МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МО ЭВМ

ОТЧЕТ

по лабораторной работе №2

по дисциплине «Базы данных»

Тема: Реализация базы данных в СУБД PostgreSQL

Студент гр. 3384 _____ Рудаков А.Л.

Преподаватель _____ Михайлова С.А.

Санкт-Петербург

2025

Цель работы.

Изучение и практическое освоение навыков работы с СУБД PostgreSQL путем создания таблиц, их заполнения и запросов их данных через SQL-запросы.

Задание.

Вариант 22.

Пусть требуется создать программную систему для поиска вакансий (аналог hh.ru). Такая система должна обеспечивать хранение сведений о работодателях и работниках. Эти сведения включают в себя (для работника) - паспортные работника, данные трудовой книжки, ИНН, дата рождения, информацию о среднем/высшем(их) образованиях, дата поступления на работу, в институт, информация об предыдущей работе(ах) из трудовой книжки. Данные трудовой книжки – это ее номер и дата выдачи, а также даты и номера приказов о зачислении и увольнении, о переходе в другое подразделение или об изменении должности. Кроме того, для работник может создать 1/несколько резюме, с указанием желаемой должности, ЗП, свои умения/навыки. Работодатель имеет возможность создавать/удалять/помещать в архив вакансии. Вакания имеет название, ЗП, должность, адрес, требования, условия, комментарий, требуемый опыт. У работодателя есть страница с указанием информации о себе - название, фото, описание, файлы презентации, сфера деятельности (IT, финансовая и т.п.), количество вакансий (формируется на основе списка вакансий).

Система должна давать ответы на следующие вопросы:

- Какие вакансии есть у данной компании?
- Какая вакансия подходит мне по названию?
- Сколько поблизости вакансий от меня (указание улицы)?
- Какие вакансии были помещены в архив у компании?
- Средняя ЗП каждого работодателя?
- Сколько работников ищут работу, имея высшее образование?
- Сколько работников имело более 3-х мест работы?
- Какие вакансии имеют ЗП более 100 000р и не требуют опыта работы?

Необходимо развернуть PostgreSQL локально:

- Написать запросы для создания таблиц из предыдущей лабораторный работы
- Заполнить тестовыми данными: 5-10 строк на каждую таблицу, обязательно наличие связи между ними, данные приближены к реальности.
- Написать запросы к БД, отвечающие на вопросы из предыдущей лабораторной работы
- Исходный код выложить на www.db-fiddle.com для проверки работоспособности
- Исходный код в виде .sql файла запушить в виде PR в репо

Выполнение работы.

Были созданы таблицы:

- passport
  - passport_id SERIAL PRIMARY KEY,
  - passport_number INT NOT NULL,
  - passport_series INT NOT NULL,
  - UNIQUE (passport_number, passport_series)
- tin
  - tin_id SERIAL PRIMARY KEY,
  - tin INT UNIQUE NOT NULL
- workbook
  - workbook_id SERIAL PRIMARY KEY,
  - workbook_number INT UNIQUE NOT NULL,
  - isue_date DATE
- worker
  - worker_id SERIAL PRIMARY KEY,
  - passport_id INT NOT NULL,
  - tin_id INT NOT NULL,

    - birthday DATE NOT NULL,
    - FOREIGN KEY (passport_id) REFERENCES passport (passport_id),
    - FOREIGN KEY (tin_id) REFERENCES tin (tin_id)
- worker_workbook
    - worker_id INT NOT NULL,
    - workbook_id INT NOT NULL,
    - PRIMARY KEY (worker_id, workbook_id),
    - FOREIGN KEY (worker_id) REFERENCES worker (worker_id),
    - FOREIGN KEY (workbook_id) REFERENCES workbook (workbook_id)
- education
    - education_id SERIAL PRIMARY KEY,
    - name VARCHAR(255) NOT NULL,
    - type education_type NOT NULL
- worker_education
    - worker_id INT NOT NULL,
    - education_id INT NOT NULL,
    - enroll_date DATE NOT NULL,
    - PRIMARY KEY (worker_id, education_id),
    - FOREIGN KEY (worker_id) REFERENCES worker (worker_id),
    - FOREIGN KEY (education_id) REFERENCES education (education_id)
- post
    - post_id SERIAL PRIMARY KEY,
    - name VARCHAR(255) UNIQUE NOT NULL
- resume
    - resume_id SERIAL PRIMARY KEY,
    - worker_id INT NOT NULL,
    - post_id INT NOT NULL,

    - salary DECIMAL(8, 2),
    - FOREIGN KEY (worker_id) REFERENCES worker (worker_id),
    - FOREIGN KEY (post_id) REFERENCES post (post_id)
- skill
    - skill_id SERIAL PRIMARY KEY,
    - description VARCHAR(255) UNIQUE NOT NULL
- resume_skill
    - resume_id INT NOT NULL,
    - skill_id INT NOT NULL,
    - PRIMARY KEY (resume_id, skill_id),
    - FOREIGN KEY (resume_id) REFERENCES resume (resume_id),
    - FOREIGN KEY (skill_id) REFERENCES skill (skill_id)
- field_of_activity
    - activity_id SERIAL PRIMARY KEY,
    - name VARCHAR(255) UNIQUE NOT NULL
- employer
    - employer_id SERIAL PRIMARY KEY,
    - activity_id INT NOT NULL,
    - name VARCHAR(255) NOT NULL,
    - photo_url VARCHAR(255),
    - description VARCHAR(255),
    - FOREIGN KEY (activity_id) REFERENCES field_of_activity (activity_id)
- presentation_file
    - file_id SERIAL PRIMARY KEY,
    - employer_id INT NOT NULL,
    - name VARCHAR(255) NOT NULL,
    - url VARCHAR(255) NOT NULL,
    - FOREIGN KEY (employer_id) REFERENCES employer (employer_id)

- workbook_writing
    - writing_id SERIAL PRIMARY KEY,
    - type writing_type NOT NULL,
    - date DATE NOT NULL,
    - description VARCHAR(255)
- worker_employer_writing
    - writing_id INT NOT NULL,
    - workbook_id INT NOT NULL,
    - employer_id INT NOT NULL,
    - PRIMARY KEY (writing_id, workbook_id, employer_id),
    - FOREIGN KEY (writing_id) REFERENCES workbook_writing (writing_id),
    - FOREIGN KEY (workbook_id) REFERENCES workbook (workbook_id),
    - FOREIGN KEY (employer_id) REFERENCES employer (employer_id)
- vacancy
    - vacancy_id SERIAL PRIMARY KEY,
    - post_id INT NOT NULL,
    - employer_id INT NOT NULL,
    - name VARCHAR NOT NULL,
    - salary DECIMAL(8, 2) NOT NULL,
    - experience INT,
    - comment VARCHAR(255),
    - status vacancy_status NOT NULL,
    - FOREIGN KEY (post_id) REFERENCES post (post_id),
    - FOREIGN KEY (employer_id) REFERENCES employer (employer_id)
- requirement
    - requirement_id SERIAL PRIMARY KEY,

