ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»

Факультет инфокоммуникационных технологий

Дисциплина:
«Проектирование и реализация баз данных»

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №1
«СОЗДАНИЕ БД POSTGRESQL В PGADMIN. РЕЗЕРВНОЕ КОПИРОВАНИЕ И ВОССТАНОВЛЕНИЕ БД»

Выполнил:
студент группы K32402
Пластун Елизавета Олеговна

_____
(подпись)

Проверил(а):
Говорова Марина Михайловна

_____
(отметка о выполнении)

_____
(подпись)

Санкт-Петербург
2023 г.

Цель работы 1.1: овладеть практическими навыками установки СУБД PostgreSQL и создания базы данных в pgadmin 4.

Практическое задание 1.1:

1. Установить СУБД PostgreSQL 1X.
2. Создать базу данных с использованием pgadmin 4.

Цель работы 1.2: овладеть практическими навыками создания таблиц базы данных PostgreSQL 1X, заполнения их рабочими данными, резервного копирования и восстановления БД.

Практическое задание 1.2:

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.

Выполнение

Наименование БД: lr1.2

ERD диаграмма:

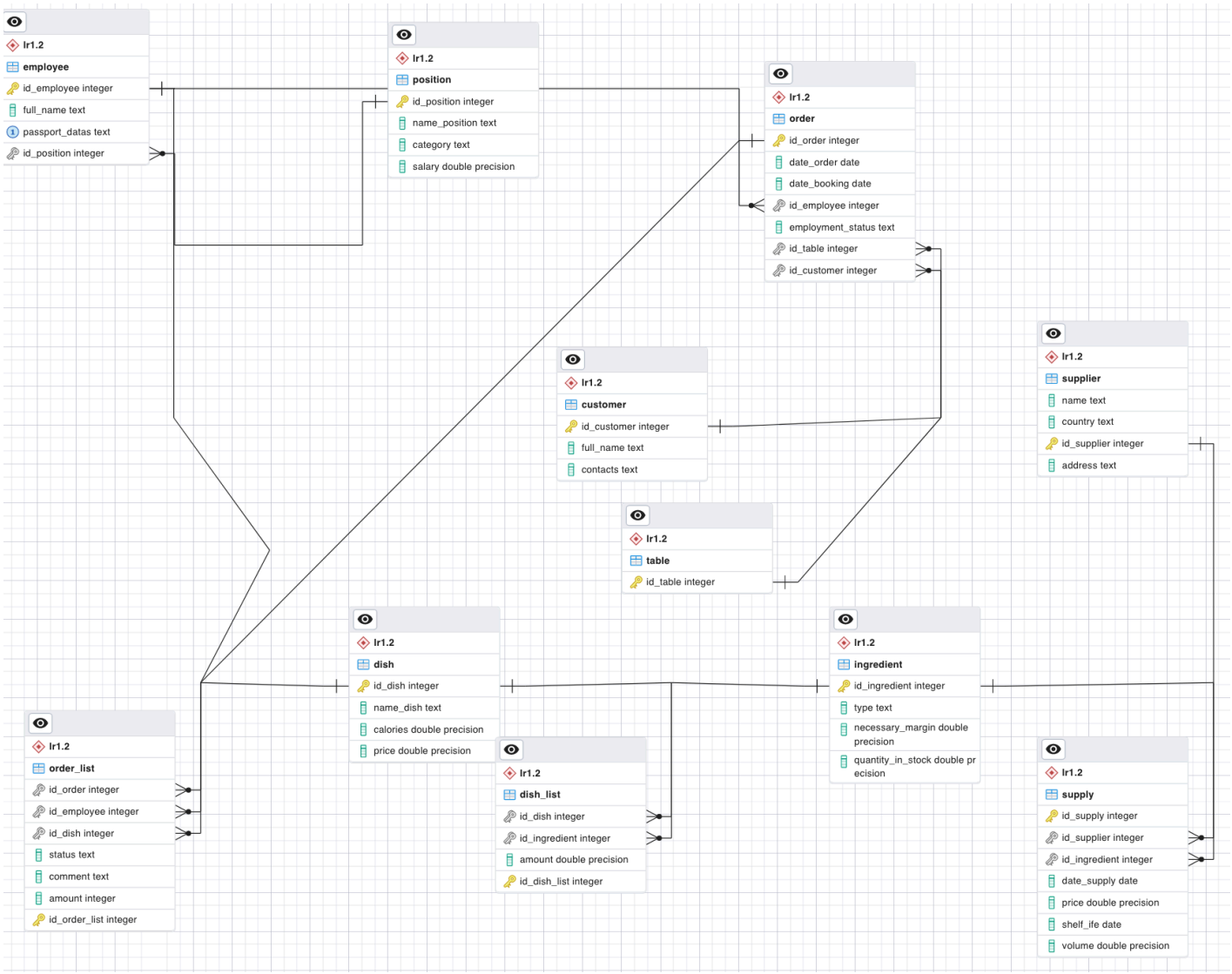

Рисунок 1 - ERD диаграмма

Dump БД: dump базы данных, сгенерированный в pgAdmin, приложен к отчету.
Файлы: lr111111

