|  |
| --- |
| Объектно-реляционные базы данных. Проектирование и создание |

1. **Вариант задания (12 вариант)**

Вакансии: волонтерские позиции, название вакансии, организация работодатель, адрес работодателя, диапазон зарплаты, требования к образованию, Обязанности, график работы, требования обязательные, желательные, дата выставления вакансии.

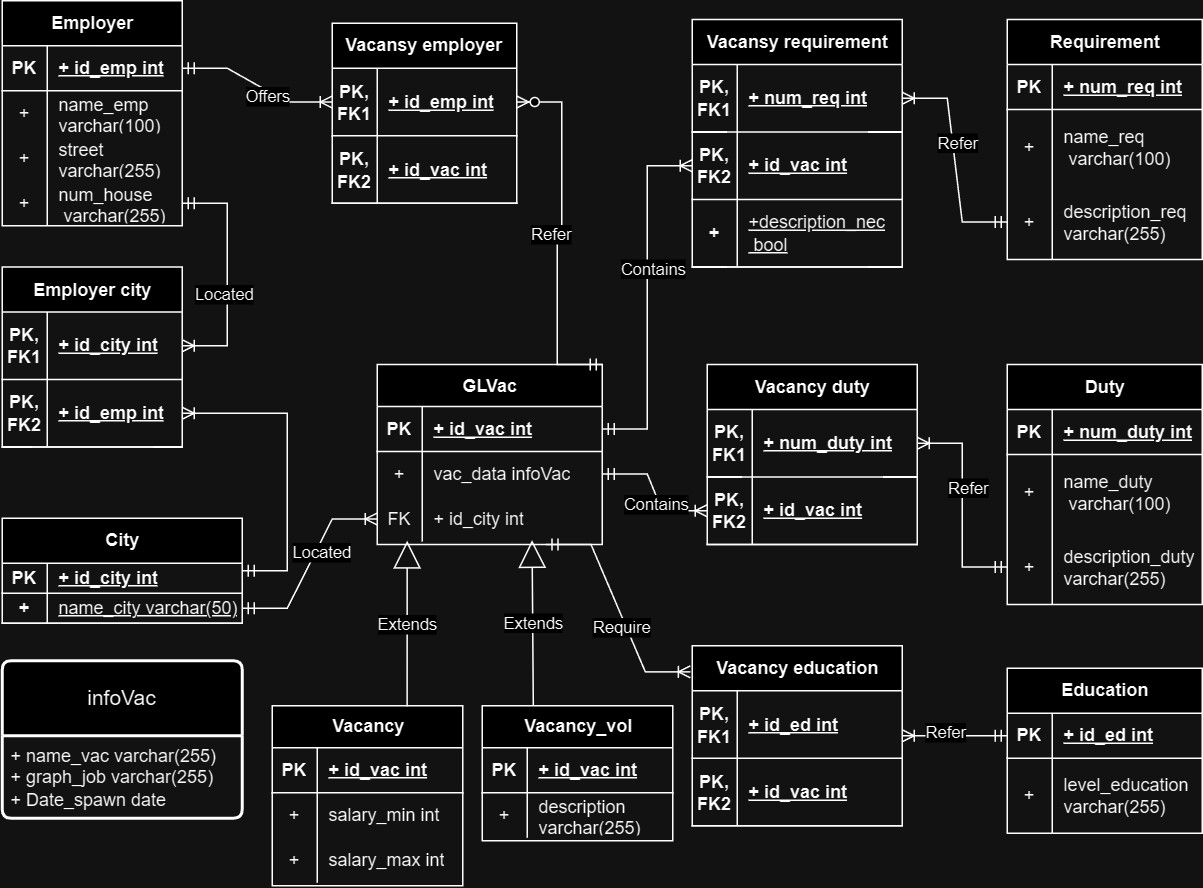
а. вакансии, имеющие в названии SQL, но не заканчивающиеся на него

б. работодатели в Санкт-Петербурге, выставившие несколько вакансий

в. вакансия с наибольшей зарплатой

г. волонтерские позиции с максимальным количеством требований

д. вакансии, в которых нет требования к опыту работы

2. **Физическая модель базы данных:** 

1. **Созданная модель базы данных:**

CREATE TYPE infoVac AS (

name\_vac VARCHAR(255),

graph\_job VARCHAR(255),

date\_spawn DATE

);

CREATE TABLE IF NOT EXISTS Employer(

id\_emp SERIAL NOT NULL PRIMARY KEY,

name\_emp VARCHAR(100) NOT NULL,

street VARCHAR(255) NOT NULL,

num\_house VARCHAR(255) NOT NULL

);

CREATE TABLE IF NOT EXISTS City(

id\_city SERIAL NOT NULL PRIMARY KEY,

name\_city VARCHAR(50) NOT NULL

);

CREATE TABLE IF NOT EXISTS Employer\_city(

id\_city INT NOT NULL,

id\_emp INT NOT NULL,

FOREIGN KEY (id\_city) REFERENCES City(id\_city) ON DELETE CASCADE ON UPDATE RESTRICT,

PRIMARY KEY (id\_city, id\_emp)

);

CREATE TABLE IF NOT EXISTS GLVac(

id\_vac SERIAL NOT NULL,

vac\_data infoVac NOT NULL,

id\_city INT NOT NULL,

FOREIGN KEY (id\_city) REFERENCES City(id\_city) ON DELETE CASCADE ON UPDATE RESTRICT,

PRIMARY KEY (id\_vac)

);

CREATE TABLE IF NOT EXISTS Vacancy(

salary\_min INT,

salary\_max INT,

FOREIGN KEY (id\_city) REFERENCES City(id\_city) ON DELETE CASCADE ON UPDATE RESTRICT,

