**ОТЧЕТ**

**О ВЫПОЛНЕНИИ ПРАКТИЧЕСКОЙ РАБОТЫ ПО БАЗАМ ДАННЫХ**

студента 3 курса, группы 22205

Рыленко Валентина Григорьевича

Направление 09.03.01 – «Информатика и вычислительная техника»

Преподаватель:

Рылов Сергей Александрович

Новосибирск 2025

**СОДЕРЖАНИЕ**

[ЗАДАНИЕ 3](#_30j0zll)

[ОПИСАНИЕ РАБОТЫ](#_1fob9te) 5-7

[ЗАКЛЮЧЕНИЕ](#_3znysh7) 7

# **ЗАДАНИЕ**

Спортивная инфраструктура города представлена спортивными сооружениями различного типа: спортивные залы, манежи, стадионы, корты и т.д. Каждая из категорий спортивных сооружений обладает атрибутами, специфичными только для нее: стадион характеризуется вместимостью, корт - типом покрытия.

Спортсмены под руководством тренеров занимаются отдельными видами спорта, при этом один и тот же спортсмен может заниматься несколькими видами спорта, и в рамках одного и того же вида спорта может тренироваться у нескольких тренеров. Все спортсмены объединяются в спортивные клубы, при этом каждый из них может выступать только за один клуб.

Организаторы соревнований проводят состязания по отдельным видам спорта на спортивных сооружениях города. По результатам участия спортсменов в соревнованиях производится награждение.