Скрипты работы с БД для создания таблиц:

```
-- Table: lr1.2.customer

-- DROP TABLE IF EXISTS "lr1.2".customer;

CREATE TABLE IF NOT EXISTS "lr1.2".customer
(
    id_customer integer NOT NULL,
    full_name text COLLATE pg_catalog."default" NOT NULL,
    contacts text COLLATE pg_catalog."default" NOT NULL,
    CONSTRAINT customer_pkey PRIMARY KEY (id_customer),
    CONSTRAINT id_customer CHECK (id_customer > 0) NOT VALID
)

TABLESPACE pg_default;

ALTER TABLE IF EXISTS "lr1.2".customer
    OWNER to postgres;


-- Table: lr1.2.dish

-- DROP TABLE IF EXISTS "lr1.2".dish;

CREATE TABLE IF NOT EXISTS "lr1.2".dish
(
    id_dish integer NOT NULL,
    name_dish text COLLATE pg_catalog."default" NOT NULL,
    calories double precision NOT NULL,
    price double precision NOT NULL,
    id_employee integer NOT NULL,
    CONSTRAINT dish_pkey PRIMARY KEY (id_dish),
    CONSTRAINT dish_id_employee_fkey FOREIGN KEY (id_employee)
        REFERENCES "lr1.2".employee (id_employee) MATCH SIMPLE
        ON UPDATE NO ACTION
        ON DELETE NO ACTION
        NOT VALID,
    CONSTRAINT id_dish CHECK (id_dish > 0) NOT VALID,
    CONSTRAINT calories CHECK (calories > 0::double precision) NOT VALID,
    CONSTRAINT price CHECK (price > 0::double precision) NOT VALID
)

TABLESPACE pg_default;

ALTER TABLE IF EXISTS "lr1.2".dish
    OWNER to postgres;

-- Table: lr1.2.dish_list

-- DROP TABLE IF EXISTS "lr1.2".dish_list;
```

```
CREATE TABLE IF NOT EXISTS "lr1.2".dish_list
(
    id_dish integer NOT NULL,
    id_ingredient integer NOT NULL,
    amount double precision NOT NULL,
    id_dish_list integer NOT NULL,
    CONSTRAINT dish_list_pkey PRIMARY KEY (id_dish_list),
    CONSTRAINT dish_list_id_dish_fkey FOREIGN KEY (id_dish)
        REFERENCES "lr1.2".dish (id_dish) MATCH SIMPLE
        ON UPDATE NO ACTION
        ON DELETE NO ACTION
        NOT VALID,
    CONSTRAINT dish_list_id_ingredient_fkey FOREIGN KEY (id_ingredient)
        REFERENCES "lr1.2".ingredient (id_ingredient) MATCH SIMPLE
        ON UPDATE NO ACTION
        ON DELETE NO ACTION
        NOT VALID,
    CONSTRAINT dish_list_amount_check CHECK (amount > 0::double precision) NOT VALID,
    CONSTRAINT dish_list_id_dish_list_check CHECK (id_dish_list > 0) NOT VALID
)

TABLESPACE pg_default;

ALTER TABLE IF EXISTS "lr1.2".dish_list
    OWNER to postgres;

-- Table: lr1.2.employee

-- DROP TABLE IF EXISTS "lr1.2".employee;

CREATE TABLE IF NOT EXISTS "lr1.2".employee
(
    id_employee integer NOT NULL,
    full_name text COLLATE pg_catalog."default" NOT NULL,
    passport_datas text COLLATE pg_catalog."default" NOT NULL,
    id_position integer NOT NULL,
    CONSTRAINT employee_pkey PRIMARY KEY (id_employee),
    CONSTRAINT employee_passport_datas_passport_datas1_key UNIQUE (passport_datas)
        INCLUDE(passport_datas),
    CONSTRAINT "position" FOREIGN KEY (id_position)
        REFERENCES "lr1.2"."position" (id_position) MATCH SIMPLE
        ON UPDATE NO ACTION
        ON DELETE NO ACTION
        NOT VALID,
    CONSTRAINT id_employee CHECK (id_employee > 0) NOT VALID,
    CONSTRAINT id_position CHECK (id_position > 0) NOT VALID
)

TABLESPACE pg_default;

ALTER TABLE IF EXISTS "lr1.2".employee
    OWNER to postgres;
```

```
-- Trigger: t_employee

-- DROP TRIGGER IF EXISTS t_employee ON "lr1.2".employee;

CREATE TRIGGER t_employee
    AFTER INSERT OR DELETE OR UPDATE
    ON "lr1.2".employee
    FOR EACH ROW
    EXECUTE FUNCTION public.do_log();

-- Table: lr1.2.ingredient

-- DROP TABLE IF EXISTS "lr1.2".ingredient;

CREATE TABLE IF NOT EXISTS "lr1.2".ingredient
(
    id_ingredient integer NOT NULL,
    type_ingredient text COLLATE pg_catalog."default" NOT NULL,
    necessary_margin double precision NOT NULL,
    quantity_in_stock double precision NOT NULL,
    price_100_gramm integer,
    CONSTRAINT ingredient_pkey PRIMARY KEY (id_ingredient),
    CONSTRAINT ingredient_id_ingredient_check CHECK (id_ingredient > 0) NOT VALID,
    CONSTRAINT ingredient_necessary_margin_check CHECK (necessary_margin > 0::double
precision) NOT VALID,
    CONSTRAINT ingredient_quantity_in_stock_check CHECK (quantity_in_stock > 0::double precision)
NOT VALID
)

TABLESPACE pg_default;

ALTER TABLE IF EXISTS "lr1.2".ingredient
    OWNER to postgres;


-- Table: lr1.2.order

-- DROP TABLE IF EXISTS "lr1.2"."order";

CREATE TABLE IF NOT EXISTS "lr1.2"."order"
(
    id_order integer NOT NULL,
    date_order date NOT NULL,
    date_booking date,
    id_employee integer NOT NULL,
    employment_status text COLLATE pg_catalog."default" NOT NULL,
    id_table integer NOT NULL,
    id_customer integer NOT NULL,
    price double precision NOT NULL,
    CONSTRAINT order_pkey PRIMARY KEY (id_order),
    CONSTRAINT customer FOREIGN KEY (id_customer)
        REFERENCES "lr1.2".customer (id_customer) MATCH SIMPLE
        ON UPDATE NO ACTION
```

```
        ON DELETE NO ACTION
        NOT VALID,
    CONSTRAINT employee FOREIGN KEY (id_employee)
        REFERENCES "lr1.2".employee (id_employee) MATCH SIMPLE
        ON UPDATE NO ACTION
        ON DELETE NO ACTION
        NOT VALID,
    CONSTRAINT "table" FOREIGN KEY (id_table)
        REFERENCES "lr1.2"."table" (id_table) MATCH SIMPLE
        ON UPDATE NO ACTION
        ON DELETE NO ACTION
        NOT VALID,
    CONSTRAINT order_id_order_check CHECK (id_order > 0) NOT VALID
)

TABLESPACE pg_default;

ALTER TABLE IF EXISTS "lr1.2"."order"
    OWNER to postgres;

-- Table: lr1.2.order_list

-- DROP TABLE IF EXISTS "lr1.2".order_list;

CREATE TABLE IF NOT EXISTS "lr1.2".order_list
(
    id_order integer NOT NULL,
    id_employee integer NOT NULL,
    id_dish integer NOT NULL,
    status text COLLATE pg_catalog."default" NOT NULL,
    comment text COLLATE pg_catalog."default",
    amount integer NOT NULL,
    id_order_list integer NOT NULL,
    CONSTRAINT order_list_pkey PRIMARY KEY (id_order_list),
    CONSTRAINT dish FOREIGN KEY (id_dish)
        REFERENCES "lr1.2".dish (id_dish) MATCH SIMPLE
        ON UPDATE NO ACTION
        ON DELETE NO ACTION
        NOT VALID,
    CONSTRAINT employee FOREIGN KEY (id_employee)
        REFERENCES "lr1.2".employee (id_employee) MATCH SIMPLE
        ON UPDATE NO ACTION
        ON DELETE NO ACTION
        NOT VALID,
    CONSTRAINT "order" FOREIGN KEY (id_order)
        REFERENCES "lr1.2"."order" (id_order) MATCH SIMPLE