PRIMARY KEY (id\_vac)

) INHERITS (GLVac);

CREATE TABLE IF NOT EXISTS Vacancy\_vol(

description varchar(255) NOT NULL,

FOREIGN KEY (id\_city) REFERENCES City(id\_city) ON DELETE CASCADE ON UPDATE RESTRICT,

PRIMARY KEY (id\_vac)

) INHERITS (GLVac);

CREATE TABLE IF NOT EXISTS Vacancy\_employer(

id\_emp INT NOT NULL,

id\_vac INT NOT NULL,

FOREIGN KEY (id\_emp) REFERENCES Employer(id\_emp) ON DELETE CASCADE ON UPDATE RESTRICT,

PRIMARY KEY (id\_emp, id\_vac)

);

CREATE TABLE IF NOT EXISTS Requirement(

num\_req SERIAL NOT NULL,

name\_req VARCHAR(100) NOT NULL,

description\_req VARCHAR(255) NOT NULL,

PRIMARY KEY (num\_req)

);

CREATE TABLE IF NOT EXISTS Vacancy\_requirement(

num\_req INT NOT NULL,

id\_vac INT NOT NULL,

description\_nec BOOLEAN NOT NULL,

FOREIGN KEY (num\_req) REFERENCES Requirement(num\_req) ON DELETE CASCADE ON UPDATE RESTRICT,

PRIMARY KEY (num\_req, id\_vac)

);

CREATE TABLE IF NOT EXISTS Duty(

num\_duty SERIAL NOT NULL,

name\_duty VARCHAR(100) NOT NULL,

description\_duty VARCHAR(255) NOT NULL,

PRIMARY KEY (num\_duty)

);

CREATE TABLE IF NOT EXISTS Vacancy\_duty(

num\_duty INT NOT NULL,

id\_vac INT NOT NULL,

FOREIGN KEY (num\_duty) REFERENCES Duty(num\_duty) ON DELETE CASCADE ON UPDATE RESTRICT,

PRIMARY KEY (num\_duty, id\_vac)

);

CREATE TABLE IF NOT EXISTS Education(

id\_ed SERIAL NOT NULL,

level\_education VARCHAR(255) NOT NULL,

PRIMARY KEY (id\_ed)

);

CREATE TABLE IF NOT EXISTS Vacancy\_education(

id\_ed INT NOT NULL,

id\_vac INT NOT NULL,

FOREIGN KEY (id\_ed) REFERENCES Education(id\_ed) ON DELETE CASCADE ON UPDATE RESTRICT,

PRIMARY KEY (id\_ed, id\_vac)

);

--\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

CREATE OR REPLACE FUNCTION check\_VacReqVacEdVacDutyVacEmp()

RETURNS TRIGGER AS $$

BEGIN

UPDATE Vacancy\_requirement SET id\_vac = NEW.id\_vac

WHERE id\_vac=old.id\_vac;

UPDATE Vacancy\_education SET id\_vac = NEW.id\_vac

WHERE id\_vac=old.id\_vac;

UPDATE Vacancy\_duty SET id\_vac = NEW.id\_vac

WHERE id\_vac=old.id\_vac;

UPDATE Vacancy\_employer SET id\_vac = NEW.id\_vac

WHERE id\_vac=old.id\_vac;

RETURN NEW;

END;

$$ LANGUAGE plpgsql;

--\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

CREATE OR REPLACE FUNCTION check\_delVacReqVacEdVacDutyVacEmp()

RETURNS TRIGGER AS $$

BEGIN

DELETE from Vacancy\_requirement

WHERE id\_vac= old.id\_vac;

DELETE from Vacancy\_education

WHERE id\_vac=old.id\_vac;

DELETE from Vacancy\_duty

WHERE id\_vac=old.id\_vac;

DELETE from Vacancy\_employer

WHERE id\_vac=old.id\_vac;

RETURN NEW;

END;

$$ LANGUAGE plpgsql;

CREATE TRIGGER updForeignInsGLVac

AFTER UPDATE OF id\_vac ON GLVac

FOR EACH ROW

EXECUTE FUNCTION check\_VacReqVacEdVacDutyVacEmp();

CREATE TRIGGER delForeignInsGLVac

BEFORE DELETE ON GLVac

FOR EACH ROW

EXECUTE FUNCTION check\_delVacReqVacEdVacDutyVacEmp();

--\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

CREATE OR REPLACE FUNCTION check\_VacAndVacVol()

RETURNS TRIGGER AS $$

BEGIN

IF NOT EXISTS (SELECT 1 FROM GLVac WHERE id\_vac = NEW.id\_vac) THEN

RAISE EXCEPTION 'GLVac with id % does not exist', NEW.id\_vac;

END IF;

RETURN NEW;

END;

$$ LANGUAGE plpgsql;

CREATE TRIGGER checkGLVacInsVacEmp

BEFORE INSERT OR UPDATE ON Vacancy\_employer

FOR EACH ROW

EXECUTE FUNCTION check\_VacAndVacVol();

--\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

CREATE TRIGGER checkGLVacInsVacReq

BEFORE INSERT OR UPDATE ON Vacancy\_requirement

FOR EACH ROW

EXECUTE FUNCTION check\_VacAndVacVol();

--\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

CREATE TRIGGER checkGLVacInsVacDuty

BEFORE INSERT OR UPDATE ON Vacancy\_duty

FOR EACH ROW

EXECUTE FUNCTION check\_VacAndVacVol();

--\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

CREATE TRIGGER checkGLVacInsVacEd

BEFORE INSERT OR UPDATE ON Vacancy\_education

FOR EACH ROW

EXECUTE FUNCTION check\_VacAndVacVol();