**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ  
 РОССИЙСКОЙ ФЕДЕРАЦИИ**

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**

**НОВОСИБИРСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ**

**Факультет информационных технологий**

**ОТЧЕТ**

**О ВЫПОЛНЕНИИ СЕМЕСТРОВОЙ РАБОТЫ**

«Разработка базы данных информационной системы спортивных организаций города»

студента 3 курса, 21210 группы

**Леонтьева Данила Вячеславовича**

Направление 09.03.01 – «Информатика и вычислительная техника»

Преподаватель:

Е.И. Вершинина

Новосибирск 2024

Оглавление

[Задание 3](#_Toc167638583)

[Общая информация 3](#_Toc167638584)

[Запросы 3](#_Toc167638585)

[Инфологическое проектирование 4](#_Toc167638586)

[Необходимый набор сущностей, отражающих предметную область и информационные потребности пользователей 4](#_Toc167638587)

[Классификация сущностей и необходимых наборов атрибутов 4](#_Toc167638588)

[Сущности вида подтип/супертип 7](#_Toc167638589)

[Связи между сущностями 7](#_Toc167638590)

[Формализация и описание связей между сущностями 8](#_Toc167638591)

[ER-диаграмма модели базы данных 10](#_Toc167638592)

[Модель базы данных на языке инфологического проектирования 10](#_Toc167638593)

[Логическое проектирование 12](#_Toc167638594)

[Нормализация таблиц 12](#_Toc167638595)

[Реляционная модель данных 12](#_Toc167638596)

[Вертикальная диаграмма 14](#_Toc167638597)

[Представления 15](#_Toc167638598)

[Триггеры 16](#_Toc167638599)

[Запросы 21](#_Toc167638600)

[Клиентское приложение 27](#_Toc167638601)

[Описание 27](#_Toc167638602)

[Запросы 27](#_Toc167638603)

# Задание

## Общая информация

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.

## Запросы

1. Получить перечень спортивных сооружений указанного типа в целом или удовлетворяющих заданным характеристикам (например, стадионы, вмещающие не менее указанного числа зрителей).
2. Получить список спортсменов, занимающихся указанным видом спорта в целом либо не ниже определенного разряда.
3. Получить список спортсменов, тренирующихся у некого тренера в целом либо не ниже определенного разряда.
4. Получить список спортсменов, занимающихся более чем одним видом спорта с указанием этих видов спорта.
5. Получить список тренеров указанного спортсмена.
6. Получить перечень соревнований, проведенных в течение заданного периода времени в целом либо указанным организатором.
7. Получить список призеров указанного соревнования.
8. Получить перечень соревнований, проведенных в указанном спортивном сооружении в целом либо по определенному виду спорта.
9. Получить перечень спортивных клубов и число спортсменов этих клубов, участвовавших в спортивных соревнованиях в течение заданного интервала времени.
10. Получить список тренеров по определенному виду спорта.
11. Получить список спортсменов, не участвовавших ни в каких соревнованиях в течение определенного периода времени.
12. Получить список организаторов соревнований и число проведенных ими соревнований в течение определенного периода времени.
13. Получить перечень спортивных сооружений и даты проведения на них соревнований в течение определенного периода времени.

# Инфологическое проектирование

## Необходимый набор сущностей, отражающих предметную область и информационные потребности пользователей

**Сущности:**

1. Sport\_place (общая информация о спортивных сооружениях проведения соревнований)
2. Sport\_place\_type (таблица, хранящая типы спортивных сооружений)
3. Coating\_type(таблица, хранящая названия покрытий для корта)
4. Stadium (информация, присущая стадионам)
5. Court (информация, присущая спортивным кортам)
6. Gym (информация, присущая тренажерным залам)
7. Arena (информация, присущая спортивным манежам)
8. Competition (информация о соревнованиях, проведенных в городе)
9. Organizer (информация об организаторах соревнований)
10. Organizer\_competition (связь соревнований с их организаторами)
11. Athlete (информация о спортсменах города)
12. Competition\_athlete (связывающая таблица с информацией об итогах выступления спортсменов на соревнованиях)
13. Sport\_club (информация о спортивных клубах)
14. Sport (информация о видах спорта, развивающихся в городе)
15. Athlete\_sport (связывающая таблица с информацией о достижениях спортсменов в спорте)
16. Sports\_rank (таблица, хранящая названия спортивных разрядов)
17. Trainer (информация о тренерах города)
18. Trainer\_athlete (связь спортсменов и их тренеров)
19. Trainer\_sport (связывающая таблица с информацией о компетентности тренера в спорте)
20. Trainer\_qualification (таблица, хранящая названия тренерских квалификаций)