    - description VARCHAR(255) UNIQUE NOT NULL
- vacancy_requirement
    - vacancy_id INT NOT NULL,
    - requirement_id INT NOT NULL,
    - PRIMARY KEY (vacancy_id, requirement_id),
    - FOREIGN KEY (vacancy_id) REFERENCES vacancy (vacancy_id),
    - FOREIGN KEY (requirement_id) REFERENCES requirement (requirement_id)
- condition
    - condition_id SERIAL PRIMARY KEY,
    - description VARCHAR(255) UNIQUE NOT NULL
- vacancy_condition
    - vacancy_id INT NOT NULL,
    - condition_id INT NOT NULL,
    - PRIMARY KEY (vacancy_id, condition_id),
    - FOREIGN KEY (vacancy_id) REFERENCES vacancy (vacancy_id),
    - FOREIGN KEY (condition_id) REFERENCES condition (condition_id)

Так же созданы ENUM’ы для некоторых данных в таблицах:

- education_type
    - Высшее
    - Среднее
- writing_type
    - Зачисление,
    - Увольнение,
    - Переход,
    - Изменение должности

- vacancy_status
    - Активна,
    - В архиве,
    - Удалена

На рис.1. представлен пример выполнения запроса по созданию таблицы в СУБД PostgreSQL DataGrid.

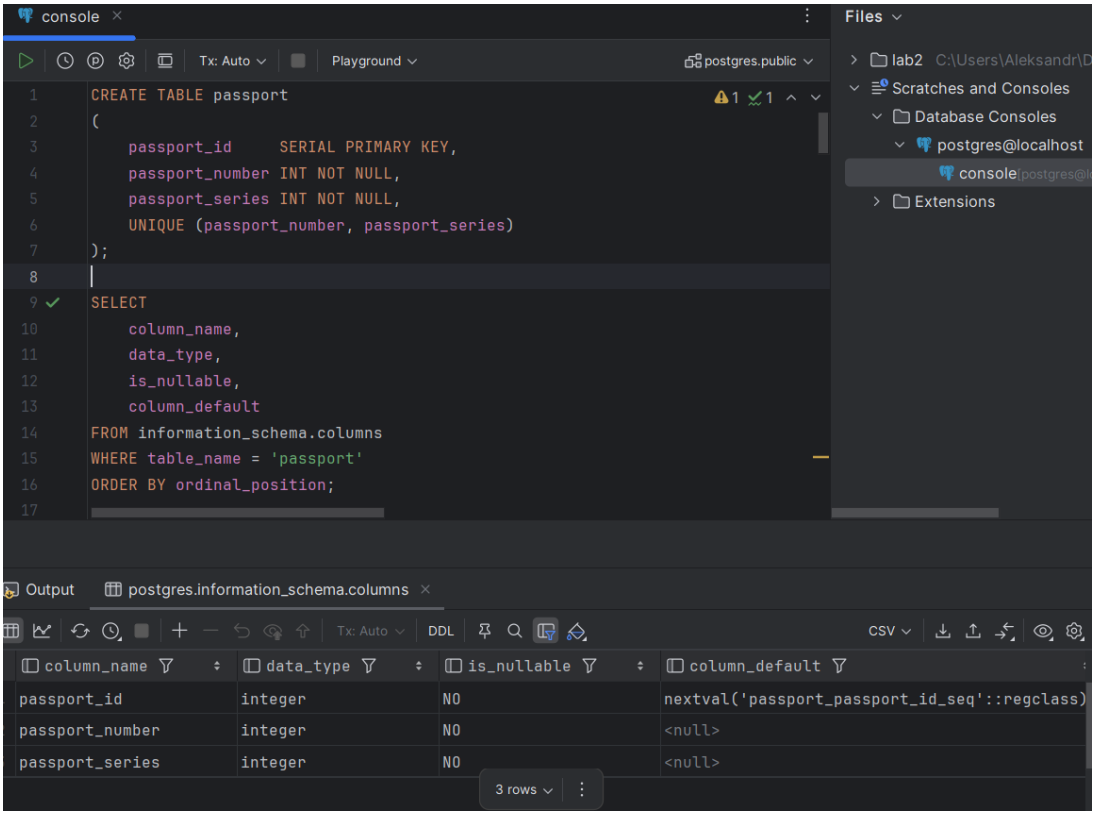

Рисунок 1 – пример выполнения запроса создания в DataGrid

Далее в каждую из таблиц были вставлены данные:

- passport – данные о номерах и сериях 6 разных паспортов.
- tin – данные о номерах 6 разных ИНН.
- worker – данные о связи 6 работников с их паспортами и ИНН, а также данные об их датах рождения.
- education – список из 8 образовательных учреждений высшего и среднего образования.
- worker_education – сведения о 7 связях работников с их образованиями, при этом у работников 1 и 5 по 2 образования.
- workbook – сведения о номерах и датах выдачи 7 трудовых книжек.

- worker_workbook – сведения о 7 связях работников с их трудовыми книжками, при этом работник 5 имеет 2 трудовые книжки.
- post – список из 10 профессий.
- skill – список из 9 умений/навыков.
- resume – данные о 7 составленных резюме, связанных с работниками, при этом у работников 3 и 5 есть 2 резюме, у работника 2 ни одного.
- resume_skill – данные о 10 связях вакансий с навыками, при этом в вакансии 1 указано 3 навыка, а у вакансии 4 ни одного.
- field_of_activity – список из 5 сфер деятельности.
- employer – данные о 6 работодателях, включающие название, связь со сферой активности, фото, и описание. При этом 4 и 6 работают в одинаковых сферах.
- presentation_file – данные о 5 файлах презентаций и их связях с работодателями. При этом у работодателя 1 2 файла, а у 3 и 6 ни одного.
- requirement – список из 10 требований.
- condition – список из 8 условий.
- vacancy – данные о 7 вакансиях, включающие пост, работодателя, название, зарплата, адрес, опыт, комментарий и статус. При этом могут попадаться вакансии на одинаковый пост или от одной и той де компании.
- vacancy_requirement – данные о 10 связях вакансий с требованиями, при этом у 1 вакансии 3 требования, а у 6 ни одного.
- vacancy_condition – данные о 8 связях вакансий с условиями, при этом у 1 вакансии 2 условия, а у 3 ни одного.
- workbook_writing – данные о 10 записях в трудовых книжках, включающие тип, дату и комментарий.
- worker_employer_writing – данные о 10 связях работников с работодателями и записями в трудовых книжках.

На рис.2. представлен пример выполнения запроса по заполнению таблицы в СУБД PostgreSQL DataGrid.

