default, 3, 'Кто', 1, 1),

(default, 4, 'Что', 1, 1);

insert into first_part_samples values

(default, 1, 1, 1),

(default, 1, 2, 1),

(default, 1, 3, 1),

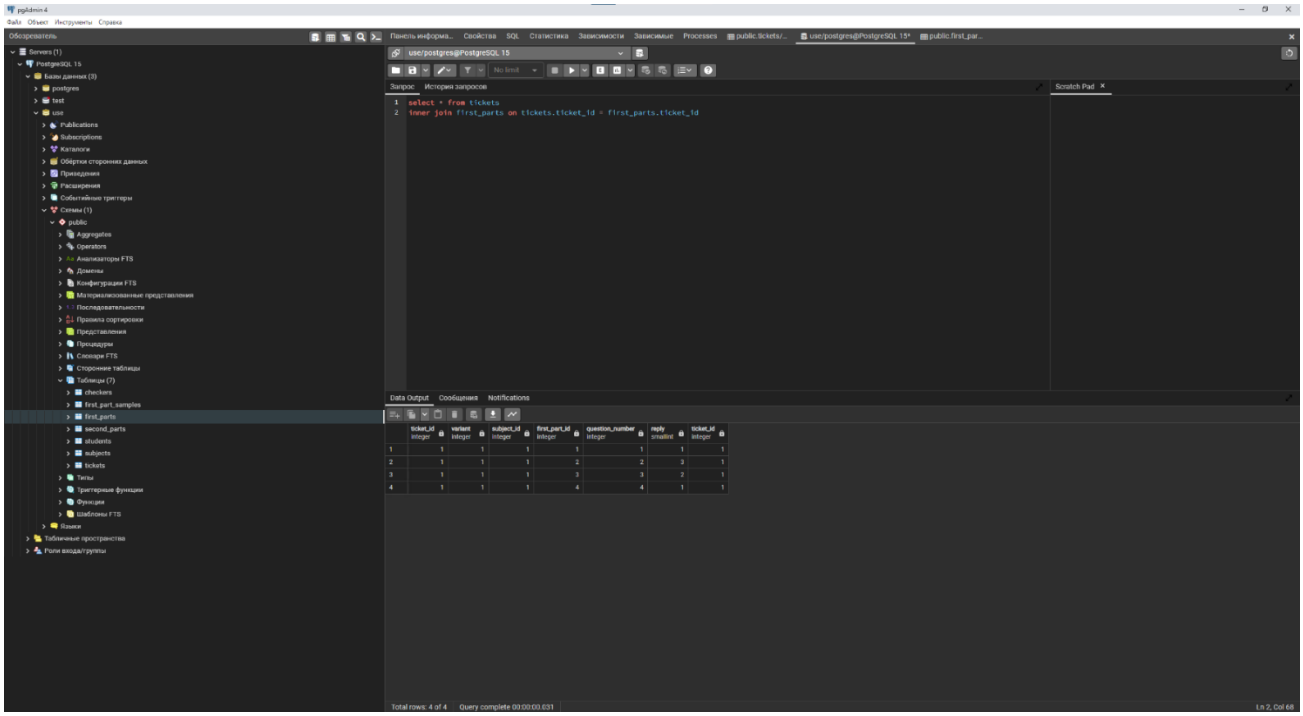
(default, 1, 4, 1);