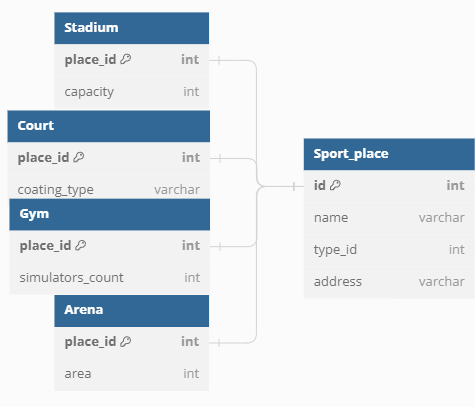
## Классификация сущностей и необходимых наборов атрибутов

**Все данные представлены в таблице:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **№** | **Сущность** | **Классификация сущности** | **Атрибут** | **Классификация атрибута** | **Primary key** | **Описание атрибута** |
| 1 | Sport\_place | Обозначающая | id | Указывающий | + | Id спортивного сооружения |
| name | Описательный | - | Название |
| type\_id | Вспомогательный | - | Id, указывающий на таблицу с типом данного сооружения |
| address | Описательный | - | адрес |
| 2 | Sport\_place\_type | Обозначающая | id | Указывающий | + | Id названия типа |
| name | Описательный | - | Название типа |
| 3 | Coating\_type | Обозначающая | id | Указывающий | + | Id покрытия |
| name | Описательный | - | Название покрытия |
| 4 | Stadium | Характеристическая | place\_id | Вспомогательный | + | Id, указывающий спортивное место |
| capacity | Описательный | - | Вместимость стадиона |
| 5 | Court | Характеристическая | place\_id | Вспомогательный | + | Id, указывающий спортивное место |
| coating\_type\_id | Вспомогательный | - | Id, указывающий на тип покрытия корта |
| 6 | Gym | Характеристическая | place\_id | Вспомогательный | + | Id, указывающий спортивное место |
| simulators\_count | Описательный | - | Количество тренажеров |
| 7 | Arena | Характеристическая | place\_id | Вспомогательный | + | Id, указывающий спортивное место |
| area | Описательный | - | Площадь манежа |
| 8 | Competition | Стержневая | id | Указывающий | + | Id соревнования |
| name | Описательный | - | Название соревнований |
| sport\_id | Вспомогательный | - | Id, указывающий на то, какие соревнования проводятся |
| place\_id | Вспомогательный | - | Id, указывающий на то, где проводятся соревнования |
| competition\_date | Описательный | - | Дата проведения |
| 9 | Organizer | Стержневая | id | Указывающий | + | Id организатора соревнований |
| first\_name | Описательный | - | Имя |
| last\_name | Описательный | - | Фамилия |
| father\_name | Описательный | - | Отчество |
| 10 | Organizer\_competition | Ассоциативная | Organizer\_id | Вспомогательный | + | Id, указывающий на организатора соревнований |
| Competition\_id | Вспомогательный | + | Id, указывающий на соревнование |
| 11 | Athlete | Стержневая | id | Указывающий | + | Id спортсмена |
| first\_name | Описательный | - | Имя |
| last\_name | Описательный | - | Фамилия |
| father\_name | Описательный | - | Отчество |
| club\_id | Вспомогательный | - | Id клуба, в котором состоит спортсмен |
| 12 | Competition\_athlete | Ассоциативная | competition\_id | Вспомогательный | + | Id, указывающий на соревнование |
| athlete\_id | Вспомогательный | + | Id, указывающий на спортсмена |
| place | Описательный | - | Место спортсмена в этих соревнованиях |
| 13 | Sport\_club | Обозначающая | id | Указывающий | + | Id клуба |
| name | Описательный | - | Название |
| creation\_date | Описательный | - | Дата создания |
| 14 | Sport | Обозначающая | id | Указывающий | + | Id спорта |
| name | Описательный | - | Название |
| description | Описательный | - | Описание |
| 15 | Athlete\_sport | Ассоциативная | athlete\_id | Вспомогательный | + | Id, указывающий на спортсмена |
| sport\_id | Вспомогательный | + | Id, указывающий на спорт |
| sports\_rank\_id | Вспомогательный | - | Спортивный разряд спортсмена в этом спорте |
| 16 | Sports\_rank | Обозначающая | id | Указывающий | + | Id спортивного разряда |
| name | Описательный | - | Его название |
| 17 | Trainer | Стержневая | id | Указывающий | + | Id тренера |
| first\_name | Описательный | - | Имя |
| last\_name | Описательный | - | Фамилия |
| father\_name | Описательный | - | Отчество |
| 18 | Trainer\_athlete | Ассоциативная | trainer\_id | Вспомогательный | + | Id, указывающий на тренера |
| athlete\_id | Вспомогательный | + | Id, указывающий на спортсмена |
| sport\_id | Вспомогательный | + | Id спорта, в рамках которого тренер занимается со спортсменом |
| 19 | Trainer\_sport | Ассоциативная | trainer\_id | Вспомогательный | + | Id, указывающий на тренера |
| sport\_id | Вспомогательный | + | Id, указывающий на спорт |
| qualification\_id | Вспомогательный | - | Id, указывающий на квалификация тренера в данном виде спорта |
| 20 | Trainer\_qualification | Обозначающая | id | Указывающий | + | Id квалификации |
| name | Описательный | - | Название квалификации |