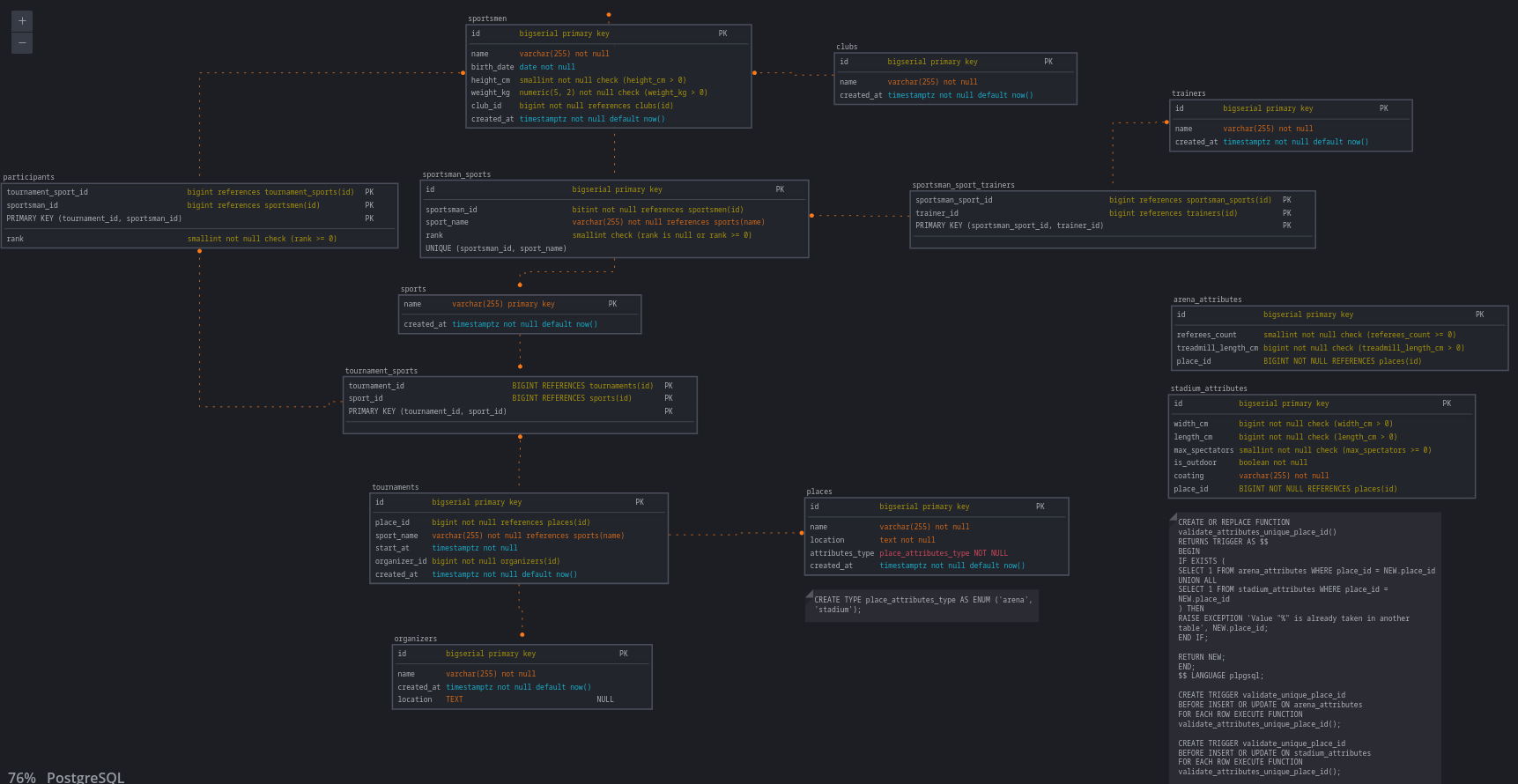
# **ОПИСАНИЕ РАБОТЫ**

Модель была реализована на сайте *sqldbm.com*: <https://app.sqldbm.com/PostgreSQL/DatabaseExplorer/p330003/> Запросы были написаны на SQL для базы данных PostgreSQL. Запросы создания таблиц находятся в приложении 1.

В таблице спортсменов задаётся идентификатор, имя, дата рождения, рост, вес, идентификатор клуба и время создания.

В таблице клубов задаётся идентификатор, имя и время создания.

В таблице организаторов задаётся идентификатор, расположение и время создания.

**

В таблице атрибутов арены задаётся идентификатор, идентификатор места, количество рефери и длина беговой дорожки.  
 В таблице атрибутов стадиона задаётся идентификатор, идентификатор места, ширина поля, длина поля, максимальное количество зрителей, открытый ли стадион и покрытие.

Для таблиц атрибутов создан триггер, который делает поле идентификатора места уникальным среди всех таблиц атрибутов.

В талице мест задаётся идентификатор, имя, расположение, типа атрибутов и время создания. Тип атрибутов является отдельным типом, определяющим перечисление.

В таблице видов спорта задаётся идентификатор, уникальное название и время создания.

В таблице тренеров задаётся идентификатор, имя и время создания.

Есть связь спорстмена и спорта, где задаётся идентификатор связи, идентификатор спортсмена, идентификатор спорта и разряд спортсмена в этом спорте.

Есть связь тренера и спортсмена определённого спорта, где задаётся идентификатор связи спортсмена и спорта и идентификатор тренера. Эта связь в отдельной таблице, чтобы не дублировать идентификаторы спортсмена и спорта для каждого тренера.

Для таблицы соревнований задаётся идентификатор, идентификатор места, время начала, идентификатор организатора и время создания.

Есть связь между соревнованиями и видами спорта: в соревновании может быть несколько видов спорта. В связи задаётся идентификатор соревнования и идентификтатор спорта.

В таблице участий спортсмена в соревновании задаётся идентификатор связи спорта в соревновании, идентификатор спортсмена, место в соревновании и необязательный текст результатов спортсмена.

# **ЗАКЛЮЧЕНИЕ**

# **Приложение 1.** Запросы создания таблиц

CREATE TABLE IF NOT EXISTS arena\_attributes (

id BIGSERIAL PRIMARY KEY,

place\_id BIGINT NOT NULL REFERENCES places(id),

referees\_count SMALLINT NOT NULL CHECK (referees\_count >= 0),

treadmill\_length\_cm BIGINT NOT NULL CHECK (treadmill\_length\_cm > 0)

);

CREATE TABLE IF NOT EXISTS stadium\_attributes (

id BIGSERIAL PRIMARY KEY,

place\_id BIGINT NOT NULL REFERENCES places(id),

width\_cm BIGINT NOT NULL CHECK (width\_cm > 0),

length\_cm BIGINT NOT NULL CHECK (length\_cm > 0),

max\_spectators SMALLINT NOT NULL CHECK (max\_spectators >= 0),

is\_outdoor BOOLEAN NOT NULL,

coating VARCHAR(255) NOT NULL

);

CREATE OR REPLACE FUNCTION validate\_attributes\_unique\_place\_id()

