|  |
| --- |
| https://lh5.googleusercontent.com/cgN-uzOJBAt2De1EgGMN6Jer2bLvC94Zpza6kgvcAD3KNukigCfC3V4kB6ENrkLRxQ1ket9jTxhQotk0fzgOaZr-ZvyWz7isc7JHiQEdWpoAXtqQEj5XfPJEqBlp60WS-AKFuwMeHEUqlxgyO1aKogrUKC5R6OctDK5VbqCCBTr-UjXvYBcmpRqSV1A |
| МИНОБРНАУКИ РОССИИ  Федеральное государственное бюджетное образовательное учреждение  высшего образования  **«МИРЭА – Российский технологический университет»**  **РТУ МИРЭА** |

Институт комплексной безопасности и специального приборостроения

Кафедра КБ-4 «Интеллектуальные системы информационной безопасности»

**Клиент-серверные системы управления банком данных**

**Практическая работа № 2-5**

**Работа со сторонними базами данными. Построение и оптимизация**

**ОТЧЕТ**

Выполнил студент группы:

БСБО-07-20

Любовский С.В.

**Москва 2022г.**

Задание на практику:

1. Работа с ER-диаграммой и структурой базы данных
2. Заполнение базы данных, создание ролей, назначение привилегий
3. Автоматизация функционала
4. Реализация индексов, работа с выгрузкой данных

