Домашнее задание №4

Шалаев Алексей Дмитриевич БПИ222

1 Задача

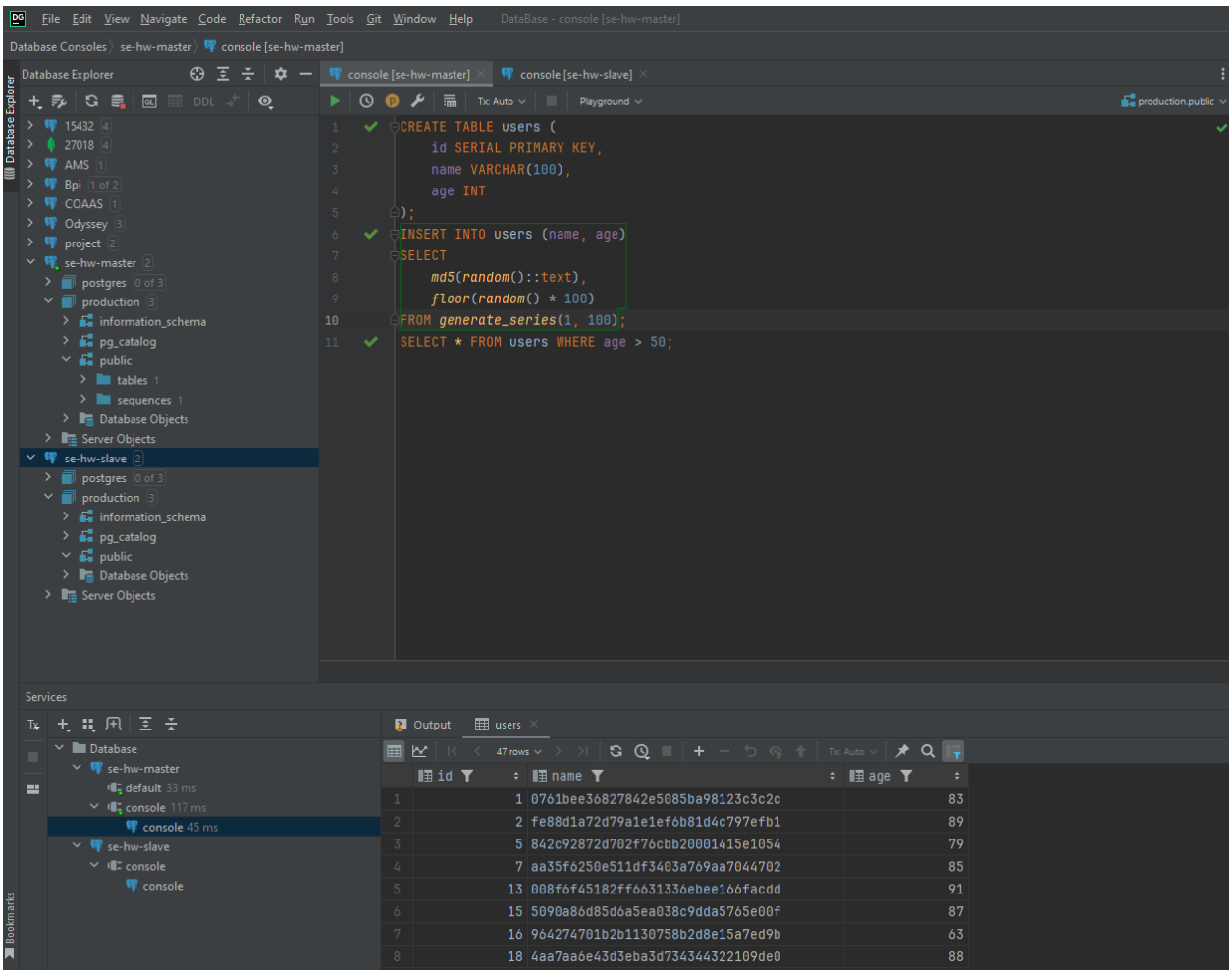
1. Создается docker-compose.yml с двумя сервисами: PostgreSQL

   Написал ради интереса 2 инстанса постгры: master и slave, где slave – реплика master, ее, например, можно исп. для чтения, а master для read & write операций.

