ГУАП

КАФЕДРА № 43

ОТЧЕТ   
ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Старший преподаватель |  |  |  | Н.В. Путилова |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

|  |
| --- |
| ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ №9  Объектно-реляционные базы данных. Проектирование и создание |
| по дисциплине: [Проектирование](https://pro.guap.ru/inside/students/subjects/3154777) баз данных |
|  |
|  |

РАБОТУ ВЫПОЛНИЛ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| СТУДЕНТ ГР. | 4134к |  |  |  | Шумков Д.В. |
|  |  |  | подпись, дата |  | инициалы, фамилия |

Санкт-Петербург

2024

25. Багтрекинг: проекты, баги, фичи (новые функции), тестировщики, разработчики (разные статусы людей (исправление и добавление багов), тестировщики не исправляют)

а. Баги, в названии которых содержится слово «размер», но оно не последнее

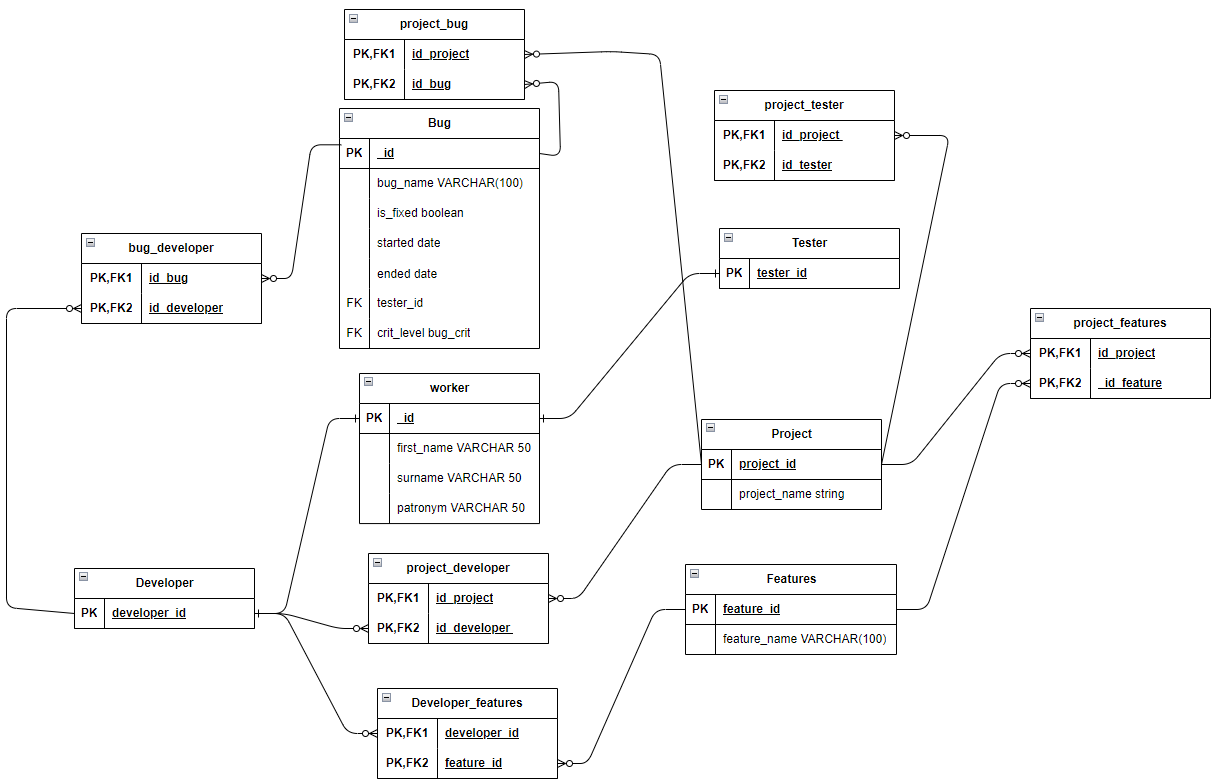
б. Функции(фичи), относящаяся к 2 различным проектам

в. Критический баг, который добавили последним

г. Тестировщик, с самым большим количеством добавленных багов

д. Разработчик, у которого нет незавершенных задач(реализация фичи или исправление бага)

Физическая модель БД



Скрипт создания БД

CREATE TABLE worker (

worker\_id SERIAL PRIMARY KEY,

first\_name VARCHAR(50) NOT NULL,

surname VARCHAR(50) NOT NULL,

patronym VARCHAR(50) NOT NULL

);

CREATE TABLE tester (

tester\_id SERIAL PRIMARY KEY

) INHERITS (worker);

CREATE TABLE developer (

developer\_id SERIAL PRIMARY KEY

) INHERITS (worker);

CREATE TABLE project (

project\_id SERIAL PRIMARY KEY,

project\_name VARCHAR(100) NOT NULL

);

CREATE TABLE features (

feature\_id SERIAL PRIMARY KEY,

feature\_name VARCHAR(100) NOT NULL

);

CREATE TYPE bug\_crit AS ENUM ('Некритичный', 'Критичный', 'Очень критичный');

CREATE TABLE bugs (

bug\_id SERIAL PRIMARY KEY,

bug\_name VARCHAR(100) NOT NULL,

is\_fixed BOOL NOT NULL,

started DATE NOT NULL,

ended DATE,

tester\_id INT NOT NULL,

crit\_level bug\_crit NOT NULL,

FOREIGN KEY (tester\_id) REFERENCES tester(tester\_id) ON DELETE RESTRICT ON UPDATE CASCADE

);

CREATE TABLE project\_bug (

project\_id INT,

bug\_id INT,

PRIMARY KEY (project\_id, bug\_id),

FOREIGN KEY (project\_id) REFERENCES project(project\_id) ON DELETE RESTRICT ON UPDATE CASCADE,

FOREIGN KEY (bug\_id) REFERENCES bugs(bug\_id) ON DELETE RESTRICT ON UPDATE CASCADE

);

CREATE TABLE bug\_developer (

bug\_id INT,

developer\_id INT,

PRIMARY KEY (bug\_id, developer\_id),

FOREIGN KEY (developer\_id) REFERENCES developer(developer\_id) ON DELETE RESTRICT ON UPDATE CASCADE,

FOREIGN KEY (bug\_id) REFERENCES bugs(bug\_id) ON DELETE RESTRICT ON UPDATE CASCADE

);

CREATE TABLE project\_developer (

project\_id INT,

developer\_id INT,

PRIMARY KEY (project\_id, developer\_id),

FOREIGN KEY (project\_id) REFERENCES project(project\_id) ON DELETE RESTRICT ON UPDATE CASCADE,

FOREIGN KEY (developer\_id) REFERENCES developer(developer\_id) ON DELETE RESTRICT ON UPDATE CASCADE

);

CREATE TABLE project\_features (

project\_id INT,

feature\_id INT,

PRIMARY KEY (project\_id,feature\_id),

FOREIGN KEY (project\_id) REFERENCES project(project\_id) ON DELETE RESTRICT ON UPDATE CASCADE,

FOREIGN KEY (feature\_id) REFERENCES features(feature\_id) ON DELETE RESTRICT ON UPDATE CASCADE

);

CREATE TABLE developer\_features (

developer\_id INT,

feature\_id INT,

PRIMARY KEY (developer\_id, feature\_id),

FOREIGN KEY (developer\_id) REFERENCES developer(developer\_id) ON DELETE RESTRICT ON UPDATE CASCADE,

FOREIGN KEY (feature\_id) REFERENCES features(feature\_id) ON DELETE RESTRICT ON UPDATE CASCADE

);