Отчет оформить в формате doc (docx) или pdf и выслать на проверку

**Выполнение задания**

1. ER-диаграмма схемы “shop”

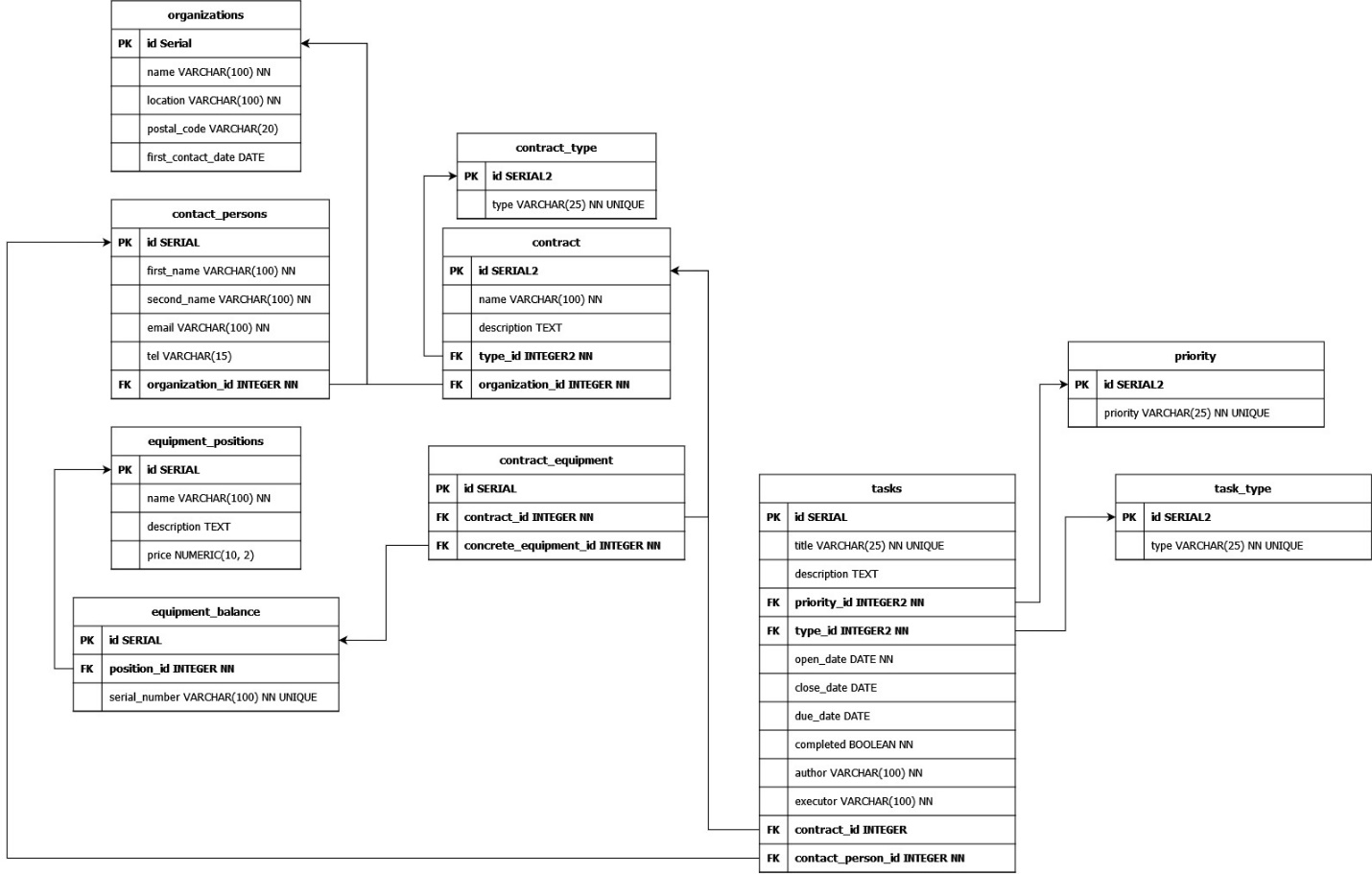


Рисунок 1. ER-диаграмма.

2. Листинг DDL скрипта для создания схемы “shop”

DROP SCHEMA IF EXISTS shop CASCADE;

CREATE SCHEMA "shop";

CREATE TABLE "shop"."organizations"(

    "id" SERIAL PRIMARY KEY,

    "name" VARCHAR(100) NOT NULL,

    "location" VARCHAR(100) NOT NULL,

    "postal\_code" VARCHAR(20),

    "first\_contract\_date" DATE

);

CREATE INDEX "ix\_organizations\_name" ON "shop"."organizations" ("name");

CREATE INDEX "ix\_organizations\_location" ON "shop"."organizations" ("location");

CREATE TABLE "shop"."contact\_persons"(

    "id" SERIAL PRIMARY KEY,

    "first\_name" VARCHAR(100) NOT NULL,

    "second\_name" VARCHAR(100) NOT NULL,

    "email" VARCHAR(100) NOT NULL,

    "tel" VARCHAR(15),

    "organization\_id" INT REFERENCES "shop"."organizations"("id") NOT NULL,

    UNIQUE("first\_name", "second\_name", "email")

);

CREATE INDEX "ix\_contact\_persons\_first\_name\_second\_name" ON "shop"."contact\_persons" ("first\_name", "second\_name");

CREATE TABLE "shop"."equipment\_positions" (

    "id" SERIAL PRIMARY KEY,

    "name" VARCHAR(100) NOT NULL,

    "description" TEXT,

    "price" NUMERIC(10, 2) NOT NULL

);

CREATE TABLE "shop"."equipment\_balance" (

    "id" SERIAL PRIMARY KEY,

    "position\_id" INT REFERENCES "shop"."equipment\_positions"("id") NOT NULL,

    "serial\_number" VARCHAR(100) NOT NULL UNIQUE

);

CREATE TABLE "shop"."contract\_type" (

    "id" SERIAL2 PRIMARY KEY,

    "type" VARCHAR(25) NOT NULL UNIQUE

);

CREATE TABLE "shop"."contract" (

    "id" SERIAL PRIMARY KEY,

    "name" VARCHAR(100) NOT NULL,

    "description" TEXT,

    "type\_id" INT2 REFERENCES "shop"."contract\_type" ("id") NOT NULL,

    "organization\_id" INT REFERENCES "shop"."organizations" ("id") NOT NULL

);

CREATE TABLE "shop"."contract\_equipment" (

    "id" SERIAL PRIMARY KEY,

    "contract\_id" INT REFERENCES "shop"."contract" ("id") NOT NULL,

    "concrete\_equipment\_id" INT REFERENCES "shop"."equipment\_balance" ("id") NOT NULL