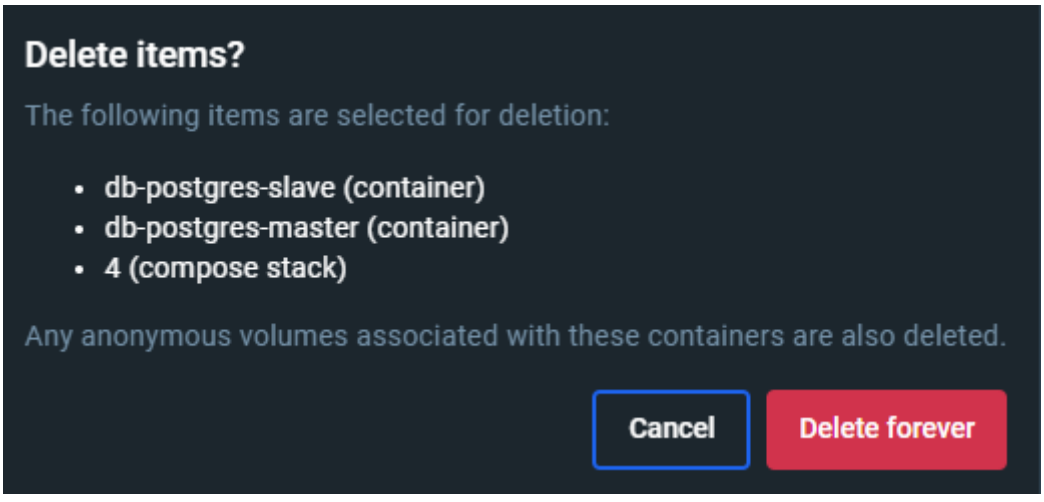
```yaml
version: "3.9"

x-postgres-common:
  &postgres-common
  image: postgres:15
  user: postgres
  healthcheck:
      test: [ "CMD-SHELL", "pg_isready", "-d", "production"]
      interval: 10s
      timeout: 3s
      retries: 3
  restart: on-failure:3
  networks:
    - db-network

services:
  postgres-master:
    <<: *postgres-common
    container_name: db-postgres-master
    environment:
      POSTGRES_USER: avito
      POSTGRES_PASSWORD: hackme
      POSTGRES_DB: production
      REPLICATION_USER: replicator
      REPLICATION_PASSWORD: aboba
      POSTGRES_HOST_AUTH_METHOD: "scram-sha-256\nhost replication all 0.0.0.0/0
md5"
      POSTGRES_INITDB_ARGS: "--auth-host=scram-sha-256"
    volumes:
      - db-postgres-master-data:/var/lib/postgresql/data
      - ./init-scripts:/docker-entrypoint-initdb.d/
    ports:
      - "15432:5432"
    command: |
      postgres
      -c wal_level=replica
      -c hot_standby=on
      -c max_wal_senders=10
```

```
      -c max_replication_slots=10
      -c hot_standby_feedback=on


  postgres-slave:
    <<: *postgres-common
    container_name: db-postgres-slave
    environment:
      PGUSER: replicator
      PGPASSWORD: replicator-password
    depends_on:
      - postgres-master
    ports:
      - "25432:5432"
    command: |
      bash -c "
      rm -rf /var/lib/postgresql/data/*
      until pg_basebackup --pgdata=/var/lib/postgresql/data -R --
slot=replication_slot --host=postgres-master --port=5432
      do
      echo 'Waiting for primary to connect...'
      sleep 1s
      done
      echo 'Backup done, starting replica...'
      chmod 0700 /var/lib/postgresql/data
      postgres
      "

networks:
  db-network:

volumes:
  db-postgres-master-data:
    name: db-postgres-master-data
```