## Сущности вида подтип/супертип

В данной модели есть супертип в виде сущности «спортивное сооружение», подтипом которого являются разновидности в виде стадионов, тренажерных залов, кортов и т.п. Супертип и подтип имеют связь один-к-одному через первичный ключ, а супертип имеет поле с типом сооружения.



## Связи между сущностями

СВЯЗЬ [сущность s1, сущность s2, …] (атрибут1, атрибут2, …)

Stadiums [Stadium 1, Sport\_place 1] (place\_id, id)

Courts [Court 1, Sport\_place 1] (place\_id, id)

Gyms [Gym 1, Sport\_place 1] (place\_id, id)

Arenas [Arena 1, Sport\_place 1] (place\_id, id)

Type of Coating[Coating\_type 1, Court M] (id, coating\_type\_id)

Type of place [Sport\_place\_type 1, Sport\_place M] (id, type\_id)

Place of competitions [Sport\_place 1, Competition M] (id, place\_id)

Competitions and organizers [Competition 1, Organizer\_competition M] (id, competition\_id)

Organizers and competitions [Organizer 1, Organizer\_competition M] (id, organizer\_id)

Sport on competition [Competition M, Sport 1] (sport\_id, id)

Sports and trainers [Sport 1, Trainer\_sport M] (id, sport\_id)

Trainers and sports [Trainer 1, Trainer\_sport M] (id, trainer\_id)

Qualification of trainer [Trainer\_sport M, Trainer\_qualification 1] (qualification\_id, id)

Trainers and athletes [Trainer 1, Trainer\_athlete M] (id, trainer\_id)

Athletes and trainers [Athlete 1, Trainer\_athlete M] (id, athlete\_id)

Club of athlete [Athlete M, Sport\_club 1] (club\_id, id)

Athletes and competitions [Athlete 1, Competition\_athlete M] (id, athlete\_id)

Competitions and athletes [Competition 1, Competition\_athlete M] (id, competition\_id)

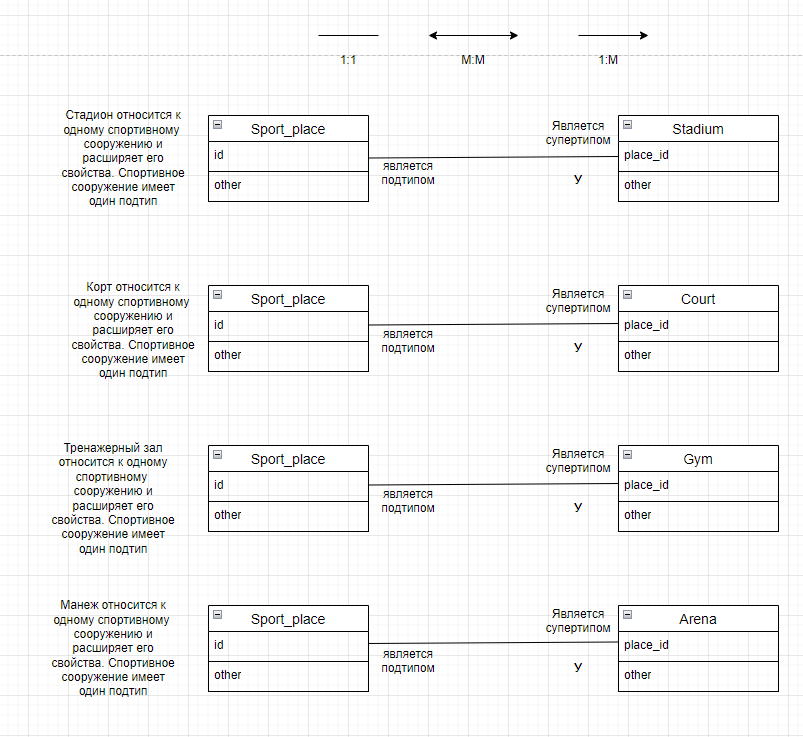
Athletes and sports [Athlete 1, Athlete\_sport M] (id, athlete\_id)

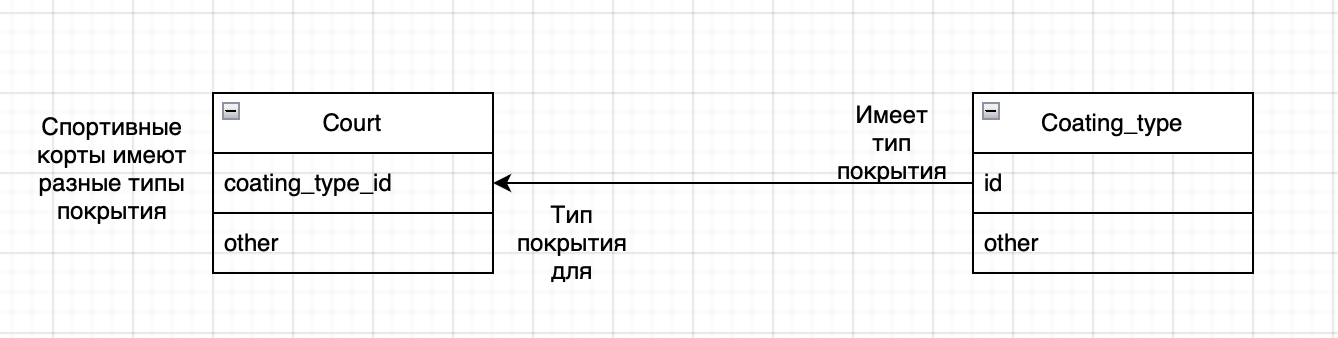
Rank of athlete [Athlete\_sport M, Sports\_rank 1] (sports\_rank\_id, id)

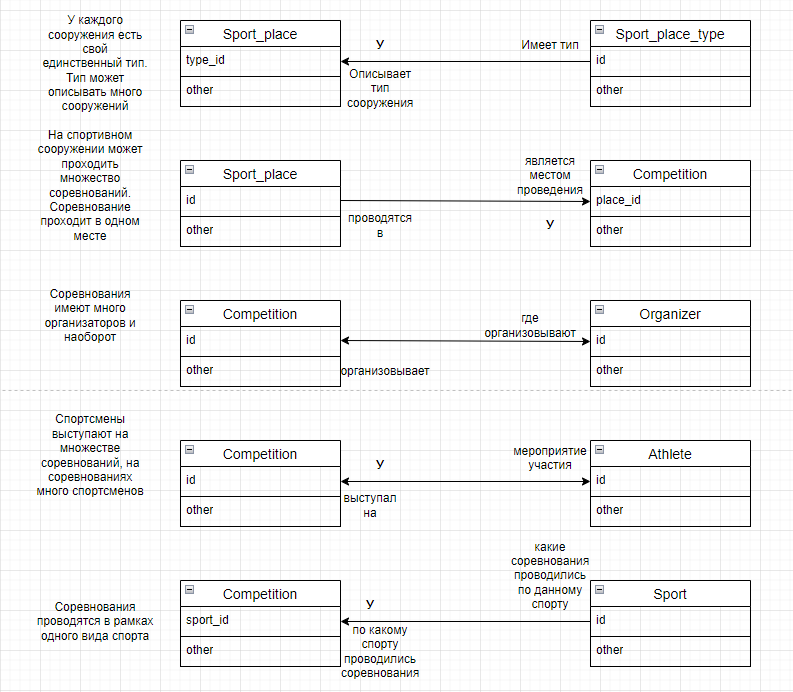
Sport of athlete [Sport 1, Trainer\_athlete M] (id, sport\_id)