        ON UPDATE NO ACTION
        ON DELETE NO ACTION
        NOT VALID,
    CONSTRAINT order_list_id_order_list_check CHECK (id_order_list > 0) NOT VALID
)

TABLESPACE pg_default;
```

```
ALTER TABLE IF EXISTS "lr1.2".order_list
    OWNER to postgres;

-- Table: lr1.2.position

-- DROP TABLE IF EXISTS "lr1.2"."position";

CREATE TABLE IF NOT EXISTS "lr1.2"."position"
(
    id_position integer NOT NULL,
    name_position text COLLATE pg_catalog."default" NOT NULL,
    salary integer NOT NULL,
    category integer NOT NULL,
    CONSTRAINT position_pkey PRIMARY KEY (id_position),
    CONSTRAINT position_id_position_check CHECK (id_position > 0) NOT VALID
)

TABLESPACE pg_default;

ALTER TABLE IF EXISTS "lr1.2"."position"
    OWNER to postgres;

-- Table: lr1.2.supplier

-- DROP TABLE IF EXISTS "lr1.2".supplier;

CREATE TABLE IF NOT EXISTS "lr1.2".supplier
(
    name text COLLATE pg_catalog."default" NOT NULL,
    country text COLLATE pg_catalog."default" NOT NULL,
    id_supplier integer NOT NULL,
    address text COLLATE pg_catalog."default" NOT NULL,
    CONSTRAINT supplier_pkey PRIMARY KEY (id_supplier),
    CONSTRAINT supplier_id_supplier_check CHECK (id_supplier > 0) NOT VALID
)

TABLESPACE pg_default;

ALTER TABLE IF EXISTS "lr1.2".supplier
    OWNER to postgres;

-- Table: lr1.2.supply

-- DROP TABLE IF EXISTS "lr1.2".supply;

CREATE TABLE IF NOT EXISTS "lr1.2".supply
(
    id_supply integer NOT NULL,
    id_supplier integer NOT NULL,
    id_ingredient integer NOT NULL,
    date_supply date NOT NULL,
    price double precision NOT NULL,
```

```
    shelf_ife date NOT NULL,
    volume double precision NOT NULL,
    CONSTRAINT supply_pkey PRIMARY KEY (id_supply),
    CONSTRAINT ingredient FOREIGN KEY (id_ingredient)
        REFERENCES "lr1.2".ingredient (id_ingredient) MATCH SIMPLE
        ON UPDATE NO ACTION
        ON DELETE NO ACTION
        NOT VALID,
    CONSTRAINT supplier FOREIGN KEY (id_supplier)
        REFERENCES "lr1.2".supplier (id_supplier) MATCH SIMPLE
        ON UPDATE NO ACTION
        ON DELETE NO ACTION
        NOT VALID,
    CONSTRAINT supply_id_supply_check CHECK (id_supply > 0) NOT VALID,
    CONSTRAINT supply_price_check CHECK (price > 0::double precision) NOT VALID
)

TABLESPACE pg_default;

ALTER TABLE IF EXISTS "lr1.2".supply
    OWNER to postgres;

-- Table: lr1.2.table

-- DROP TABLE IF EXISTS "lr1.2"."table";

CREATE TABLE IF NOT EXISTS "lr1.2"."table"
(
    id_table integer NOT NULL,
    id_employee integer NOT NULL,
    CONSTRAINT table_pkey PRIMARY KEY (id_table),
    CONSTRAINT table_id_employee_fkey FOREIGN KEY (id_employee)
        REFERENCES "lr1.2".employee (id_employee) MATCH SIMPLE
        ON UPDATE CASCADE
        ON DELETE CASCADE
        NOT VALID,
    CONSTRAINT table_id_table_check CHECK (id_table > 0) NOT VALID
)

TABLESPACE pg_default;

ALTER TABLE IF EXISTS "lr1.2"."table"
    OWNER to postgres;
```

Выводы

В процессе выполнения лабораторной работы удалось более детально ознакомится с работой в pgAdmin 4, получить практические навыки создания таблиц, установки ограничений на таблицы, создания и восстановления резервных копий баз данных.