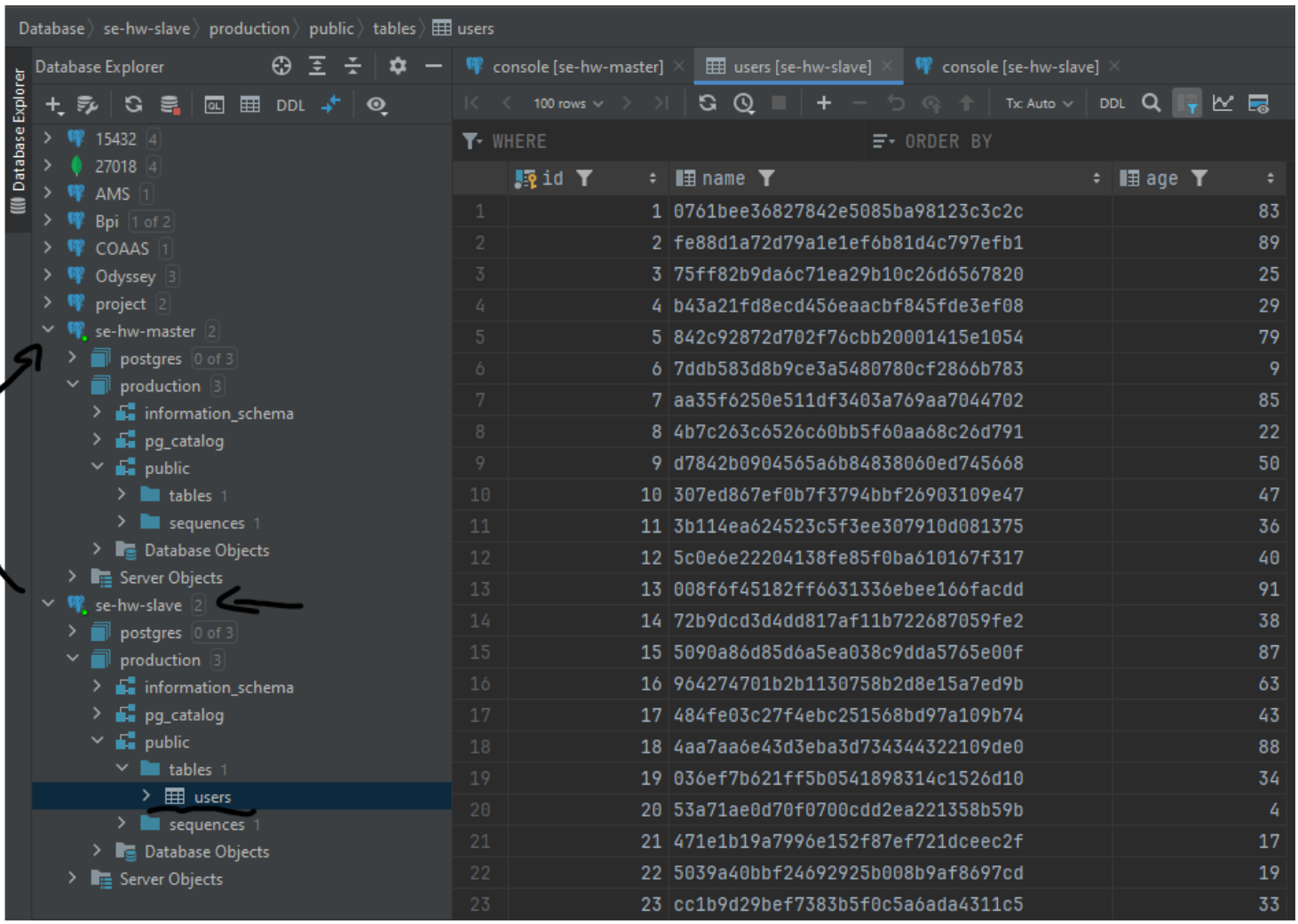
2. Загрузите туда данные
3. Подключитесь к базе данных через DataGrip
4. Выполните тестовые SQL-запросы



5. Удалите контейнеры
6. Сделайте скриншоты и опишите результаты



7. Запустите контейнеры снова и проверьте, что данные остались или нет
8. Сделайте скриншоты и опишите результаты



Так как я создал volume для master postgres, то данные остались, более того из настроенной репликации, они есть и в slave db.

2 Задача

В папке init-scripts есть 3 скрипта, которые создают таблицы в master:

01_library.sql

```
CREATE TABLE "books" (
  "ISBN" varchar PRIMARY KEY,
  "title" varchar NOT NULL,
  "author" varchar NOT NULL,
  "year" int NOT NULL,
  "pages" int NOT NULL,
  "publisher_id" int NOT NULL,
  "created_at" timestamp NOT NULL
);

CREATE TABLE "categories" (
```

```
  "id" SERIAL PRIMARY KEY,
  "name" varchar UNIQUE NOT NULL,
  "parent_category_id" int
);

CREATE TABLE "book_categories" (
  "book_ISBN" varchar NOT NULL,
  "category_id" int NOT NULL,
  PRIMARY KEY ("book_ISBN", "category_id")
);

CREATE TABLE "book_copies" (
  "copy_number" SERIAL PRIMARY KEY,
  "ISBN" varchar NOT NULL,
  "shelf_location" varchar NOT NULL,
  "created_at" timestamp NOT NULL
);

CREATE TABLE "publishers" (
  "id" SERIAL PRIMARY KEY,
  "name" varchar NOT NULL,
  "address" varchar NOT NULL,
  "created_at" timestamp NOT NULL
);

CREATE TABLE "readers" (
  "id" SERIAL PRIMARY KEY,
  "first_name" varchar NOT NULL,
  "last_name" varchar NOT NULL,
  "address" varchar NOT NULL,
  "birth_date" date NOT NULL,
  "created_at" timestamp NOT NULL
);

CREATE TABLE "borrows" (
  "id" SERIAL PRIMARY KEY,
  "reader_id" int NOT NULL,
  "copy_number" int NOT NULL,
  "borrow_date" timestamp NOT NULL,
  "return_date" timestamp
);

ALTER TABLE "books" ADD FOREIGN KEY ("publisher_id") REFERENCES "publishers"
("id");

ALTER TABLE "categories" ADD FOREIGN KEY ("parent_category_id") REFERENCES
"categories" ("id");

ALTER TABLE "book_categories" ADD FOREIGN KEY ("book_ISBN") REFERENCES "books"
("ISBN");
```

```
ALTER TABLE "book_categories" ADD FOREIGN KEY ("category_id") REFERENCES
"categories" ("id");

ALTER TABLE "book_copies" ADD FOREIGN KEY ("ISBN") REFERENCES "books" ("ISBN");

ALTER TABLE "borrows" ADD FOREIGN KEY ("reader_id") REFERENCES "readers" ("id");

ALTER TABLE "borrows" ADD FOREIGN KEY ("copy_number") REFERENCES "book_copies"
("copy_number");
```