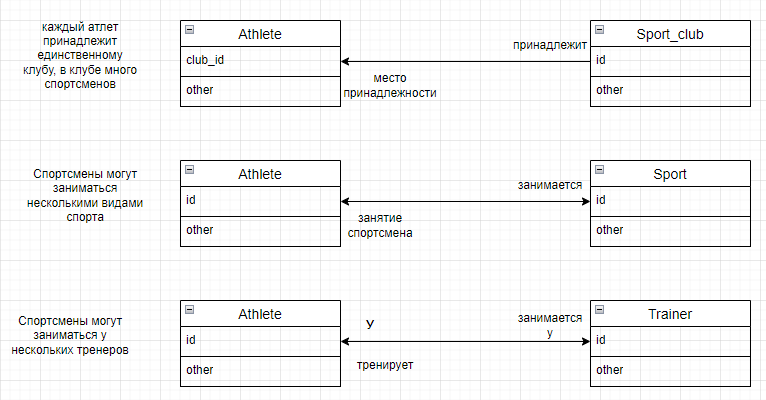
Sports and athletes [Athlete\_sport M, Sport 1] (sport\_id, id)

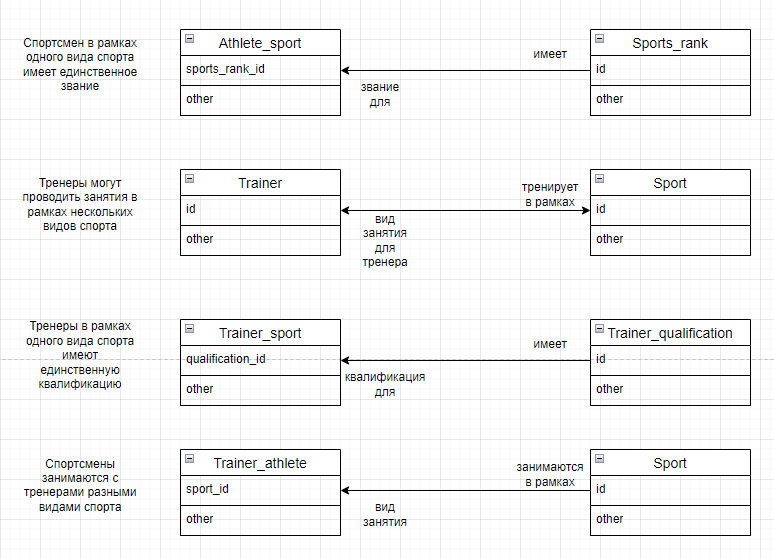
## Формализация и описание связей между сущностями