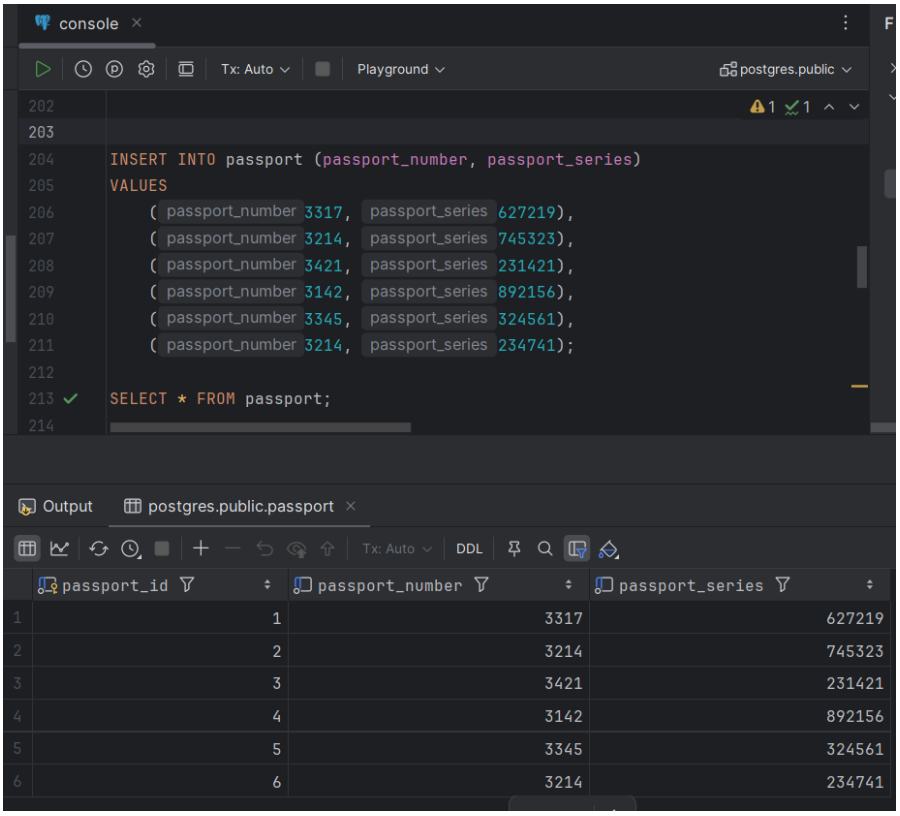

Рисунок 2 – пример выполнения запроса заполнения в DataGrid

Запросы к вопросам:

1. Какие вакансии есть у данной компании?

Запрос выполнен путем соединения таблиц vacancy, employer, post, при помощи JOIN … USING. Столбцы с id каждой таблицы названы с учетом названия таблицы как раз для того, чтобы можно было использовать USING вместо ON. Из таблиц выбираются только те записи, в названии компаний которых указано заданное название.

На рис.3. представлен вывод выполнения запроса 1 в СУБД PostgreSQL DataGrid.

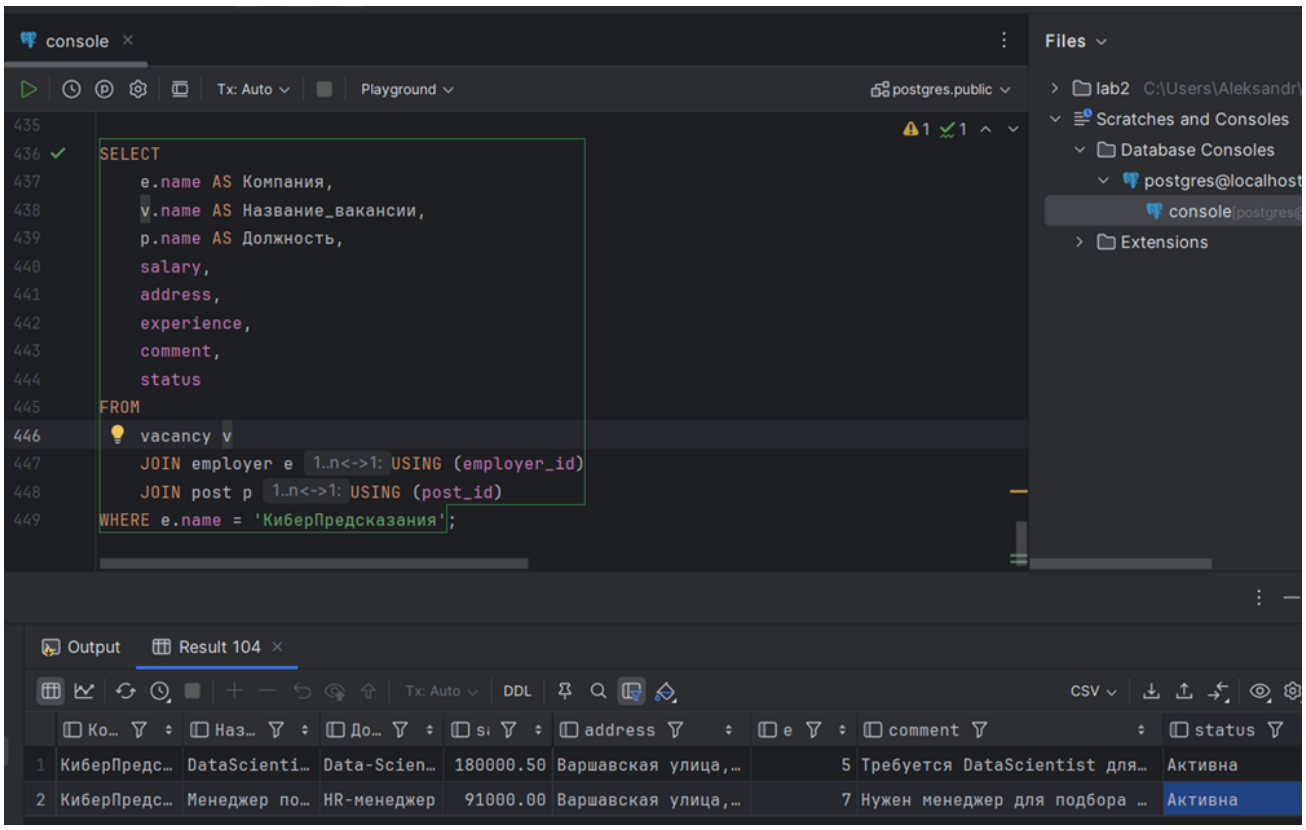

Рисунок 3 – Выполнение запроса 1 выборки в DataGrid

2. Какая вакансия подходит мне по названию?

При помощи JOIN … USING соединены таблицы vacancy и post (через post_id) и employer (через employer_id). Выбраны вакансии, в названии которых есть запрашиваемое слово.

На рис.4 представлен вывод выполнения запроса 2 в СУБД PostgreSQL DataGrid.

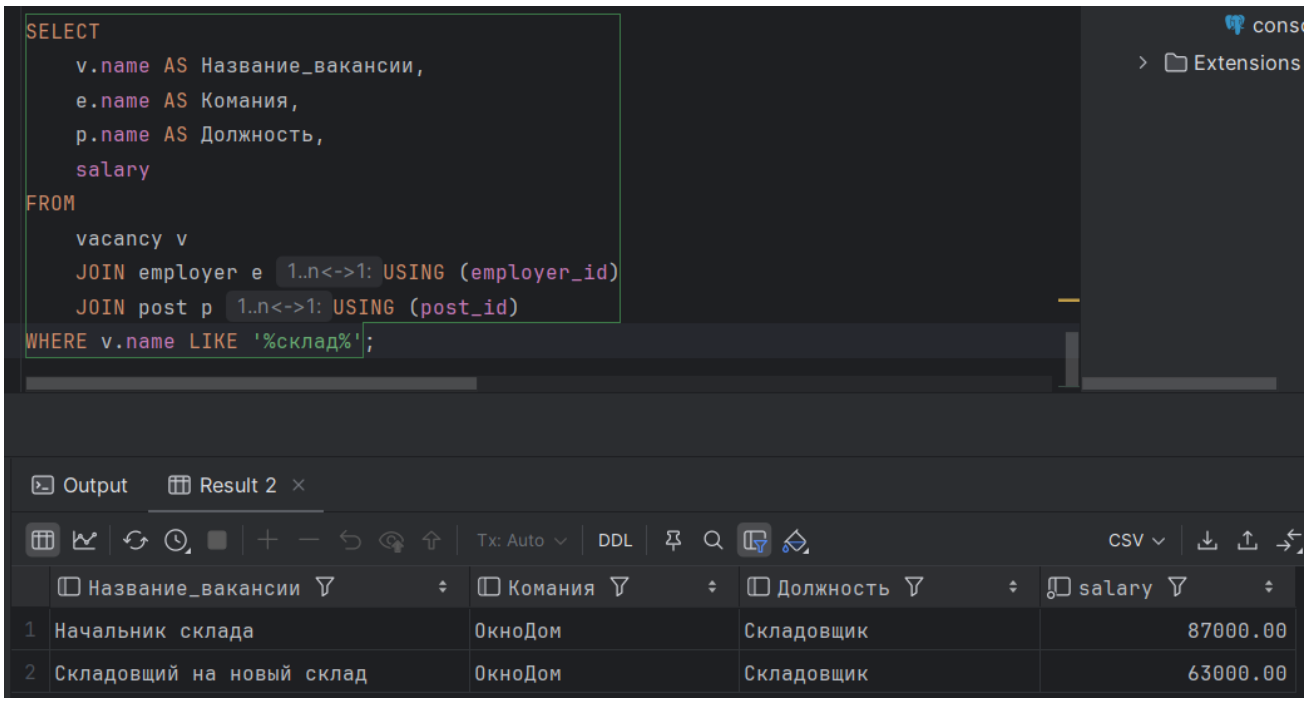

Рисунок 4 – Выполнение запроса 2 выборки в DataGrid

3. Сколько поблизости вакансий от меня (указание улицы)?

Запрос выполнен путем подсчета (count) количества записей, выдаваемых по запросу поиска вакансий с указанием улицы.

На рис.5 представлен вывод выполнения запроса 3 в СУБД PostgreSQL DataGrid.

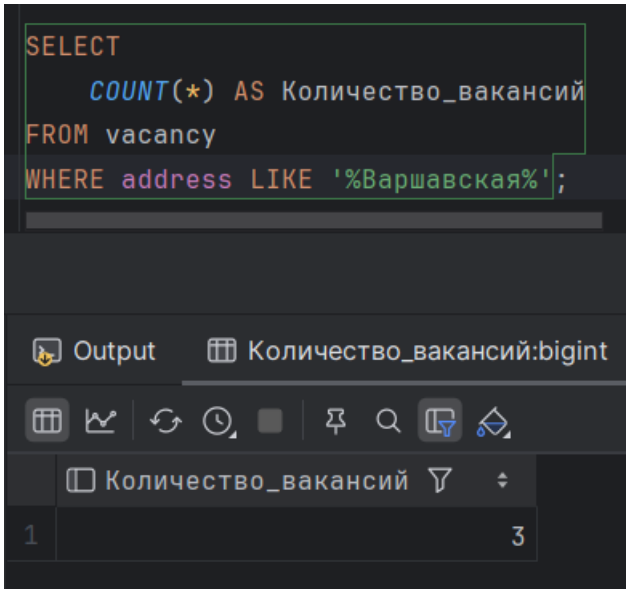

Рисунок 5 – Выполнение запроса 3 выборки в DataGrid

4. Какие вакансии были помещены в архив у компании?

При помощи JOIN … USING соединены таблицы vacancy и post (через post_id) и employer (через employer_id). Выбраны вакансии, статус которых ‘В архиве’ и название компании соответствует заданному.

На рис.6 представлен вывод выполнения запроса 4 в СУБД PostgreSQL DataGrid.

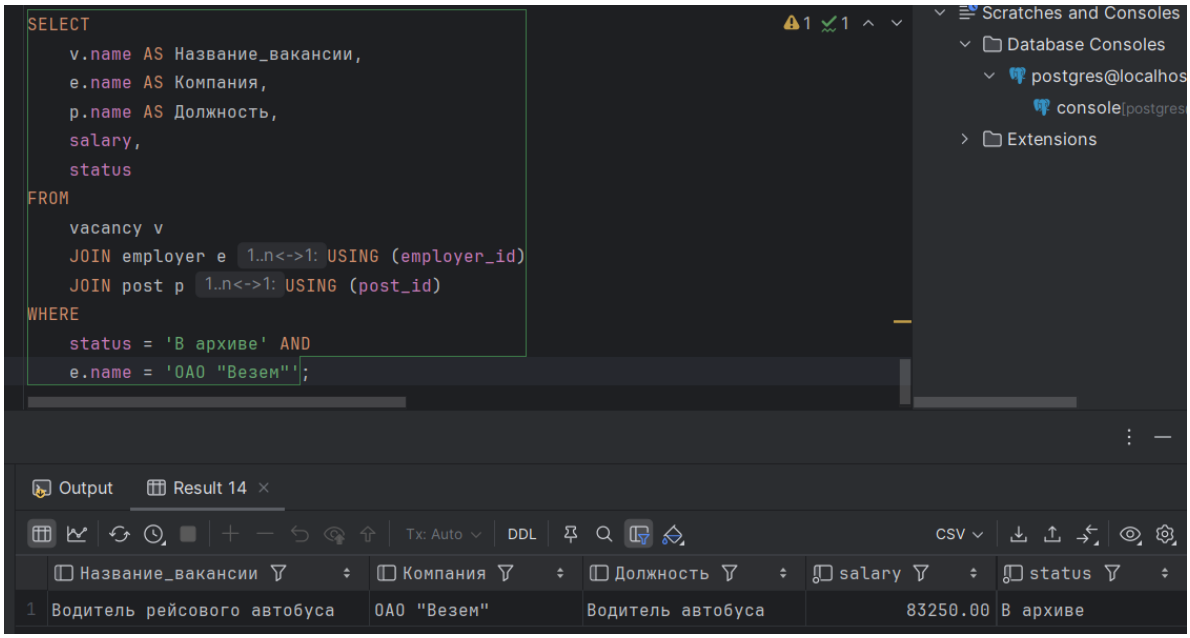

Рисунок 6 – Выполнение запроса 4 выборки в DataGrid

5. Средняя ЗП каждого работодателя?

При помощи JOIN … USING соединены таблицы vacancy и post (через post_id) и employer (через employer_id). Выводятся названия компаний и средние зарплаты в них, исходя из вакансий. Группировка выполнена по названиям компаний. Сортировка вывода по убыванию средней зп.

На рис.7 представлен вывод выполнения запроса 5 в СУБД PostgreSQL DataGrid.

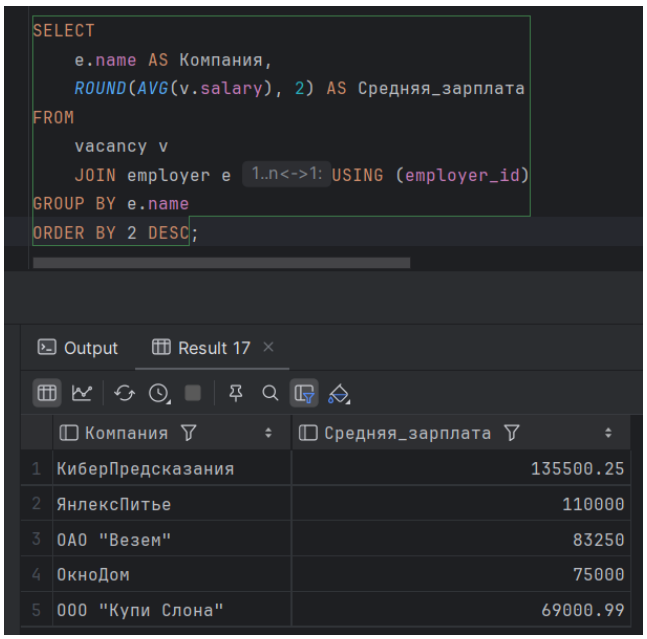

Рисунок 7 – Выполнение запроса 5 выборки в DataGrid

6. Сколько работников ищут работу, имея высшее образование?

При помощи JOIN … USING соединены таблицы resume и worker (через worker_id) и worker_education (через worker_id) и education (через education_id). Выводится количество уникальных записей о работниках, у которых созданы вакансии, а так же имеется тип образования ‘Высшее’.

На рис.8 представлен вывод выполнения запроса 6 в СУБД PostgreSQL DataGrid.

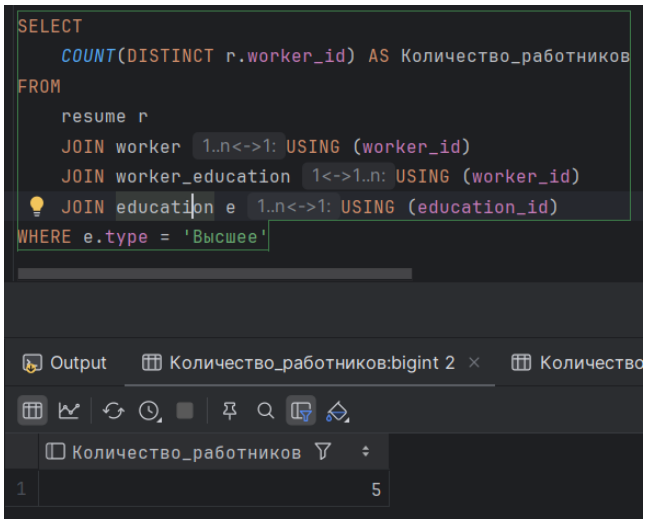

Рисунок 8 – Выполнение запроса 6 выборки в DataGrid

7. Сколько работников имело более 3-х мест работы?

При помощи JOIN … USING соединены таблицы worker и worker_workbook (через worker_id) и workbook (через workbook_id) и

worker_employer_writing (через workbook_id). Выводится количество уникальных записей о работниках, у которых имеются записи в записных книжках от более чем 3 уникальных работодателей.

На рис.9 представлен вывод выполнения запроса 7 в СУБД PostgreSQL DataGrid.

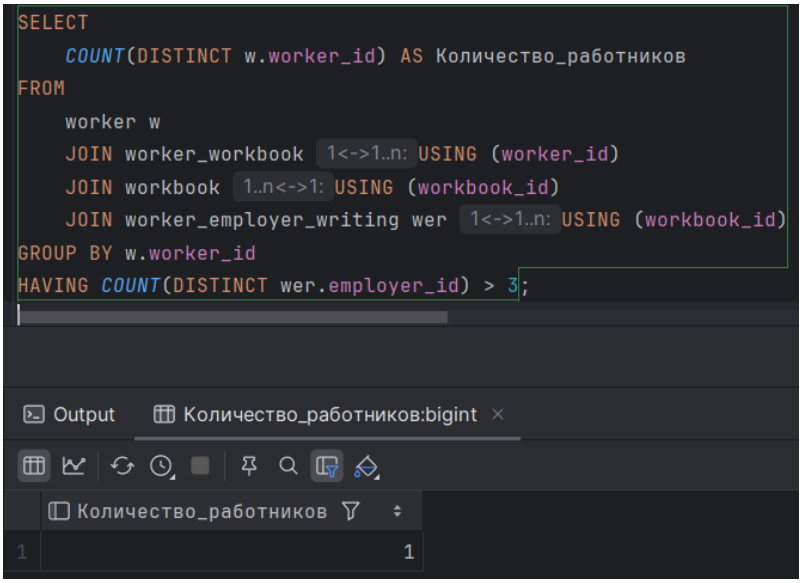

Рисунок 9 – Выполнение запроса 7 выборки в DataGrid

8. Какие вакансии имеют ЗП более 100 000р и не требуют опыта работы?

При помощи JOIN … USING соединены таблицы vacancy и employer (через employer_id) и post (через post_id). Выводятся записи о вакансиях, которые имеют зарплату более 100000 и нулевой опыт работы.

На рис.10 представлен вывод выполнения запроса 8 в СУБД PostgreSQL DataGrid.

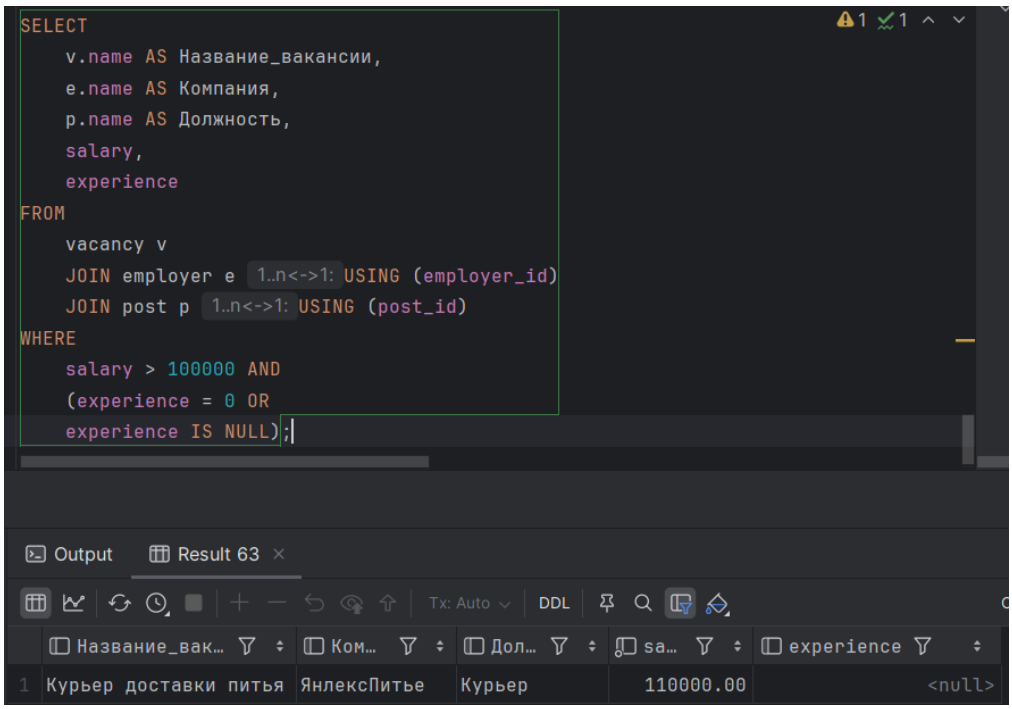

Рисунок 10 – Выполнение запроса 8 выборки в DataGrid

Исходный код см. Приложение А.

Ссылка на pr и db-fuddle см. Приложение Б.

Выводы.

Были изучены опробованы на практике навыки работы с СУБД PostgreSQL путем создания таблиц, их заполнения и запросов их данных через SQL-запросы.

ПРИЛОЖЕНИЕ А
ИСХОДНЫЙ КОД

Файл init.sql:

```
CREATE TABLE passport
(
    passport_id     SERIAL PRIMARY KEY,
    passport_number INT NOT NULL,
    passport_series INT NOT NULL,
    UNIQUE (passport_number, passport_series)
);

CREATE TABLE tin
(
    tin_id SERIAL PRIMARY KEY,
    tin    BIGINT UNIQUE NOT NULL
);

CREATE TABLE workbook
(
    workbook_id     SERIAL PRIMARY KEY,
    workbook_number INT UNIQUE NOT NULL,
    isue_date       DATE
);

CREATE TABLE worker
(
    worker_id   SERIAL PRIMARY KEY,
    passport_id INT  NOT NULL,
    tin_id      INT  NOT NULL,
    birthday    DATE NOT NULL,
    FOREIGN KEY (passport_id) REFERENCES passport (passport_id),
    FOREIGN KEY (tin_id) REFERENCES tin (tin_id)
);

CREATE TABLE worker_workbook
(
    worker_id   INT NOT NULL,
    workbook_id INT NOT NULL,
    PRIMARY KEY (worker_id, workbook_id),
    FOREIGN KEY (worker_id) REFERENCES worker (worker_id),
    FOREIGN KEY (workbook_id) REFERENCES workbook (workbook_id)
);

CREATE TYPE education_type AS ENUM
    (
        'Высшее',
        'Среднее'
        );

CREATE TABLE education
(
    education_id SERIAL PRIMARY KEY,
    name         VARCHAR(255)   NOT NULL,
    type         education_type NOT NULL
);

CREATE TABLE worker_education
(
    worker_id    INT  NOT NULL,
    education_id INT  NOT NULL,
```

```
    enroll_date  DATE NOT NULL,
    PRIMARY KEY (worker_id, education_id),
    FOREIGN KEY (worker_id) REFERENCES worker (worker_id),
    FOREIGN KEY (education_id) REFERENCES education (education_id)
);

CREATE TABLE post
(
    post_id SERIAL PRIMARY KEY,
    name    VARCHAR(255) UNIQUE NOT NULL
);

CREATE TABLE resume
(
    resume_id SERIAL PRIMARY KEY,
    worker_id INT NOT NULL,
    post_id   INT NOT NULL,
    salary    DECIMAL(8, 2),
    FOREIGN KEY (worker_id) REFERENCES worker (worker_id),
    FOREIGN KEY (post_id) REFERENCES post (post_id)
);

CREATE TABLE skill
(
    skill_id    SERIAL PRIMARY KEY,
    description VARCHAR(255) UNIQUE NOT NULL
);

CREATE TABLE resume_skill
(
    resume_id INT NOT NULL,
    skill_id  INT NOT NULL,
    PRIMARY KEY (resume_id, skill_id),
    FOREIGN KEY (resume_id) REFERENCES resume (resume_id),
    FOREIGN KEY (skill_id) REFERENCES skill (skill_id)
);

CREATE TABLE field_of_activity
(
    activity_id SERIAL PRIMARY KEY,
    name        VARCHAR(255) UNIQUE NOT NULL
);

CREATE TABLE employer
(
    employer_id SERIAL PRIMARY KEY,
    activity_id INT          NOT NULL,
    name        VARCHAR(255) NOT NULL,
    photo_url   VARCHAR(255),
    description VARCHAR(255),
    FOREIGN KEY (activity_id) REFERENCES field_of_activity (activity_id)
);

CREATE TABLE presentation_file
(
    file_id     SERIAL PRIMARY KEY,
    employer_id INT          NOT NULL,
    name        VARCHAR(255) NOT NULL,
    url         VARCHAR(255) NOT NULL,
    FOREIGN KEY (employer_id) REFERENCES employer (employer_id)
);

CREATE TYPE writing_type AS ENUM
```

```
    (
        'Зачисление',
        'Увольнение',
        'Переход',
        'Изменение должности'
        );

CREATE TABLE workbook_writing
(
    writing_id  SERIAL PRIMARY KEY,
    type        writing_type NOT NULL,
    date        DATE         NOT NULL,
    description VARCHAR(255)
);

CREATE TABLE worker_employer_writing
(
    writing_id  INT NOT NULL,
    workbook_id INT NOT NULL,
    employer_id INT NOT NULL,
    PRIMARY KEY (writing_id, workbook_id, employer_id),
    FOREIGN KEY (writing_id) REFERENCES workbook_writing (writing_id),
    FOREIGN KEY (workbook_id) REFERENCES workbook (workbook_id),
    FOREIGN KEY (employer_id) REFERENCES employer (employer_id)
);

CREATE TYPE vacancy_status AS ENUM
    (
        'Активна',
        'В архиве',
        'Удалена'
        );

CREATE TABLE vacancy
(
    vacancy_id  SERIAL PRIMARY KEY,
    post_id     INT            NOT NULL,
    employer_id INT            NOT NULL,
    name        VARCHAR(255)   NOT NULL,
    address     VARCHAR(255)   NOT NULL,
    salary      DECIMAL(8, 2)  NOT NULL,
    experience  INT,
    comment     VARCHAR(255),
    status      vacancy_status NOT NULL,
    FOREIGN KEY (post_id) REFERENCES post (post_id),
    FOREIGN KEY (employer_id) REFERENCES employer (employer_id)
);

CREATE TABLE requirement
(
    requirement_id SERIAL PRIMARY KEY,
    description    VARCHAR(255) UNIQUE NOT NULL
);

CREATE TABLE vacancy_requirement
(
    vacancy_id     INT NOT NULL,
    requirement_id INT NOT NULL,
    PRIMARY KEY (vacancy_id, requirement_id),
    FOREIGN KEY (vacancy_id) REFERENCES vacancy (vacancy_id),
    FOREIGN KEY (requirement_id) REFERENCES requirement (requirement_id)
);
```

```
CREATE TABLE condition
(
    condition_id SERIAL PRIMARY KEY,
    description  VARCHAR(255) UNIQUE NOT NULL
);

CREATE TABLE vacancy_condition
(
    vacancy_id   INT NOT NULL,
    condition_id INT NOT NULL,
    PRIMARY KEY (vacancy_id, condition_id),
    FOREIGN KEY (vacancy_id) REFERENCES vacancy (vacancy_id),
    FOREIGN KEY (condition_id) REFERENCES condition (condition_id)
);
```