RETURNS TRIGGER AS $$

BEGIN

IF EXISTS (

SELECT 1 FROM arena\_attributes WHERE place\_id = NEW.place\_id

UNION ALL

SELECT 1 FROM stadium\_attributes WHERE place\_id = NEW.place\_id

) THEN

RAISE EXCEPTION 'Value "%" is already taken in another table', NEW.place\_id;

END IF;

RETURN NEW;

END;

$$ LANGUAGE plpgsql;

CREATE TRIGGER validate\_unique\_place\_id

BEFORE INSERT OR UPDATE ON arena\_attributes

FOR EACH ROW EXECUTE FUNCTION validate\_attributes\_unique\_place\_id();

CREATE TRIGGER validate\_unique\_place\_id

BEFORE INSERT OR UPDATE ON stadium\_attributes

FOR EACH ROW EXECUTE FUNCTION validate\_attributes\_unique\_place\_id();

CREATE TABLE IF NOT EXISTS clubs (

id BIGSERIAL PRIMARY KEY,

name VARCHAR(255) NOT NULL,

created\_at TIMESTAMPTZ NOT NULL DEFAULT NOW()

);

CREATE TABLE IF NOT EXISTS organizers (

id BIGSERIAL PRIMARY KEY,

name VARCHAR(255) NOT NULL,

location TEXT,

created\_at TIMESTAMPTZ NOT NULL DEFAULT NOW()

);

CREATE TYPE place\_attributes\_type AS ENUM ('arena', 'stadium');

CREATE TABLE IF NOT EXISTS places (

id BIGSERIAL PRIMARY KEY,

name VARCHAR(255) NOT NULL,

location TEXT NOT NULL,

attributes\_type place\_attributes\_type NOT NULL,

created\_at TIMESTAMPTZ NOT NULL DEFAULT NOW()

);

CREATE TABLE IF NOT EXISTS sports (

id BIGSERIAL PRIMARY KEY,

name VARCHAR(255) UNIQUE,

created\_at TIMESTAMPTZ NOT NULL DEFAULT NOW()

);

CREATE TABLE IF NOT EXISTS tournament\_sports (

tournament\_id BIGINT REFERENCES tournaments(id),

sport\_id BIGINT REFERENCES sports(id),

PRIMARY KEY (tournament\_id, sport\_id)

);

CREATE TABLE IF NOT EXISTS tournaments (

id BIGSERIAL PRIMARY KEY,

place\_id BIGINT NOT NULL REFERENCES places(id),

start\_at TIMESTAMPTZ NOT NULL,

organizer\_id BIGINT NOT NULL organizers(id),

created\_at TIMESTAMPTZ NOT NULL DEFAULT NOW()

);

CREATE TABLE IF NOT EXISTS sportsmen (

id BIGSERIAL PRIMARY KEY,

name VARCHAR(255) NOT NULL,

birth\_date DATE NOT NULL,

height\_cm SMALLINT NOT NULL CHECK (height\_cm > 0),

weight\_kg NUMERIC(5, 2) NOT NULL CHECK (weight\_kg > 0),

club\_id BIGINT NOT NULL REFERENCES clubs(id),

created\_at TIMESTAMPTZ NOT NULL DEFAULT NOW()

);

CREATE TABLE IF NOT EXISTS participations (

tournament\_sport\_id BIGINT REFERENCES tournament\_sports(id),

sportsman\_id BIGINT REFERENCES sportsmen(id),

rank SMALLINT NOT NULL CHECK (rank >= 0),

results TEXT,

PRIMARY KEY (tournament\_sport\_id, sportsman\_id)

);

CREATE TABLE IF NOT EXISTS sportsman\_sports (

id BIGSERIAL PRIMARY KEY,

sportsman\_id BIGINT NOT NULL REFERENCES sportsmen(id),

sport\_id BIGINT NOT NULL REFERENCES sports(id),

rank SMALLINT CHECK (rank IS NULL OR rank >= 0),

UNIQUE (sportsman\_id, sport\_id)

);

CREATE TABLE IF NOT EXISTS trainers (

id BIGSERIAL PRIMARY KEY,

name VARCHAR(255) NOT NULL,

created\_at TIMESTAMPTZ NOT NULL DEFAULT NOW()

);

CREATE TABLE IF NOT EXISTS sportsman\_sport\_trainers (

sportsman\_sport\_id BIGINT REFERENCES sportsman\_sports(id),

trainer\_id BIGINT REFERENCES trainers(id),

PRIMARY KEY (sportsman\_sport\_id, trainer\_id)

);