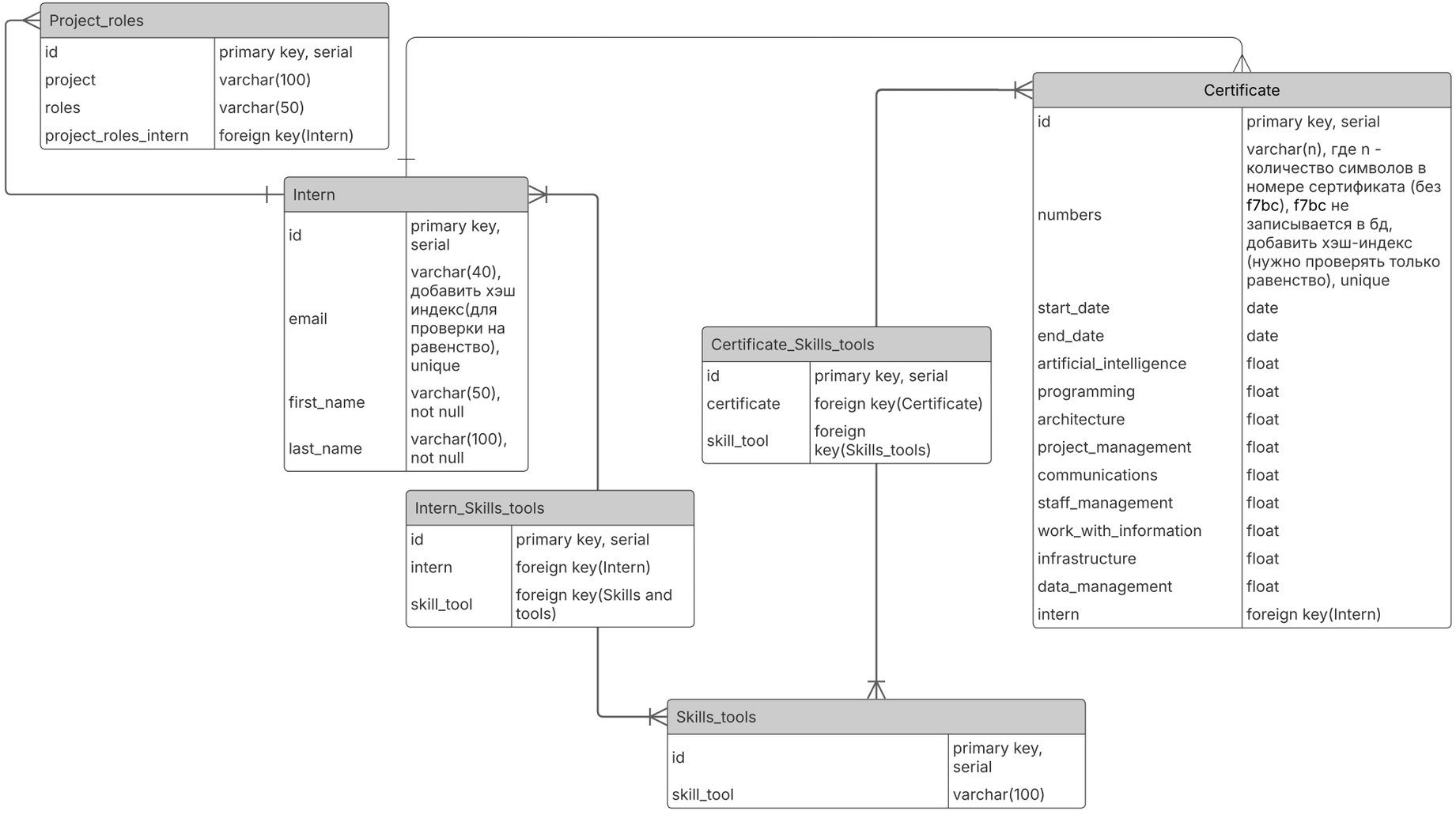
Тестовое задание для Python-программистов

Представьте, что вы участвуете в разработке сервиса генерации сертификатов.
Ваша задача — спроектировать структуру базы данных для хранения информации о сертификате.
Проектирование осуществляется на основе макета сертификата:
https://cloveri.com/certificate/f7bcd817-42f9-e2ca-0226-f6e916ce6b73
Менеджер продукта вам сказал, что обычно сертификаты ищут по номеру сертификата (f7bc...) или по электронному адресу почты владельца сертификата (того, кто его получил).

Задача:
1. Пришлите описание используемых таблиц, включая информацию о названиях и формате полей.
2. Предложите индексы для каждой из таблиц, которые помогут быстрее выдавать информацию о сертификате
3. Напишите SQL-запрос, который выведет все сертификаты, для которых не заданы навыки, которые есть у владельца сертификата
4. Какой фреймворк вы бы выбрали для реализации и почему?

Решение:

1.



Код для создания таблиц:

```
CREATE TABLE Intern (
    id SERIAL PRIMARY KEY,
    email VARCHAR(40) NOT NULL UNIQUE,
    first_name VARCHAR(50) NOT NULL,
    last_name VARCHAR(100) NOT NULL
);

CREATE TABLE Certificate (
    id SERIAL PRIMARY KEY,
    numbers VARCHAR(10) NOT NULL UNIQUE,
    start_date DATE,
    end_date DATE,
    artifical_intelligence FLOAT CHECK(artifical_intelligence >0 AND artifical_intelligence <= 10),
    programming FLOAT CHECK(programming >0 AND programming <= 10),
    architecture FLOAT CHECK(architecture >0 AND architecture <= 10),
    project_management FLOAT CHECK(project_management >0 AND project_management <= 10),
    communications FLOAT CHECK(communications >0 AND communications <= 10),
    staff_management FLOAT CHECK(staff_management >0 AND staff_management <= 10),
    work_with_information FLOAT CHECK(work_with_information >0 AND work_with_information <= 10),
    infrastructure FLOAT CHECK(infrastructure >0 AND infrastructure <= 10),
    data_management FLOAT CHECK(data_management >0 AND data_management <= 10),
    intern INT,
    FOREIGN KEY (intern) REFERENCES Intern(id) ON DELETE CASCADE
);

CREATE TABLE Project_roles (
    id SERIAL PRIMARY KEY,
    project VARCHAR(100),
    roles VARCHAR(50),
    project_roles_intern INT,
    FOREIGN KEY (project_roles_intern) REFERENCES Intern(id) ON DELETE CASCADE
);

CREATE TABLE Skills_tools (
    id SERIAL PRIMARY KEY,
    skill_tool VARCHAR(100)
);

CREATE TABLE Intern_Skills_tools (
    id SERIAL PRIMARY KEY,
    intern INT,
    skill_tool INT,
    FOREIGN KEY (intern) REFERENCES Intern(id) ON DELETE CASCADE,
    FOREIGN KEY (skill_tool) REFERENCES Skills_tools(id) ON DELETE CASCADE
);

CREATE TABLE Certificate_Skills_tools (
    id SERIAL PRIMARY KEY,
    certificate INT,
    skill_tool INT,
    FOREIGN KEY (certificate) REFERENCES Certificate(id) ON DELETE CASCADE,
    FOREIGN KEY (skill_tool) REFERENCES Skills_tools(id) ON DELETE CASCADE
);
```

Код для заполнения таблиц:

```
INSERT INTO Intern(email, first_name, last_name)
VALUES
    ('example_1@example.com', 'Name_1', 'Lastname_1'),
    ('example_2@example.com', 'Name_2', 'Lastname_2'),
    ('example_3@example.com', 'Name_3', 'Lastname_3');

INSERT INTO Certificate(start_date, end_date, numbers, artifical_intelligence, programming, architecture, project_management, communications, staff_management, work_with_information, infrastructure, data_management, intern)
VALUES
    ('2024-01-01', '2024-07-01', '1234567890', '5.5', '6', '6.5', '7', '7.5', '8', '8.5', '9', '9.5', '1'),
    ('2024-02-02', '2024-08-02', '2345678901', '5', '5.5', '6', '6.5', '7', '7.5', '8', '8.5', '9', '2'),
    ('2024-03-03', '2024-09-03', '3456789012', '4.5', '5', '5.5', '6', '6.5', '7', '7.5', '8', '8.5', '3');

INSERT INTO Project_roles(project, roles, project_roles_intern)
VALUES
    ('Project_1', 'Role_1', '1'),
    ('Project_1', 'Role_2', '2'),
    ('Project_2', 'Role_1', '3'),
    ('Project_2', 'Role_2', '2');

INSERT INTO Skills_tools(skill_tool)
VALUES
    ('Skill_tool_1'),
    ('Skill_tool_2'),
    ('Skill_tool_3'),
    ('Skill_tool_4'),
    ('Skill_tool_5'),
    ('Skill_tool_6');

INSERT INTO Certificate_Skills_tools(certificate, skill_tool)
VALUES
    ('1', '1'),
    ('1', '2'),
    ('1', '3'),
    ('2', '2'),
    ('2', '3'),
    ('3', '3');

INSERT INTO Intern_Skills_tools(intern, skill_tool)
VALUES
    ('1', '1'),
    ('1', '2'),
    ('1', '3'),
    ('1', '4'),
    ('1', '5'),
    ('1', '6'),
    ('2', '2'),
    ('2', '3'),
    ('2', '4'),
    ('3', '3');
```

2.

Код для создания индексов:

```
CREATE INDEX idx_hash_index_numbers ON Certificate USING HASH (numbers);

CREATE INDEX idx_hash_index_email ON Intern USING HASH (email);
```

Я выбрала хэш-индекс, т.к обычно сертификаты ищут по номеру сертификата или по электронному адресу почты владельца сертификата, для которых будет достаточно сравнения на равенство. Кроме того, считаю ненужным хранить повторяющиеся символы в номере сертификата (f7bc) в базе данных.

3.

SQL-запрос, который выведет все сертификаты, для которых не заданы навыки, которые есть у владельца сертификата:

```
SELECT DISTINCT Cert.numbers FROM Certificate Cert JOIN intern_skills_tools i_s_t ON Cert.intern = i_s_t.intern JOIN certificate_skills_tools c_s_t ON Cert.id = c_s_t.certificate WHERE c_s_t.skill_tool != i_s_t.skill_tool;
```

4.

Мне нравится Django, тк в нем множество библиотек и модулей (django-allauth, django-filter), дженерики (ListView, DetailView), но он больше подходит для крупных проектов. В связи с этим, я выбрала бы FastApi, как знакомый мне фреймворк, который подойдет для небольшого сервиса.