);

CREATE TABLE "shop"."task\_type" (

    "id" SERIAL2 PRIMARY KEY,

    "type" VARCHAR(25) NOT NULL UNIQUE

);

CREATE TABLE "shop"."priority" (

    "id" SERIAL2 PRIMARY KEY,

    "priority" VARCHAR(25) NOT NULL UNIQUE

);

CREATE TABLE "shop"."tasks" (

    "id" SERIAL PRIMARY KEY,

    "title" VARCHAR(100) NOT NULL,

    "description" TEXT,

    "priority\_id" INT2 REFERENCES "shop"."priority" ("id") NOT NULL DEFAULT 0,

    "type\_id" INT2 REFERENCES "shop"."task\_type" ("id") NOT NULL,

    "open\_date" DATE NOT NULL DEFAULT current\_date,

    "close\_date" DATE,

    "due\_date" DATE,

    "completed" BOOLEAN NOT NULL DEFAULT False,

    "author" VARCHAR(100) NOT NULL,

    "executor" VARCHAR(100) NOT NULL,

    "contract\_id" INT REFERENCES "shop"."contract" ("id"),

    "contact\_person\_id" INT REFERENCES "shop"."contact\_persons" ("id") NOT NULL

);

3. Заполнение базы данных, создание ролей, назначение привилегий

3.1 Создание ролей и раздача привилегий

DROP ROLE IF EXISTS "admin";

CREATE ROLE "admin" LOGIN CREATEROLE ENCRYPTED PASSWORD 'supersecret';

DROP ROLE IF EXISTS "manager";

CREATE ROLE "manager" INHERIT;

DROP ROLE IF EXISTS "worker";

CREATE ROLE "worker" INHERIT;

REVOKE ALL ON schema public FROM public;

REVOKE ALL ON schema shop from manager, worker, public;

GRANT USAGE ON SCHEMA shop TO manager, worker;

GRANT SELECT ON ALL TABLES IN SCHEMA "shop" TO manager, worker;

GRANT INSERT, UPDATE ON "shop"."tasks" TO manager;

GRANT UPDATE ("close\_date", "completed") ON "shop"."tasks" TO worker;

GRANT EXECUTE ON FUNCTION "shop"."process\_check\_completed"() TO manager, worker;

GRANT ALL ON SCHEMA shop TO "admin";

3.2 Создание политик доступа к таблицам

ALTER TABLE "shop"."tasks" ENABLE ROW LEVEL SECURITY;

CREATE POLICY M\_\_W\_\_update ON "shop"."tasks"

    FOR UPDATE

    TO manager, worker

    USING (true)

    WITH CHECK (author = current\_user or executor = current\_user);

CREATE POLICY M\_\_W\_\_select ON "shop"."tasks"

    FOR SELECT

    TO manager, worker

    USING (author = current\_user or executor = current\_user);

CREATE POLICY A\_\_all ON "shop"."tasks"

    TO "admin"

    USING (true)

    WITH CHECK (true);

3.3 Заполнение таблицы данными по умолчанию и тестовыми данными

DO

$body$

    DECLARE

        organization\_id INTEGER;

        contract\_id INTEGER;

    BEGIN

        --- ENUMS

        INSERT INTO

            "shop"."contract\_type" ("type")

        VALUES

            ('supply'),

            ('installation'),

            ('service'),

            ('warranty repair'),

            ('post-warranty repair')

        ON CONFLICT DO NOTHING;

        INSERT INTO

            "shop"."task\_type" ("type")

        VALUES

            ('call'),

            ('email'),

            ('meeting'),

            ('contract service')

        ON CONFLICT DO NOTHING;

        INSERT INTO

            "shop"."priority" ("priority")

        VALUES

            ('low'),

            ('medium'),

            ('high')

        ON CONFLICT DO NOTHING;

        --- Organization, eqipment, contact persons, contract

        INSERT INTO

            "shop"."organizations" ("name", "location", "postal\_code", "first\_contract\_date")

        VALUES

            ('ООО Рога и Копыта', 'г. Москва, ул. Пушкина д. 5к1с1', '188213', current\_date)