Файл insert.sql:

```
INSERT INTO passport (passport_number, passport_series)
VALUES
    (3317, 627219),
    (3214, 745323),
    (3421, 231421),
    (3142, 892156),
    (3345, 324561),
    (3214, 234741);

INSERT INTO tin (tin)
VALUES
    (505021345692),
    (432102345672),
    (476221556514),
    (813267892103),
    (456969651393),
    (158923546790);

INSERT INTO worker (passport_id, tin_id, birthday)
VALUES
    (2, 3,'1992-02-14'),
    (1, 4, '1996-04-19'),
    (5, 6, '1982-10-01'),
    (6, 1, '2003-01-27'),
    (3, 5, '1979-07-08'),
    (4, 2, '2000-01-01');

INSERT INTO education (name, type)
VALUES
    ('СПБГЭТУ', 'Высшее'),
    ('МАИ', 'Высшее'),
    ('КЛПК', 'Среднее'),
    ('ВятГУ', 'Высшее'),
    ('ИТМО', 'Высшее'),
    ('РГГУ', 'Среднее'),
    ('КПиАС', 'Среднее'),
    ('СПБГУГА', 'Высшее');

INSERT INTO worker_education (worker_id, education_id, enroll_date)
VALUES
    (1, 3, '2010-09-01'),
    (1, 1, '2015-08-25'),
    (3, 4, '2012-07-29'),
    (5, 8, '2000-08-27'),
    (5, 5, '1992-08-28'),
```

```
    (4, 2, '2019-09-01'),
    (6, 1, '2018-08-23');


INSERT INTO workbook (workbook_number, isue_date)
VALUES
    (8627821, '2006-12-14'),
    (5423109, '2001-03-27'),
    (1234901, '1997-09-02'),
    (4298134, '2015-02-13'),
    (9823518, '2020-07-09'),
    (7349821, '2025-01-01'),
    (2354890, '2017-09-18');

INSERT INTO worker_workbook (worker_id, workbook_id)
VALUES
    (1, 1),
    (3, 2),
    (6, 5),
    (5, 3),
    (4, 6),
    (5, 4),
    (2, 7);

INSERT INTO post (name)
VALUES
    ('Программист на python'),
    ('Курьер'),
    ('Складовщик'),
    ('HR-менеджер'),
    ('Кассир'),
    ('Водитель автобуса'),
    ('Маркетолог'),
    ('Data-Scientist'),
    ('Парикмахер'),
    ('Разнорабочий');

INSERT INTO skill (description)
VALUES
    ('Умение работать в команде'),
    ('Водительская категория D'),
    ('Программирование на python'),
    ('Мат. статистика'),
    ('Трудолюбие'),
    ('Терпеливость'),
    ('Быстрая адаптация к изменениям'),
    ('Умение работать в стрессовой обстановке'),
    ('Спортивное физическое телосложение');

INSERT INTO resume (worker_id, post_id, salary)
VALUES
    (1, 4, 75000),
    (5, 6, 57500.50),
    (3, 3, 70000),
    (3, 10, 47000.99),
    (4, 8, 250000),
    (6, 1, 150000),
    (5, 2, 180000);

INSERT INTO resume_skill (resume_id, skill_id)
VALUES
    (1, 7),
    (1, 8),
```

```
        (1, 1),
        (5, 4),
        (6, 3),
        (7, 5),
        (7, 7),
        (2, 2),
        (3, 9),
        (3, 5);

    INSERT INTO field_of_activity (name)
    VALUES
        ('Перевозка пассажиров'),
        ('IT'),
        ('Доставка'),
        ('Продажи'),
        ('Производство продукции');

    INSERT INTO employer (activity_id, name, photo_url, description)
    VALUES
        (2, 'КиберПредсказания', 'https://photo1', 'Компания, занимающаяся
аналитикой данных, от сторонних организаций'),
        (3, 'ЯнлексПитье', 'https://proho2', 'Слежба доставки питья на дом'),
        (1, 'ОАО "Везем"', 'https://prohot3', 'Организация, осуществляющая
перевозку пассажиров по маршрутам дальнего следования'),
        (5, 'SuperFlowers', 'https://photo4', 'Производство блоков питания'),
        (4, 'ООО "Купи Слона"', 'https://photo5', 'Зоомагазин'),
        (5, 'ОкноДом', 'https://photo6', 'Отечественный производитель
стеклопакетов');

    INSERT INTO presentation_file (employer_id, name, url)
    VALUES
        (1, 'Презентация 1', 'https://file1'),
        (1, 'Презентация 2', 'https://file2'),
        (2, 'Презентация 1', 'https://file3'),
        (5, 'Презентация 1', 'https://file4'),
        (4, 'Презентация 1', 'https://file5');

    INSERT INTO requirement (description)
    VALUES
        ('Стрессоустойчивость'),
        ('Знание ПДД'),
        ('Знание python на базовом уровне'),
        ('Знание python на повышенном уровне'),
        ('Знание мат. статистики'),
        ('Работа в команде'),
        ('Хорошо поставленная речь'),
        ('Наличие автомобиля'),
        ('Работа из офиса'),
        ('Работа из дома');

    INSERT INTO condition (description)
    VALUES
        ('Обустроенное рабочее место'),
        ('Предоставление ноутбука'),
        ('ДМС'),
        ('Гибкий рабочий график'),
        ('График работы 2 на 2'),
        ('Работа в большой компании'),
        ('Официальное трудоустройство'),
        ('Рабочий автомобиль от компании');

    INSERT INTO vacancy (post_id, employer_id, name, salary, address,
experience, comment, status)
```

```
VALUES
    (8, 1, 'DataScientist для Предсказаний', 180000.50, 'Варшавская улица, 63к1, Санкт-Петербург', 5, 'Требуется DataScientist для предсказаний кибер будущего', 'Активна'),
    (6, 3, 'Водитель рейсового автобуса', 83250, 'улица Костюшко, 17, Санкт-Петербург', 3, NULL, 'В архиве'),
    (4, 1, 'Менеджер по потбору кадров (HR)', 91000, 'Варшавская улица, 44, Санкт-Петербург', 7, 'Нужен менеджер для подбора персонала в новую команду', 'Активна'),
    (2, 2, 'Курьер доставки питья', 110000, 'Варшавская улица, 124, Санкт-Петербург', NULL, NULL, 'Активна'),
    (3, 6, 'Начальник склада', 87000, 'Софийская улица, 8к5с4, Санкт-Петербург', 5, 'Начальник-заведущий складом, управляет приемом и отправкой товара', 'В архиве'),
    (3, 6, 'Складовщий на новый склад', 63000, 'Софийская улица, 8к5с4, Санкт-Петербург', NULL, NULL, 'Активна'),
    (7, 5, 'Маркетолог бренда', 69000.99, 'Московский проспект, 9, Санкт-Петербург', NULL, 'Ждем креативных людей для продвижения бренда!', 'Активна');

INSERT INTO vacancy_requirement (vacancy_id, requirement_id)
VALUES
    (1, 3),
    (1, 4),
    (3, 6),
    (3, 7),
    (7, 9),
    (4, 8),
    (5, 1),
    (5, 7),
    (2, 2),
    (1, 10);

INSERT INTO vacancy_condition (vacancy_id, condition_id)
VALUES
    (1, 2),
    (1, 4),
    (2, 5),
    (2, 8),
    (6, 3),
    (7, 6),
    (4, 7),
    (5, 1);

INSERT INTO workbook_writing (type, date, description)
VALUES
    ('Зачисление', '2010-01-01', NULL),
    ('Изменение должности', '2018-02-13', 'Повышение должности'),
    ('Зачисление', '1995-07-01', NULL),
    ('Увольнение', '1995-11-18', NULL),
    ('Зачисление', '1995-12-01', NULL),
    ('Увольнение', '1997-05-10', NULL),
    ('Зачисление', '1998-01-13', NULL),
    ('Увольнение', '2003-03-08', NULL),
    ('Зачисление', '2004-05-21', NULL),
    ('Зачисление', '2023-07-18', NULL);


INSERT INTO worker_employer_writing (writing_id, workbook_id, employer_id)
VALUES
    (1, 1, 6),
    (2, 1, 6),
    (4, 4, 2),
    (3, 4, 2),
```

```
    (10, 6, 5),
    (5, 4, 4),
    (6, 4, 4),
    (8, 3, 3),
    (7, 3, 3),
    (9, 3, 1);
```

Файл query.sql:

```
SELECT
    e.name AS Компания,
    v.name AS Название_вакансии,
    p.name AS Должность,
    salary,
    address,
    experience,
    comment,
    status
FROM
    vacancy v
    JOIN employer e USING (employer_id)
    JOIN post p USING (post_id)
WHERE e.name = 'КиберПредсказания';

SELECT
    v.name AS Название_вакансии,
    e.name AS Комания,
    p.name AS Должность,
    salary
FROM
    vacancy v
    JOIN employer e USING (employer_id)
    JOIN post p USING (post_id)
WHERE v.name LIKE '%склад%';

SELECT
    COUNT(*) AS Количество_вакансий
FROM vacancy
WHERE address LIKE '%Варшавская%';

SELECT
    v.name AS Название_вакансии,
    e.name AS Компания,
    p.name AS Должность,
    salary,
    status
FROM
    vacancy v
    JOIN employer e USING (employer_id)
    JOIN post p USING (post_id)
WHERE
    status = 'В архиве' AND
    e.name = 'ОАО "Везем"';

SELECT
    e.name AS Компания,
    ROUND(AVG(v.salary), 2) AS Средняя_зарплата
FROM
    vacancy v
    JOIN employer e USING (employer_id)
GROUP BY e.name
ORDER BY 2 DESC;
```

```
SELECT
    COUNT(DISTINCT r.worker_id) AS Количество_работников
FROM
    resume r
    JOIN worker USING (worker_id)
    JOIN worker_education USING (worker_id)
    JOIN education e USING (education_id)
WHERE e.type = 'Высшее';

SELECT
    COUNT(DISTINCT w.worker_id) AS Количество_работников
FROM
    worker w
    JOIN worker_workbook USING (worker_id)
    JOIN workbook USING (workbook_id)
    JOIN worker_employer_writing wer USING (workbook_id)
GROUP BY w.worker_id
HAVING COUNT(DISTINCT wer.employer_id) > 3;

SELECT
    v.name AS Название_вакансии,
    e.name AS Компания,
    p.name AS Должность,
    salary,
    experience
FROM
    vacancy v
    JOIN employer e USING (employer_id)
    JOIN post p USING (post_id)
WHERE
    salary > 100000 AND
    (experience = 0 OR
    experience IS NULL);
```

ПРИЛОЖЕНИЕ Б
ССЫЛКИ

Ссылка на Pull-Request: https://github.com/moevm/sql-2025-3384/pull/23

Ссылка на db-fiddle: https://www.db-fiddle.com/f/vtSApk3cQsN8NzJQuUYo75/0