Министерство науки и высшего образования
Российской Федерации Федеральное
государственное автономное образовательное
учреждение высшего образования
«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
УНИВЕРСИТЕТ ИТМО»
Факультет инфокоммуникационных технологий

ОТЧЕТ
ПО ЛАБОРАТОРНОЙ РАБОТЕ № 1.2
по теме: Создание таблиц базы данных postgresql. Заполнениетаблиц рабочими данными.
по дисциплине: Проектирование и реализация баз данных

Специальность:
09.03.03 Мобильные и сетевые технологии

Выполнил:
студент 2 курса ИКТ группа К3241
Сизей Омар

Проверила:
Говорова Марина Михайловна

Санкт-Петербург 2022

ЦЕЛЬ РАБОТЫ

Овладеть практическими навыками создания таблиц базы данных PostgreSQL 1X, заполнения их рабочими данными, резервного копирования и восстановления БД.

ПРАБТИЧЕСROE ЗАДАНИЕ

Оборудование: компьютерный класс.
Программное обеспечение: СУБД PostgreSQL 1X, pgAdmin 4.
Практическое задание:

1. Создать базу данных с использованием pgAdmin 4 (согласно индивидуальному заданию).
2. Создать схему в составе базы данных.
3. Создать таблицы базы данных.
4. Установить ограничения на данные: Primary Key, Unique, Check, Foreign Key.
5. Заполнить таблицы БД рабочими данными.
6. Создать резервную копию БД.

Указание:
Создать две резервные копии:

- с расширением CUSTOM для восстановления БД;
- с расширением PLAIN для листинга (в отчете);
- при создании резервных копий БД настроить параметры Dump options для Type of objects и Queries.

1. Восстановить БД.

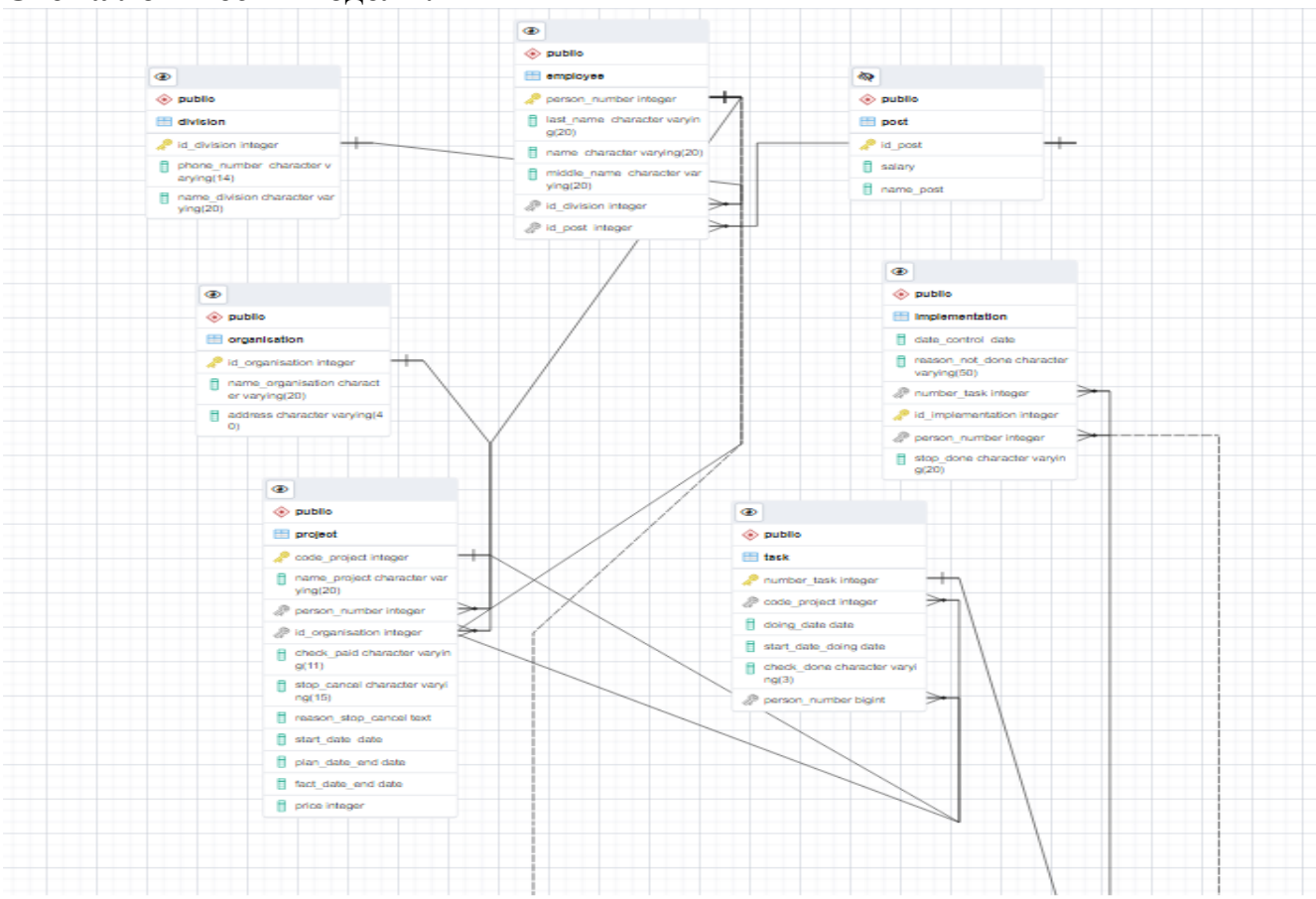
Вариант 4. БД «Учет выполнения заданий»
Описание предметной области: Сотрудники организации выполняют проекты. Проекты состоят из нескольких заданий. Каждый проект имеет руководителя проекта из числа сотрудников. Каждый сотрудник может участвовать в одном или нескольких проектах, или временно не участвовать ни в каких проектах. Над каждым проектом может работать несколько сотрудников отделов, или временно проект может быть приостановлен, тогда над ним не работает ни один сотрудник. Над каждым заданием (этапом) в проекте может работать несколько сотрудников сотрудник. Каждый сотрудник числится в одном отделе.
БД должна содержать следующий минимальный набор сведений: Номер сотрудника. Фамилия сотрудника. Имя сотрудника. Отчество сотрудника. Должность сотрудника. Оклад сотрудника. Название организации-заказчика. Номер организации. Адрес организации. Номер телефона отдела. Номер отдела. Название отдела. Код проекта. Название проекта. Сроки выполнения проекта. Руководитель проекта. Номер задания. Дата начала выполнения задания. Срок выполнения задания. Отметка о выполнении задания. Отметка

о выполнении задания каждым сотрудником. Дата контроля выполнения задания. Причина невыполнения задания.

ХОД РАБОТЫ

1) Наименование БД: ProjectDB

2) **Схема логическй модели:**



3) **Dump, содержащий скрипты работы с БД**.

```
-- PostgreSQL database dump
--

-- Dumped from database version 14.2
-- Dumped by pg_dump version 14.2

SET statement_timeout = 0;
SET lock_timeout = 0;
SET idle_in_transaction_session_timeout = 0;
SET client_encoding = 'UTF8';
SET standard_conforming_strings = on;
SELECT pg_catalog.set_config('search_path', '', false);
SET check_function_bodies = false;
SET xmloption = content;
SET client_min_messages = warning;
SET row_security = off;

DROP DATABASE "ProjectDB";
--
-- Name: ProjectDB; Type: DATABASE; Schema: -;
Owner: postgres
--

CREATE DATABASE "ProjectDB" WITH
TEMPLATE = template0 ENCODING = 'UTF8'
LOCALE = 'English_United States.1251';


ALTER DATABASE "ProjectDB" OWNER TO
postgres;

\connect "ProjectDB"

SET statement_timeout = 0;
SET lock_timeout = 0;
SET idle_in_transaction_session_timeout = 0;
SET client_encoding = 'UTF8';
SET standard_conforming_strings = on;
SELECT pg_catalog.set_config('search_path', '', false);
SET check_function_bodies = false;
SET xmloption = content;
SET client_min_messages = warning;
SET row_security = off;

SET default_tablespace = '';

SET default_table_access_method = heap;
```

```
--
-- Name: division; Type: TABLE; Schema: public;
Owner: postgresCREATE TABLE IF NOT EXISTS
public.division
(
        id_division integer NOT NULL,
         "phone_number " character varying(14)
        COLLATE pg_catalog."default" NOT NULL,
        name_division character varying(20) COLLATE
        pg_catalog."default" NOT NULL,
       CONSTRAINT division_pkey PRIMARY KEY
       (id_division)
);



ALTER TABLE public." division " OWNER TO postgres;
```

```
CREATE TABLE IF NOT EXISTS public.employee
(
    person_number integer NOT NULL,
    "last_name " character varying(20) COLLATE pg_catalog."default" NOT
NULL,
    "name " character varying(20) COLLATE pg_catalog."default" NOT NULL,
    "middle_name " character varying(20) COLLATE pg_catalog."default",
    id_division integer NOT NULL,
    "id_post " integer NOT NULL,
    CONSTRAINT employee_pkey PRIMARY KEY (person_number)
);

ALTER TABLE public." employee " OWNER TO postgres;



CREATE TABLE IF NOT EXISTS public.implementation
(
    "date_control " date NOT NULL,
    reason_not_done character varying(50) COLLATE pg_catalog."default",
    number_task integer NOT NULL,
    id_implementation integer NOT NULL,
    person_number integer NOT NULL,
    stop_done character varying(20) COLLATE pg_catalog."default",
    CONSTRAINT implementation_pkey PRIMARY KEY (id_implementation)
);
ALTER TABLE public." implementation " OWNER TO
        postgres;



CREATE TABLE IF NOT EXISTS public.organisation
(
    id_organisation integer NOT NULL,
    name_organisation character varying(20) COLLATE pg_catalog."default"
    NOT NULL,
    address character varying(40) COLLATE pg_catalog."default" NOT NULL,
    CONSTRAINT organisation_pkey PRIMARY KEY (id_organisation)
);

ALTER TABLE public." organisation " OWNER TO
        postgres;

CREATE TABLE IF NOT EXISTS public.post
(
    id_post integer NOT NULL,
    salary integer NOT NULL,
```

```
    name_post character varying(20) COLLATE pg_catalog."default",
    CONSTRAINT position_pkey PRIMARY KEY (id_post)
);


ALTER TABLE public." post " OWNER TO postgres;


CREATE TABLE IF NOT EXISTS public.project
(
    code_project integer NOT NULL,
    name_project character varying(20) COLLATE pg_catalog."default" NOT
    NULL,
    person_number integer,
    id_organisation integer,
    check_paid character varying(11) COLLATE pg_catalog."default",
    stop_cancel character varying(15) COLLATE pg_catalog."default",
    reason_stop_cancel text COLLATE pg_catalog."default",
    "start_date " date NOT NULL,
    plan_date_end date NOT NULL,
    fact_date_end date,
     price integer NOT NULL,
    CONSTRAINT project_pkey PRIMARY KEY (code_project)
);



ALTER TABLE public." project " OWNER TO postgres;


CREATE TABLE IF NOT EXISTS public.task
(
    number_task integer NOT NULL,
     code_project integer NOT NULL,
    doing_date date NOT NULL,
    start_date_doing date NOT NULL,
    check_done character varying(3) COLLATE pg_catalog."default" NOT
    NULL,
    person_number bigint NOT NULL,
    CONSTRAINT task_pkey PRIMARY KEY (number_task)
);



ALTER TABLE public." task " OWNER TO postgres;
```

--
-- Data for Name: division; Type: TABLE DATA; Schema: public; Owner: postgres
--

COPY public.division (id_division, "phone_number ", name_division) FROM stdin;
\.
COPY public.division (id_division, "phone_number ", name_division) FROM
'$$PATH$$/3349.dat';

--
-- Data for Name: employee; Type: TABLE DATA; Schema: public; Owner: postgres
--

COPY public.employee (person_number, "last_name ", "name ", "middle_name ",
id_division, "id_post ") FROM stdin;
\.
COPY public.employee (person_number, "last_name ", "name ", "middle_name ",
id_division, "id_post ") FROM '$$PATH$$/3348.dat';

--
-- Data for Name: implementation; Type: TABLE DATA; Schema: public; Owner: postgres
--

COPY public.implementation ("date_control ", reason_not_done, number_task,
id_implementation, person_number, stop_done) FROM stdin;
\.
COPY public.implementation ("date_control ", reason_not_done, number_task,
id_implementation, person_number, stop_done) FROM '$$PATH$$/3350.dat';

--
-- Data for Name: organisation; Type: TABLE DATA; Schema: public; Owner: postgres
--

COPY public.organisation (id_organisation, name_organisation, address) FROM stdin;
\.
COPY public.organisation (id_organisation, name_organisation, address) FROM
'$$PATH$$/3351.dat';

--
-- Data for Name: post; Type: TABLE DATA; Schema: public; Owner: postgres
--

COPY public.post (id_post, salary, name_post) FROM stdin;
\.
COPY public.post (id_post, salary, name_post) FROM '$$PATH$$/3352.dat';

--
-- Data for Name: project; Type: TABLE DATA; Schema: public; Owner: postgres
--

COPY public.project (code_project, name_project, person_number, id_organisation,
check_paid, stop_cancel, reason_stop_cancel, "start_date ", plan_date_end, fact_date_end,
price) FROM stdin;