        RETURNING id INTO organization\_id;

        INSERT INTO

            "shop"."contact\_persons" ("first\_name", "second\_name", "email", "tel", "organization\_id")

        VALUES

            ('Иван', 'Иванов', 'ivanov.ivan@org.id', '+79995551122', organization\_id);

        INSERT INTO

            "shop"."contract" ("name", "description", "type\_id", "organization\_id")

        VALUES

            ('Договор поставки оборудования 111147299', '', (SELECT id FROM "shop"."contract\_type" WHERE "type" = 'supply'), organization\_id)

        RETURNING id INTO contract\_id;

        INSERT INTO

            "shop"."equipment\_positions" ("name", "description", "price")

        VALUES

            ('Принтер', 'Принтер для печати документов', 10000),

            ('Сканер', 'Сканер для сканирования документов', 20000),

            ('Компьютер', 'Компьютер для работы с документами', 30000),

            ('МФУ', 'МФУ для печати и сканирования документов', 40000),

            ('Сервер', 'Сервер для хранения документов', 50000);

        INSERT INTO

            "shop"."equipment\_balance" ("position\_id", "serial\_number")

        VALUES

            ((SELECT id FROM "shop"."equipment\_positions" WHERE "name" = 'Принтер'), '000000001'),

            ((SELECT id FROM "shop"."equipment\_positions" WHERE "name" = 'Сканер'), '000000002'),

            ((SELECT id FROM "shop"."equipment\_positions" WHERE "name" = 'Компьютер'), '000000003'),

            ((SELECT id FROM "shop"."equipment\_positions" WHERE "name" = 'МФУ'), '000000004'),

            ((SELECT id FROM "shop"."equipment\_positions" WHERE "name" = 'Сервер'), '000000005'),

            ((SELECT id FROM "shop"."equipment\_positions" WHERE "name" = 'Сервер'), '100000005');

        INSERT INTO

            "shop"."contract\_equipment" ("contract\_id", "concrete\_equipment\_id")

        VALUES

            (contract\_id, (SELECT id FROM "shop"."equipment\_balance" WHERE serial\_number = '100000005')),

            (contract\_id, (SELECT id FROM "shop"."equipment\_balance" WHERE serial\_number = '000000003'));

        --- Manager user

        CREATE USER default\_worker1 IN ROLE worker ENCRYPTED PASSWORD '1234';

        CREATE USER default\_worker2 IN ROLE worker ENCRYPTED PASSWORD '1234';

        CREATE USER default\_manager1 IN ROLE manager ENCRYPTED PASSWORD '1234';

        CREATE USER default\_manager2 IN ROLE manager ENCRYPTED PASSWORD '1234';

        --- Create tasks

        INSERT INTO

            "shop"."tasks" ("title", "description", "priority\_id", "type\_id", "due\_date", "author", "executor", "contract\_id", "contact\_person\_id")

        VALUES

            ('Задача 1', 'Описание задачи 1', (SELECT id FROM "shop"."priority" WHERE "priority" = 'low'), (SELECT id FROM "shop"."task\_type" WHERE "type" = 'meeting'), current\_date + 1, 'default\_manager1', 'default\_worker1', contract\_id, (SELECT id FROM "shop"."contact\_persons" WHERE "first\_name" = 'Иван')),

            ('Задача 2', 'Описание задачи 2', (SELECT id FROM "shop"."priority" WHERE "priority" = 'medium'), (SELECT id FROM "shop"."task\_type" WHERE "type" = 'call'), current\_date + 2, 'default\_manager1', 'default\_worker2', contract\_id, (SELECT id FROM "shop"."contact\_persons" WHERE "first\_name" = 'Иван')),

            ('Задача 3', 'Описание задачи 3', (SELECT id FROM "shop"."priority" WHERE "priority" = 'high'), (SELECT id FROM "shop"."task\_type" WHERE "type" = 'email'), current\_date + 3, 'default\_manager2', 'default\_worker1', contract\_id, (SELECT id FROM "shop"."contact\_persons" WHERE "first\_name" = 'Иван')),

            ('Задача 4', 'Описание задачи 4', (SELECT id FROM "shop"."priority" WHERE "priority" = 'low'), (SELECT id FROM "shop"."task\_type" WHERE "type" = 'contract service'), current\_date + 4, 'default\_manager2', 'default\_worker2', contract\_id, (SELECT id FROM "shop"."contact\_persons" WHERE "first\_name" = 'Иван'));

    END;

$body$

LANGUAGE 'plpgsql';

4. Автоматизация функционала. Создание функций и триггеров.

CREATE OR REPLACE FUNCTION "shop"."get\_roles\_list"()

RETURNS TABLE("role" TEXT)

LANGUAGE plpgsql

AS $$

BEGIN

    RETURN QUERY WITH RECURSIVE cte AS (

        SELECT oid FROM pg\_roles WHERE rolname = current\_user

        UNION ALL

        SELECT m.roleid

        FROM cte

        JOIN pg\_auth\_members m ON m.member = cte.oid

    )

    SELECT oid::regrole::text AS rolename FROM cte;

END;

$$;

CREATE OR REPLACE FUNCTION "shop"."process\_check\_completed"()

RETURNS TRIGGER

LANGUAGE plpgsql

AS $$

    BEGIN

        IF (TG\_OP = 'UPDATE') THEN

            IF (OLD.completed = True and 'admin' not in (SELECT "shop"."get\_roles\_list"())) THEN

                RAISE EXCEPTION 'Can not change completed task';

            END IF;

            NEW.close\_date = current\_date;

        END IF;

        RETURN NEW;

    END;

$$;

CREATE TRIGGER "check\_completed"

BEFORE UPDATE ON "shop"."tasks"

    FOR EACH ROW EXECUTE PROCEDURE "shop"."process\_check\_completed"();

CREATE OR REPLACE FUNCTION "shop"."process\_expire\_complited\_tasks"()

RETURNS trigger

LANGUAGE plpgsql

    AS $$

        BEGIN

        DELETE FROM "shop"."tasks" WHERE close\_date < current\_date - INTERVAL '12 month';

        RETURN NEW;

    END;

$$;

CREATE TRIGGER expire\_complited\_tasks

AFTER INSERT OR UPDATE ON "shop"."tasks"

    FOR EACH ROW EXECUTE PROCEDURE "shop"."process\_expire\_complited\_tasks"();

CREATE OR REPLACE FUNCTION "shop"."create\_report"("user" VARCHAR(100), date\_start DATE, date\_end DATE)

RETURNS TABLE(

    "task\_count" INT,

    "completed\_task\_count" INT,

    "completed\_out\_of\_date\_task\_count" INT,

    "not\_completed\_task\_count" INT,

    "not\_completed\_out\_of\_date\_task\_count" INT

)

LANGUAGE plpgsql

AS $$

BEGIN

    RETURN QUERY SELECT

        count(\*) :: int as task\_count,

        sum(case when completed and close\_date <= due\_date then 1 else 0 end) :: int as completed\_task\_count,

        sum(case when completed and close\_date > due\_date then 1 else 0 end) :: int as completed\_out\_of\_date\_task\_count,

        sum(case when not completed and due\_date >= current\_date then 1 else 0 end) :: int as not\_completed\_task\_count,

        sum(case when not completed and due\_date < current\_date then 1 else 0 end) :: int as not\_completed\_out\_of\_date\_task\_count

    FROM "shop"."tasks"

    WHERE executor = "user" and open\_date between date\_start and date\_end;

END;

$$;

5. Для выгрузки отчета в файл необходимо зайти на сервер с помощью утилиты psql. Далее ввести следующую команду:

\copy (select \* from shop.create\_report('[ЛОГИН\_ПОЛЬЗОВАТЕЛЯ]', [ДАТА\_НАЧАЛА\_ОТЧЕТА], [ДАТА\_КОНЦА\_ОТЧЕТА])) TO '[ПОЛЫЙ\_ПУТЬ\_ДО\_ФАЙЛА]' DELIMITER ',' CSV HEADER