02_stations.sql

```
CREATE TABLE "train_stations" (
  "name" varchar PRIMARY KEY,
  "num_tracks" int NOT NULL,
  "city_name" varchar NOT NULL
);

CREATE TABLE "cities" (
  "name" varchar PRIMARY KEY,
  "region" varchar NOT NULL
);

CREATE TABLE "trains" (
  "train_number" varchar PRIMARY KEY,
  "length" int NOT NULL
);

CREATE TABLE "journeys" (
  "id" SERIAL PRIMARY KEY,
  "train_number" varchar NOT NULL,
  "departure_station_name" varchar NOT NULL,
  "arrival_station_name" varchar NOT NULL,
  "departure_time" timestamp NOT NULL,
  "arrival_time" timestamp NOT NULL
);

ALTER TABLE "train_stations" ADD FOREIGN KEY ("city_name") REFERENCES "cities"
("name");

ALTER TABLE "journeys" ADD FOREIGN KEY ("train_number") REFERENCES "trains"
("train_number");

ALTER TABLE "journeys" ADD FOREIGN KEY ("departure_station_name") REFERENCES
"train_stations" ("name");

ALTER TABLE "journeys" ADD FOREIGN KEY ("arrival_station_name") REFERENCES
"train_stations" ("name");
```

03_hospital.sql

```
CREATE TABLE "stations" (
  "station_number" int PRIMARY KEY,
  "name" varchar NOT NULL
);

CREATE TABLE "rooms" (
  "room_number" int PRIMARY KEY,
  "number_of_beds" int NOT NULL,
  "station_number" int NOT NULL
);

CREATE TABLE "patients" (
  "patient_number" int PRIMARY KEY,
  "name" varchar NOT NULL,
  "disease" varchar NOT NULL,
  "doctor_personnel_number" int NOT NULL
);

CREATE TABLE "station_personnel" (
  "personnel_number" int PRIMARY KEY,
  "name" varchar NOT NULL,
  "station_number" int NOT NULL
);

CREATE TABLE "doctors" (
  "personnel_number" int PRIMARY KEY,
  "rank" varchar NOT NULL,
  "area" varchar NOT NULL
);

CREATE TABLE "caregivers" (
  "personnel_number" int PRIMARY KEY,
  "qualification" varchar NOT NULL
);

CREATE TABLE "admissions" (
  "patient_number" int NOT NULL,
  "room_number" int NOT NULL,
  "admission_from" timestamp NOT NULL,
  "admission_to" timestamp,
  PRIMARY KEY ("patient_number", "room_number")
);

ALTER TABLE "rooms" ADD FOREIGN KEY ("station_number") REFERENCES "stations"
("station_number");

ALTER TABLE "patients" ADD FOREIGN KEY ("doctor_personnel_number") REFERENCES
"doctors" ("personnel_number");
```

```
ALTER TABLE "station_personnel" ADD FOREIGN KEY ("station_number") REFERENCES
"stations" ("station_number");

ALTER TABLE "doctors" ADD FOREIGN KEY ("personnel_number") REFERENCES
"station_personnel" ("personnel_number");

ALTER TABLE "caregivers" ADD FOREIGN KEY ("personnel_number") REFERENCES
"station_personnel" ("personnel_number");

ALTER TABLE "admissions" ADD FOREIGN KEY ("patient_number") REFERENCES "patients"
("patient_number");

ALTER TABLE "admissions" ADD FOREIGN KEY ("room_number") REFERENCES "rooms"
("room_number");
```

Конечно, стоило эти таблица создавать в своих бд или хотя бы в разных схемах: