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. Заполнение таблиц рабочими данными.
по дисциплине: Проектирование и реализация баз данных

Специальность:
09.03.03 Мобильные и сетевые технологии

Проверил:
Говорова М.М. _____
Дата: «__» _____ 20__г.
Оценка _____

Выполнил:
студент группы К3241
Коник А.А.

Санкт-Петербург
2022

ЦЕЛЬ РАБОТЫ

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

ПРАКТИЧЕСКОЕ 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.*

7. 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

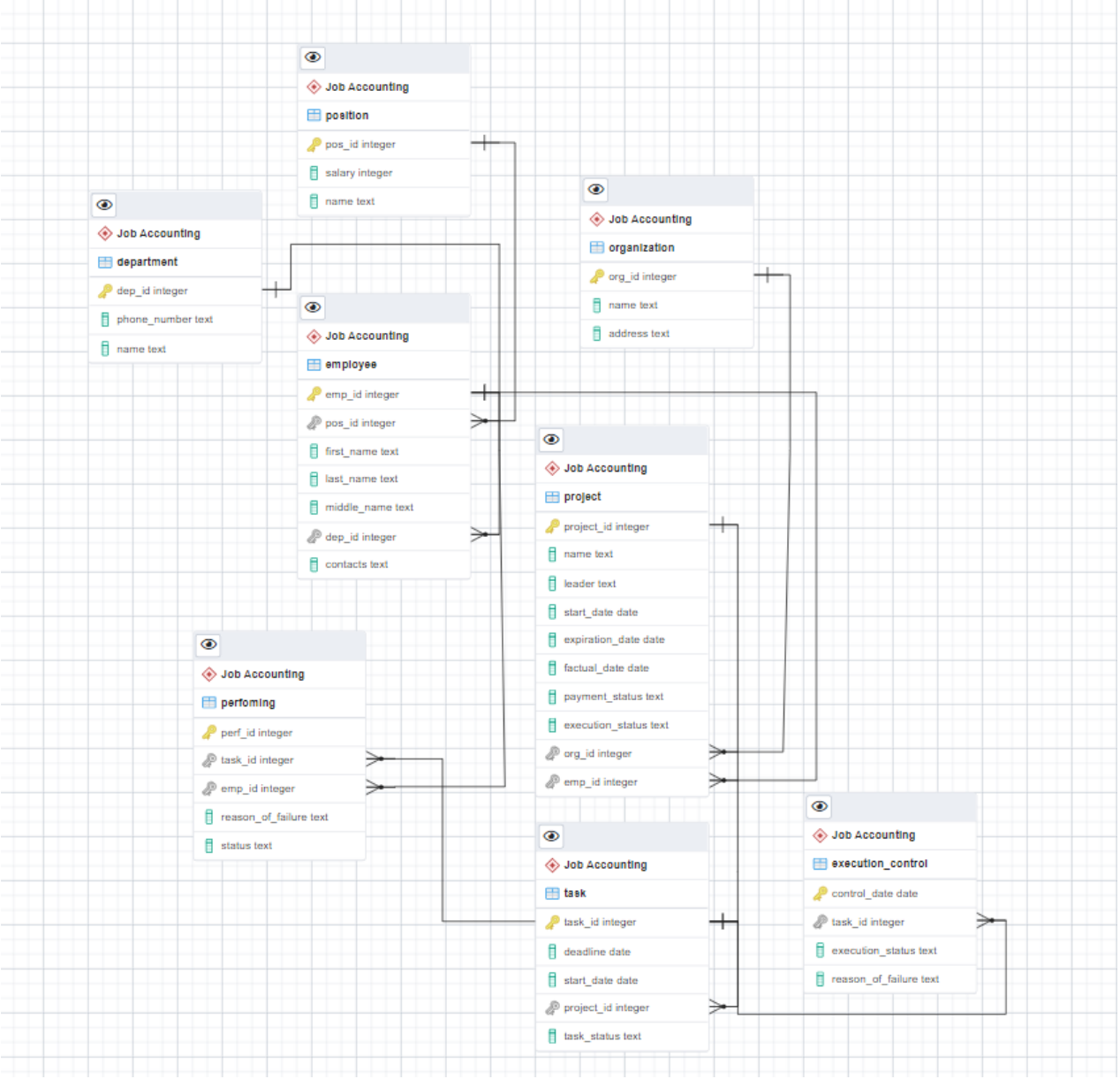
задания. Срок выполнения задания. Отметка о выполнении задания. Отметка о выполнении задания каждым сотрудником. Дата контроля выполнения задания. Причина невыполнения задания.

ХОД РАБОТЫ

1) **Наименование БД:**

Job accounting

2) **Схема логической модели:**



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

Создание базы данных:

```
-- Database: Job Accounting
-- DROP DATABASE IF EXISTS "Job Accounting";
CREATE DATABASE "Job Accounting"
```

```
    WITH
    OWNER = postgres
    ENCODING = 'UTF8'
    LC_COLLATE = 'Russian_Russia.1251'
    LC_CTYPE = 'Russian_Russia.1251'
    TABLESPACE = pg_default
    CONNECTION LIMIT = -1;
```

Создание схемы:

```
CREATE SCHEMA "Job Accounting";
ALTER SCHEMA "Job Accounting" OWNER TO postgres;
SET default_tablespace = '';
SET default_table_access_method = heap;
```

Создание таблицы project и определение ограничений:

```
CREATE TABLE "Job Accounting".project (
    project_id integer NOT NULL,
    name text NOT NULL,
    leader text NOT NULL,
    start_date date NOT NULL,
    expiration_date date NOT NULL,
    factual_date date,
    payment_status text NOT NULL,
    execution_status text NOT NULL,
    org_id integer NOT NULL,
    emp_id integer NOT NULL
);
ALTER TABLE "Job Accounting".project OWNER TO postgres;

ALTER TABLE ONLY "Job Accounting".project
    ADD CONSTRAINT "Project_pkey" PRIMARY KEY (project_id);

ALTER TABLE "Job Accounting".project
    ADD CONSTRAINT chk_end_date CHECK ((start_date < expiration_date)) NOT
     VALID;

ALTER TABLE "Job Accounting".project
    ADD CONSTRAINT chk_execution_status CHECK ((execution_status = ANY
     (ARRAY['In work'::text, 'Finished'::text, 'Suspended'::text, 'Canceled'::text, 'in
     work'::text, 'finished'::text, 'suspended'::text, 'canceled'::text]))) NOT VALID;
```

```
ALTER TABLE "Job Accounting".project
    ADD CONSTRAINT chk_factual_date CHECK ((start_date < factual_date)) NOT
     VALID;

ALTER TABLE "Job Accounting".project
    ADD CONSTRAINT chk_payment_status CHECK ((payment_status = ANY
     (ARRAY['Paid'::text, 'Processing'::text, 'Unpaid'::text, 'paid'::text, 'paid'::text,
     'unpaid'::text]))) NOT VALID;

ALTER TABLE "Job Accounting".project
    ADD CONSTRAINT chk_project_id CHECK ((project_id > 0)) NOT VALID;

ALTER TABLE ONLY "Job Accounting".project
    ADD CONSTRAINT project_emp_id_fkey FOREIGN KEY (emp_id)
     REFERENCES "Job Accounting".employee(emp_id) NOT VALID;

ALTER TABLE ONLY "Job Accounting".project
    ADD CONSTRAINT project_org_id_fkey FOREIGN KEY (org_id)
     REFERENCES "Job Accounting".organization(org_id) NOT VALID;
```

Создание таблицы employee и определение ограничений:

```
CREATE TABLE "Job Accounting".employee (
    emp_id integer NOT NULL,
    pos_id integer NOT NULL,
    first_name text NOT NULL,
    last_name text NOT NULL,
    middle_name text,
    dep_id integer NOT NULL,
    contacts text NOT NULL
);
ALTER TABLE "Job Accounting".employee OWNER TO postgres;

ALTER TABLE "Job Accounting".employee
    ADD CONSTRAINT chk_emp_id CHECK ((emp_id > 0)) NOT VALID;

ALTER TABLE ONLY "Job Accounting".employee
    ADD CONSTRAINT employee_pkey PRIMARY KEY (emp_id);

ALTER TABLE ONLY "Job Accounting".employee
```

```
    ADD CONSTRAINT employee_dep_id_fkey FOREIGN KEY (dep_id)
     REFERENCES "Job Accounting".department(dep_id) NOT VALID;

ALTER TABLE ONLY "Job Accounting".employee
    ADD CONSTRAINT employee_pos_id_fkey FOREIGN KEY (pos_id)
     REFERENCES "Job Accounting"."position"(pos_id) NOT VALID;
```

Создание таблицы department и определение ограничений:

```
CREATE TABLE "Job Accounting".department (
    dep_id integer NOT NULL,
    phone_number text NOT NULL,
    name text NOT NULL
);
ALTER TABLE "Job Accounting".department OWNER TO postgres;

ALTER TABLE "Job Accounting".department
    ADD CONSTRAINT chk_dep_id CHECK ((dep_id > 0)) NOT VALID;

ALTER TABLE ONLY "Job Accounting".department
    ADD CONSTRAINT department_pkey PRIMARY KEY (dep_id);
```

Создание таблицы organization и определение ограничений:

```
CREATE TABLE "Job Accounting".organization (
    org_id integer NOT NULL,
    name text NOT NULL,
    address text NOT NULL
);
ALTER TABLE "Job Accounting".organization OWNER TO postgres;

ALTER TABLE "Job Accounting".organization
    ADD CONSTRAINT chk_org_id CHECK ((org_id > 0)) NOT VALID;

ALTER TABLE ONLY "Job Accounting".organization
    ADD CONSTRAINT organization_pkey PRIMARY KEY (org_id);
```

Создание таблицы task и определение ограничений:

```
CREATE TABLE "Job Accounting".task (
    task_id integer NOT NULL,
    deadline date NOT NULL,
```

```
    start_date date NOT NULL,
    project_id integer NOT NULL,
    task_status text NOT NULL
);
ALTER TABLE "Job Accounting".task OWNER TO postgres;

ALTER TABLE "Job Accounting".task
    ADD CONSTRAINT chk_deadline CHECK ((start_date < deadline)) NOT
     VALID;

ALTER TABLE "Job Accounting".task
    ADD CONSTRAINT chk_task_id CHECK ((task_id > 0)) NOT VALID;

ALTER TABLE "Job Accounting".task
    ADD CONSTRAINT chk_task_status CHECK ((task_status = ANY (ARRAY['In
     work'::text, 'Finished'::text, 'Suspended'::text, 'Canceled'::text, 'in work'::text,
     'finished'::text, 'suspended'::text, 'canceled'::text]))) NOT VALID;

ALTER TABLE ONLY "Job Accounting".task
    ADD CONSTRAINT task_pkey PRIMARY KEY (task_id);

ALTER TABLE ONLY "Job Accounting".task
    ADD CONSTRAINT task_project_id_fkey FOREIGN KEY (project_id)
     REFERENCES "Job Accounting".project(project_id) NOT VALID;
```

Создание таблицы performing и определение ограничений:

```
CREATE TABLE "Job Accounting".perfoming (
    perf_id integer NOT NULL,
    task_id integer NOT NULL,
    emp_id integer NOT NULL,
    reason_of_failure text,
    status text NOT NULL
);
ALTER TABLE "Job Accounting".perfoming OWNER TO postgres;

ALTER TABLE "Job Accounting".perfoming
    ADD CONSTRAINT chk_status CHECK ((status = ANY (ARRAY['In work'::text,
     'Finished'::text, 'Suspended'::text, 'Canceled'::text, 'in work'::text, 'finished'::text,
     'suspended'::text, 'canceled'::text]))) NOT VALID;

ALTER TABLE ONLY "Job Accounting".perfoming
```

```
    ADD CONSTRAINT chk_unique_emp_id UNIQUE (emp_id, task_id);

ALTER TABLE "Job Accounting".perfoming
    ADD CONSTRAINT perf_id CHECK ((perf_id > 0)) NOT VALID;

ALTER TABLE ONLY "Job Accounting".perfoming
    ADD CONSTRAINT perfoming_pkey PRIMARY KEY (perf_id);

ALTER TABLE ONLY "Job Accounting".perfoming
    ADD CONSTRAINT perfoming_emp_id_fkey FOREIGN KEY (emp_id)
     REFERENCES "Job Accounting".employee(emp_id) NOT VALID;

ALTER TABLE ONLY "Job Accounting".perfoming
    ADD CONSTRAINT perfoming_task_id_fkey FOREIGN KEY (task_id)
     REFERENCES "Job Accounting".task(task_id) NOT VALID;
```

Создание таблицы execution control и определение ограничений:

```
CREATE TABLE "Job Accounting".execution_control (
    control_date date NOT NULL,
    task_id integer NOT NULL,
    execution_status text NOT NULL,
    reason_of_failure text
);
ALTER TABLE "Job Accounting".execution_control OWNER TO postgres;

ALTER TABLE ONLY "Job Accounting".execution_control
    ADD CONSTRAINT chk_unique_task_id UNIQUE (control_date, task_id);

ALTER TABLE ONLY "Job Accounting".execution_control
    ADD CONSTRAINT execution_control_pkey PRIMARY KEY (control_date);

ALTER TABLE ONLY "Job Accounting".execution_control
    ADD CONSTRAINT execution_control_task_id_fkey FOREIGN KEY (task_id)
     REFERENCES "Job Accounting".task(task_id) NOT VALID;
```

Создание таблицы position и определение ограничений:

```
CREATE TABLE "Job Accounting"."position" (
    pos_id integer NOT NULL,
    salary integer NOT NULL,
    name text NOT NULL
```

```
);
ALTER TABLE "Job Accounting"."position" OWNER TO postgres;

ALTER TABLE "Job Accounting"."position"
    ADD CONSTRAINT chk_pos_id CHECK ((pos_id > 0)) NOT VALID;

ALTER TABLE "Job Accounting"."position"
    ADD CONSTRAINT chk_salary CHECK ((salary > 0)) NOT VALID;

ALTER TABLE ONLY "Job Accounting"."position"
    ADD CONSTRAINT position_pkey PRIMARY KEY (pos_id);
```

Заполнение таблицы project рабочими данными:

```
COPY "Job Accounting".project (project_id, name, leader, start_date, expiration_date,
    factual_date, payment_status, execution_status, org_id, emp_id) FROM stdin;
111     App "North"     Oleg Ivanov     2021-10-24      2022-03-24      \N      Unpaid  In work 11      1
222     Delivery service        Nikolas Luff    2021-06-12      2021-10-14      2021-10-12      Paid    Finished        22      2
333     Web-site "Magnit"       Svetlana Korsakova      2022-01-20      2022-05-27      \N      Processing      In work 33      3
```

Заполнение таблицы employee рабочими данными:

```
COPY "Job Accounting".employee (emp_id, pos_id, first_name, last_name,
    middle_name, dep_id, contacts) FROM stdin;
1   1       Anastasia   Vorobieva   Alexandrovna    1   sbssu@mail.ru
2   2       Olga        Minaeva     Igorevna        2   hjkaw@gmail.com
3   3       Alex        Kross       \N              3   dguws@ya.ru
```

Заполнение таблицы department рабочими данными:

```
COPY "Job Accounting".department (dep_id, phone_number, name) FROM stdin;
1   +79198341423    Finance
2   +79274137832    Sales
3   +79215673252    General Managament
```

Заполнение таблицы organization рабочими данными:

```
COPY "Job Accounting".organization (org_id, name, address) FROM stdin;
11 Diasoft      Russia, Moscow, st. Regiment 3, 127018
22 INITI        Russia, Moscow, Party lane 1
33 Yandex       Russia, St. Petersburg, Piskarevsky prospect 2, 195067
```

Заполнение таблицы task рабочими данными:

COPY "Job Accounting".task (task_id, deadline, start_date, project_id, task_status)
FROM stdin;
1 2022-03-24 2022-02-24 111 In work
2 2021-09-24 2021-09-12 222 Finished
3 2022-01-24 2022-01-20 333 Canceled

Заполнение таблицы performing рабочими данными:

COPY "Job Accounting".perfoming (perf_id, task_id, emp_id, reason_of_failure, status) FROM stdin;
1 1 1 \N In work
2 2 2 \N Finished
3 3 3 Unnecessarity Canceled

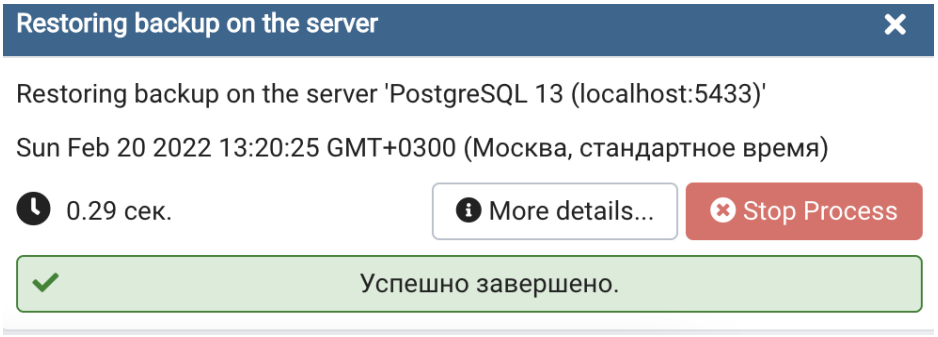
Заполнение таблицы execution control рабочими данными:

COPY "Job Accounting".execution_control (control_date, task_id, execution_status, reason_of_failure) FROM stdin;
2022-03-12 1 In work \N
2021-09-12 2 Finished \N
2022-01-24 3 Canceled Unnecessarity

Заполнение таблицы position рабочими данными:

COPY "Job Accounting"."position" (pos_id, salary, name) FROM stdin;
1 40000 Marketing Coordinator
2 75000 Manager
3 30000 Vendor

Восстановление базы данных:



Вывод:

В ходе выполнения работы была создана база данных в PostgreSQL; логическая схема в ее составе; созданы таблицы и заданы ограничения на

данные: Primary Key, Unique, Check, Foreign Key; в базу данных были внесены рабочие данные; также созданы две резервные копии и произведено восстановление базы данных.