insert into students values

(default, 'Олег', 'Михайлович', 'Чебурашкин', 123131, 1),

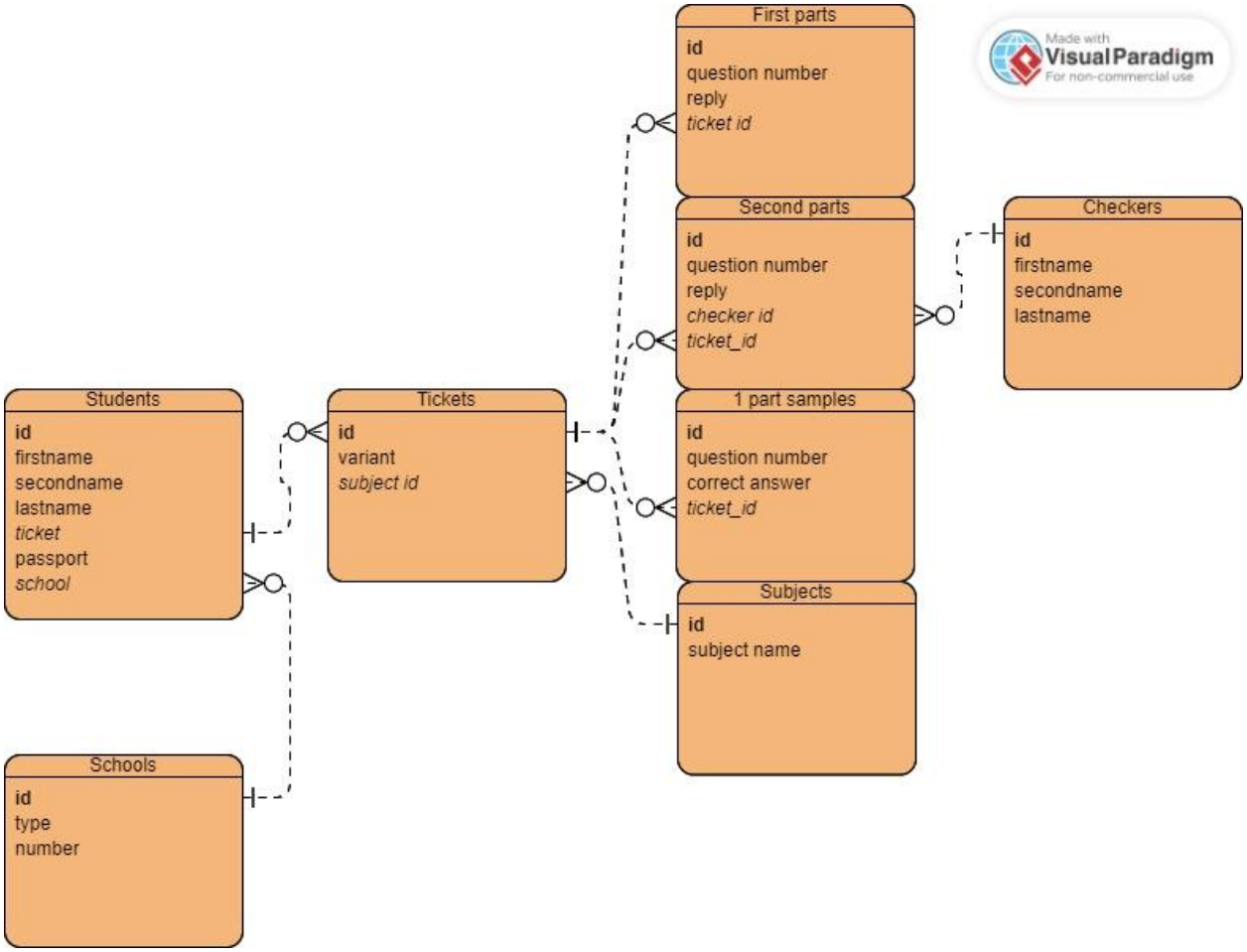
(default, 'Виктор', 'Олегович', 'Гачимучин', 2232121, 2);

Скриншот экспортированной базы данных в СУБД



Изменение базы данных

Скриншот ERD



Создадим сущность schools, которая будет отражать школы, в которых обучаются ученики. Добавим к сущности student поле school_id, являющееся ссылкой на первичный ключ сущности schools. Так мы реализуем связь «один ко многим».

Код изменений

```
create table schools
(
	school_id serial not NULL,
	type text not NULL,
	number integer not NULL,
	check (number > 0),
	constraint school_id_pk primary key (school_id)
);
```

```
alter table students
        add column school_id integer,
        add constraint school_id_fkey foreign key (school_id)
                references schools (school_id);

insert into schools values
(default, 'Лицей', 3),
(default, 'Гимназия', 5);

update students set school_id = 1 where student_id = 1;
update students set school_id = 2 where student_id = 2;
```

Скриншот изменённой базы данных в СУБД