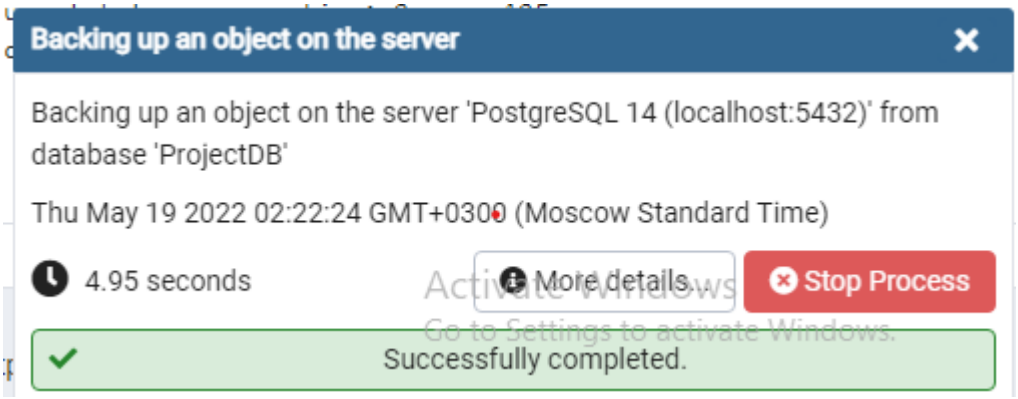
```
\.
COPY public.project (code_project, name_project, person_number, id_organisation,
check_paid, stop_cancel, reason_stop_cancel, "start_date ", plan_date_end, fact_date_end,
price) FROM '$$PATH$$/3353.dat';

--
-- Data for Name: task; Type: TABLE DATA; Schema: public; Owner: postgres
--

COPY public.task (number_task, code_project, doing_date, start_date_doing, check_done,
person_number) FROM stdin;
\.
COPY public.task (number_task, code_project, doing_date, start_date_doing, check_done,
person_number) FROM '$$PATH$$/3354.dat';

--
-- Name: division division_pkey; Type: CONSTRAINT; Schema: public; Owner: postgres
--

ALTER TABLE ONLY public.division
    ADD CONSTRAINT division_pkey PRIMARY KEY (id_division);


--
-- Name: employee employee_pkey; Type: CONSTRAINT; Schema: public; Owner:
postgres
--

ALTER TABLE ONLY public.employee
    ADD CONSTRAINT employee_pkey PRIMARY KEY (person_number);


--
-- Name: implementation implementation_pkey; Type: CONSTRAINT; Schema: public;
Owner: postgres
--

ALTER TABLE ONLY public.implementation
    ADD CONSTRAINT implementation_pkey PRIMARY KEY (id_implementation);


--
-- Name: organisation organisation_pkey; Type: CONSTRAINT; Schema: public; Owner:
postgres
--

ALTER TABLE ONLY public.organisation
    ADD CONSTRAINT organisation_pkey PRIMARY KEY (id_organisation);


--
-- Name: post position_pkey; Type: CONSTRAINT; Schema: public; Owner: postgres
--
```

```
ALTER TABLE ONLY public.post
    ADD CONSTRAINT position_pkey PRIMARY KEY (id_post);


--
-- Name: project project_pkey; Type: CONSTRAINT; Schema: public; Owner: postgres
--

ALTER TABLE ONLY public.project
    ADD CONSTRAINT project_pkey PRIMARY KEY (code_project);


--
-- Name: task task_pkey; Type: CONSTRAINT; Schema: public; Owner: postgres
--

ALTER TABLE ONLY public.task
    ADD CONSTRAINT task_pkey PRIMARY KEY (number_task);


--
-- Name: employee ER1; Type: FK CONSTRAINT; Schema: public; Owner: postgres
--

ALTER TABLE ONLY public.employee
    ADD CONSTRAINT "ER1" FOREIGN KEY (id_division) REFERENCES
public.division(id_division) NOT VALID;


--
-- Name: employee ER2; Type: FK CONSTRAINT; Schema: public; Owner: postgres
--

ALTER TABLE ONLY public.employee
    ADD CONSTRAINT "ER2" FOREIGN KEY ("id_post ") REFERENCES
public.post(id_post) NOT VALID;


--
-- Name: project ER3; Type: FK CONSTRAINT; Schema: public; Owner: postgres
--

ALTER TABLE ONLY public.project
    ADD CONSTRAINT "ER3" FOREIGN KEY (person_number) REFERENCES
public.employee(person_number) NOT VALID;


--
-- Name: project ER4; Type: FK CONSTRAINT; Schema: public; Owner: postgres
--

ALTER TABLE ONLY public.project
```

```
    ADD CONSTRAINT "ER4" FOREIGN KEY (id_organisation) REFERENCES
public.organisation(id_organisation) NOT VALID;


--
-- Name: task ER5; Type: FK CONSTRAINT; Schema: public; Owner: postgres
--

ALTER TABLE ONLY public.task
    ADD CONSTRAINT "ER5" FOREIGN KEY (code_project) REFERENCES
public.project(code_project) NOT VALID;


--
-- Name: task ER6; Type: FK CONSTRAINT; Schema: public; Owner: postgres
--

ALTER TABLE ONLY public.task
    ADD CONSTRAINT "ER6" FOREIGN KEY (person_number) REFERENCES
public.employee(person_number) NOT VALID;


--
-- Name: implementation ER7; Type: FK CONSTRAINT; Schema: public; Owner: postgres
--

ALTER TABLE ONLY public.implementation
    ADD CONSTRAINT "ER7" FOREIGN KEY (number_task) REFERENCES
public.task(number_task) NOT VALID;


--
-- Name: implementation ER8; Type: FK CONSTRAINT; Schema: public; Owner: postgres
--

ALTER TABLE ONLY public.implementation
    ADD CONSTRAINT "ER8" FOREIGN KEY (person_number) REFERENCES
public.employee(person_number) NOT VALID;


--
-- PostgreSQL database dump complete
--
```

4) Резервное копирование базы даннных

5) Восстановление базы данных



Выаод:

В ходе работы была создана база данных в PostgreSQL, таблицы и ограничения на значение столбцов, введены рабочие данные, сделана резервная копия и восстановлена база данных. Программа pgAdmin позволяет создавать базы данных непосредственно через взаимодействие с ее графическим интерфейсом или через работу со встроенным генератором диаграмм ER.