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

ОТЧЕТ
О ЛАБОРАТОРНОЙ РАБОТЕ № 3

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: «27» апреля 2021г.

Оценка _____

Выполнил:

студент
группы К3241

Шутов Д. Э.

Санкт-Петербург 2021 г.

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

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.

Индивидуальное задание:

Вариант №9 “Оптовая база”

ВЫПОЛНЕНИЕ

1. Название БД

«Wholesale base»

2. Схема инфологической модели данных БД

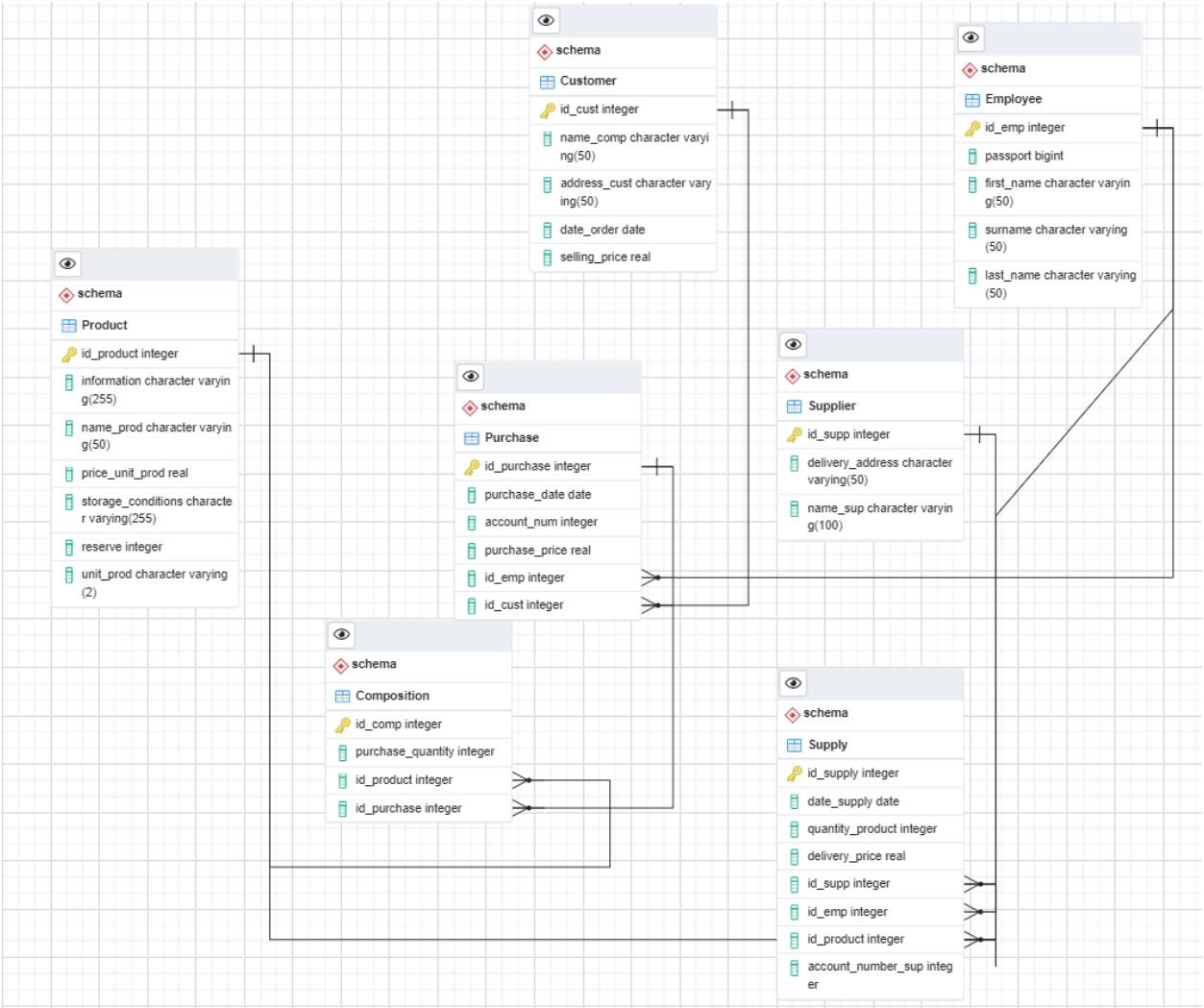

Рисунок 1 – Схема инфологической модели БД, сгенерированная в Generate ERD

3. Plain dump

```
-- Создание базы данных
CREATE DATABASE "Wholesale base"
    WITH
    OWNER = postgres
    ENCODING = 'UTF8'
    LC_COLLATE = 'Russian_Russia.1251'
```

```
    LC_CTYPE = 'Russian_Russia.1251'
    TABLESPACE = pg_default
    CONNECTION LIMIT = -1;

GRANT ALL ON DATABASE "Wholesale base" TO postgres;

GRANT TEMPORARY, CONNECT ON DATABASE "Wholesale base" TO
PUBLIC;

-- Создание схемы
CREATE SCHEMA schema
    AUTHORIZATION postgres;

-- Создание таблицы Employee и определение ограничений
CREATE TABLE schema."Employee"
(
    id_emp integer NOT NULL,
    passport bigint NOT NULL,
    first_name character varying (50) COLLATE pg_catalog."default" NOT NULL,
    surname character varying (50) COLLATE pg_catalog."default" NOT NULL,
    last_name character varying (50) COLLATE pg_catalog."default" NOT NULL,
    CONSTRAINT "Employee_pkey" PRIMARY KEY (id_emp),
    CONSTRAINT emp_chk CHECK (passport >= 1000000000 AND passport <=
'9999999999'::bigint)
)
ALTER TABLE schema."Employee"
    OWNER to postgres;
```

```
--Создание таблицы Composition и определение ограничений
CREATE TABLE schema."Composition"
(
    id_comp integer NOT NULL,
    purchase_quantity integer NOT NULL,
    id_product integer NOT NULL,
    id_purchase integer NOT NULL,
    CONSTRAINT "Composition_pkey" PRIMARY KEY (id_comp)
        WITH (FILLFACTOR=50),
    CONSTRAINT "CompositionFK" FOREIGN KEY (id_product)
        REFERENCES schema."Product" (id_product) MATCH SIMPLE
        ON UPDATE RESTRICT
        ON DELETE RESTRICT,
    CONSTRAINT "CompositionFK2" FOREIGN KEY (id_purchase)
        REFERENCES schema."Purchase" (id_purchase) MATCH SIMPLE
        ON UPDATE RESTRICT
        ON DELETE RESTRICT,
    CONSTRAINT comp_chk CHECK (id_comp >= 0),
    CONSTRAINT "Composition_purchase_quantity_check" CHECK
(purchase_quantity >= 0 AND purchase_quantity <= 99999)
)
ALTER TABLE schema."Composition"
    OWNER to postgres;

--Создание таблицы Customer и определение ограничений
CREATE TABLE schema."Customer"
(
    id_cust integer NOT NULL,
```

```
    name_comp character varying(50) COLLATE pg_catalog."default" NOT NULL,
    address_cust character varying(50) COLLATE pg_catalog."default" NOT NULL,
    date_order date NOT NULL,
    selling_price real NOT NULL,
    CONSTRAINT "Customer_pkey" PRIMARY KEY (id_cust),
    CONSTRAINT cust_chk CHECK (id_cust >= 0),
    CONSTRAINT "Customer_name_comp_check" CHECK (name_comp::text = ANY (ARRAY['ИЛИМ'::character varying, 'Титан'::character varying, 'УЛК'::character varying, 'АЦБК'::character varying]::text[]))
)
ALTER TABLE schema."Customer"
    OWNER to postgres;

--Создание таблицы Product и определение ограничений
CREATE TABLE schema."Product"
(
    id_product integer NOT NULL,
    information character varying(255) COLLATE pg_catalog."default" NOT NULL,
    name_prod character varying(50) COLLATE pg_catalog."default" NOT NULL,
    price_unit_prod real NOT NULL,
    storage_conditions character varying(255) COLLATE pg_catalog."default" NOT NULL,
    reserve integer NOT NULL,
    unit_prod character varying(2) COLLATE pg_catalog."default" NOT NULL,
    CONSTRAINT "Product_pkey" PRIMARY KEY (id_product),
    CONSTRAINT prod_chk CHECK (reserve >= 0),
    CONSTRAINT product_chk CHECK (id_product >= 0),
```

```
    CONSTRAINT "Product_name_prod_check" CHECK (name_prod::text = ANY (ARRAY['пластина'::character varying, 'четвертина'::character varying, 'горбыль'::character varying, 'обрезная доска'::character varying, 'необрезная доска'::character varying]::text[]))
)
ALTER TABLE schema."Product"
    OWNER to postgres;


--Создание таблицы Purchase и определение ограничений
CREATE TABLE schema."Purchase"
(
    id_purchase integer NOT NULL,
    purchase_date date NOT NULL,
    account_num integer NOT NULL,
    purchase_price real NOT NULL,
    id_emp integer NOT NULL,
    id_cust integer NOT NULL,
    CONSTRAINT "Purchase_pkey" PRIMARY KEY (id_purchase),
    CONSTRAINT "Purchase1_FK" FOREIGN KEY (id_cust)
        REFERENCES schema."Customer" (id_cust) MATCH SIMPLE
        ON UPDATE RESTRICT
        ON DELETE RESTRICT,
    CONSTRAINT "Purchase_FK" FOREIGN KEY (id_emp)
        REFERENCES schema."Employee" (id_emp) MATCH SIMPLE
        ON UPDATE RESTRICT
        ON DELETE RESTRICT,
    CONSTRAINT purch_chk CHECK (account_num >= 0 AND account_num <= 99999),
    CONSTRAINT "Purchase_id_purchase_check" CHECK (id_purchase >= 0)
)
```

```
ALTER TABLE schema."Purchase"
    OWNER to postgres;


--Создание таблицы Supplier и определение ограничений
CREATE TABLE schema."Supplier"
(
    id_supp integer NOT NULL,
    delivery_address character varying(50) COLLATE pg_catalog."default" NOT NULL,
    name_sup character varying(100) COLLATE pg_catalog."default" NOT NULL,
    CONSTRAINT "Supplier_pkey" PRIMARY KEY (id_supp),
    CONSTRAINT sup_chk CHECK (id_supp >= 0),
    CONSTRAINT "Supplier_name_sup_check" CHECK (name_sup::text = ANY (ARRAY['Плайком'::character varying, 'Аврора'::character varying, 'Проксима'::character varying]::text[]))
)
ALTER TABLE schema."Supplier"
    OWNER to postgres;


--Создание таблицы Supply и определение ограничений
CREATE TABLE schema."Supply"
(
    id_supply integer NOT NULL,
    date_supply date NOT NULL,
    quantity_product integer NOT NULL,
    delivery_price real NOT NULL,
    id_supp integer NOT NULL,
    id_emp integer NOT NULL,
    id_product integer NOT NULL,
    account_number_sup integer NOT NULL,
```

```
    CONSTRAINT "Supply_pkey" PRIMARY KEY (id_supply),
    CONSTRAINT "SupplyFK" FOREIGN KEY (id_supp)
        REFERENCES schema."Supplier" (id_supp) MATCH SIMPLE
        ON UPDATE RESTRICT
        ON DELETE RESTRICT,
    CONSTRAINT "SupplyFK1" FOREIGN KEY (id_emp)
        REFERENCES schema."Employee" (id_emp) MATCH SIMPLE
        ON UPDATE RESTRICT
        ON DELETE RESTRICT,
    CONSTRAINT "SupplyFK2" FOREIGN KEY (id_product)
        REFERENCES schema."Product" (id_product) MATCH SIMPLE
        ON UPDATE RESTRICT
        ON DELETE RESTRICT,
    CONSTRAINT supp_chk CHECK (quantity_product >= 0 AND
quantity_product <= 99999),
    CONSTRAINT "Supply_id_supply_check" CHECK (id_supply >= 0),
    CONSTRAINT "Supply_account_number_sup_check" CHECK
(account_number_sup >= 0 AND account_number_sup <= 99999)
)
ALTER TABLE schema."Supply"
    OWNER to postgres;


--Заполнение таблицы Composition рабочими данными
INSERT INTO schema."Composition"(
        id_comp, purchase_quantity, id_product, id_purchase)
        VALUES (1, 100, 1, 1), (2, 200, 2, 2), (3, 300, 3, 3);


--Заполнение таблицы Customer рабочими данными
INSERT INTO schema."Customer"(
```

id_cust, name_comp, address_cust, date_order, selling_price)

VALUES (1,'ИЛИМ', 'г. Ярославль', 2020-02-12, 11000), (2, 'Титан', 'г. Кострома', 2020-02-15, 11500), (3, 'УЛК', 'г. Вологда', 2020-04-01, 12500);

--Заполнение таблицы Product рабочими данными

INSERT INTO schema."Product"(

id_product, information, name_prod, price_unit_prod, storage_conditions, reserve, unit_prod)

VALUES (1, 'Доска', 'пластина', 150, 'Ни в коем случае на закрывать пиломатериалы естественной влажности пленкой, чтобы не создавать парниковый эффект', 1000, 'шт'),

(2, 'Доска', 'четвертина', 200, 'Ни в коем случае на закрывать пиломатериалы естественной влажности пленкой, чтобы не создавать парниковый эффект', 1000, 'шт'),

(3, 'Доска', 'горбыль', 250, 'Ни в коем случае на закрывать пиломатериалы естественной влажности пленкой, чтобы не создавать парниковый эффект', 1000, 'шт');

--Заполнение таблицы Supplier рабочими данными

INSERT INTO schema."Supplier"(

id_supp, delivery_address, name_sup)

VALUES (1, 'г. Ярославль', 'Плайком'), (2, 'г. Кострома', 'Аврора'), (3, 'г. Вологда', 'Проксима');

--Заполнение таблицы Supply рабочими данными

INSERT INTO schema."Supply"(

id_supply, date_supply, quantity_product, delivery_price, account_number_sup, id_supp, id_emp, id_product)

VALUES (1, 2020-01-10, 30, 500.5, 10123, 1, 1, 1), (2, 2020-01-15, 30, 500.5, 20123, 2, 2, 2), (3, 2020-01-15, 32, 500.5, 30123, 3, 3, 3);

--Заполнение таблицы Purchase рабочими данными

INSERT INTO schema."Purchase"(

id_purchase, purchase_date, account_num, purchase_price, id_emp, id_cust)

VALUES (1, 2020-02-28, 10123, 4500, 1, 1), (2, 2020-03-02, 20123, 6000, 2, 2), (3, 2020-04-15, 30123, 800, 3, 3);

--Заполнение таблицы Employee рабочими данными

INSERT INTO schema."Employee"(

id_emp, passport, first_name, surname, last_name)

VALUES (1, 1000000009, 'Артем','Власов','Александрович'), (2, 2000000009, 'Егор','Зубов','Витальевич'), (3, 3000000009, 'Никита','Тихоненков','Алексеевич');

Вывод:

В ходе выполнения работы была создана база данных в PostgreSQL, созданы таблицы и ограничения на значение столбцов, в базу данных были занесены рабочие данные, а также была создана логическая модель базы данных и dump.