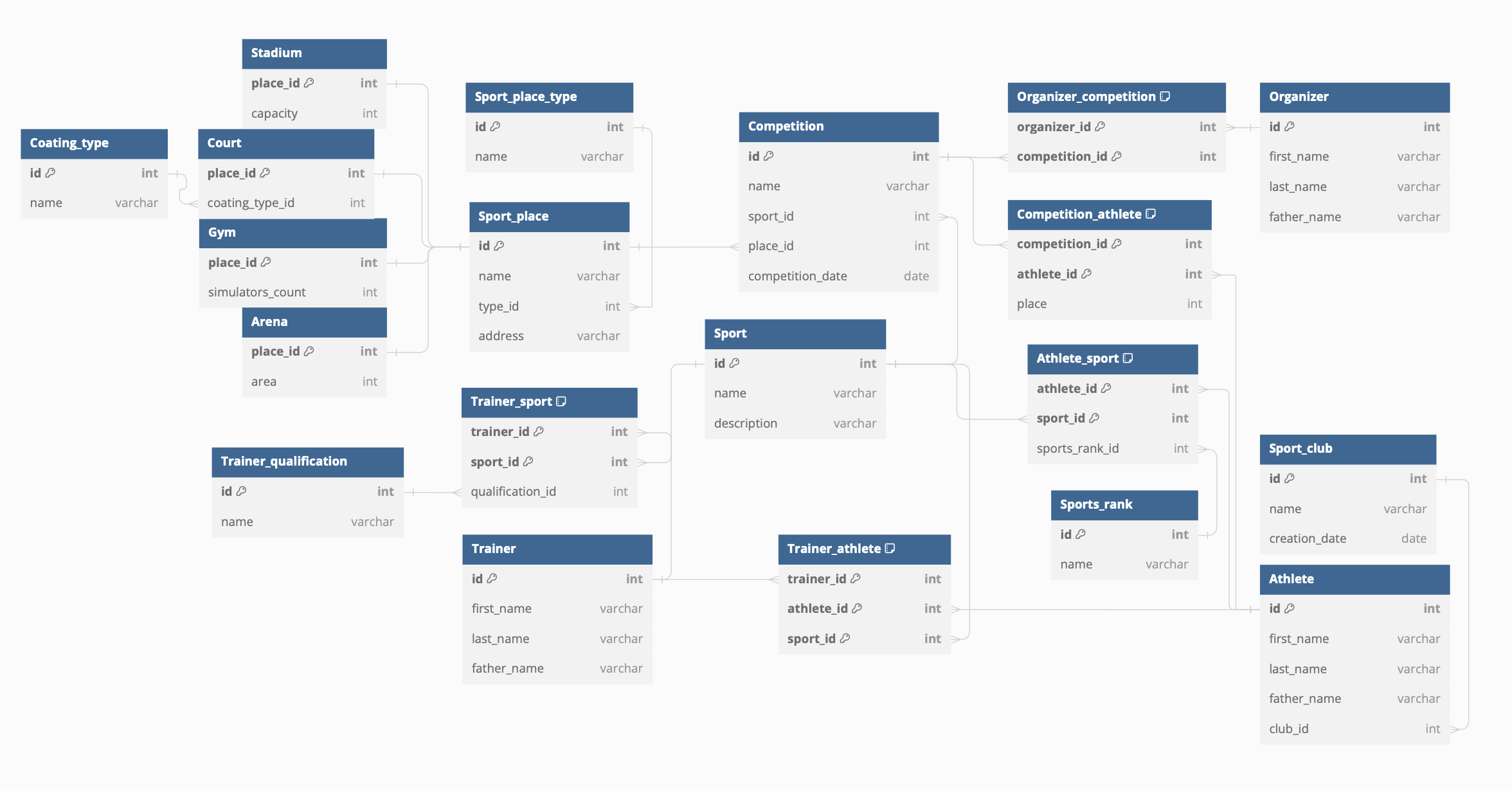








## ER-диаграмма модели базы данных



## Модель базы данных на языке инфологического проектирования

СТЕРЖЕНЬ (атрибут1, …)

АССОЦИАЦИЯ [сущность s1, сущность s2, …] (атрибут1, …)

ХАРАКТЕРИСТИКА (атрибут1, …) {список характеризуемых сущностей}

ОБОЗНАЧЕНИЕ (атрибут1, …) [список обозначаемых сущностей]

Sport\_place (id, name, type\_id, address) [Competition]

Sport\_place\_type (id, name) [Sport\_place]

Coating\_type (id, name) [Court]

Stadium (place\_id, capacity) {Sport\_place}

Court (place\_id, coating\_type\_id) {Sport\_place}

Gym (place\_id, simulators\_count) {Sport\_place}

Arena (place\_id, area) {Sport\_place}

Competition (id, name, sport\_id, place\_id, competition\_date)

Organizer (id, first\_name, last\_name, father\_name)

Organizer\_competition [Organizer M, Competition N] (organizer\_id, competition\_id)

Athlete (id, first\_name, last\_name, father\_name, club\_id)

Competition\_athlete [Competition M, Athlete N] (competition\_id, athlete\_id, place)

Sport\_club (id, name, creation\_date) [Athlete]

Sport (id, name, description) [Competition, Trainer\_athlete]

Athlete\_sport [Athlete M, Sport N] (athlete\_id, sport\_id, sports\_rank\_id)

Sports\_rank (id, name) [Athlete\_sport]

Trainer (id, first\_name, last\_name, father\_name)

Trainer\_athlete [Trainer M, Athlete N] (trainer\_id, athlete\_id, sport\_id)

Trainer\_sport [Trainer, Sport] (trainer\_id, sport\_id, qualification\_id)

Trainer\_qualification (id, name) [Trainer\_sport]

Связи на инфологическом языке представлены в главе «[Определение связи между сущностями](#связи)»

# Логическое проектирование

## Нормализация таблиц

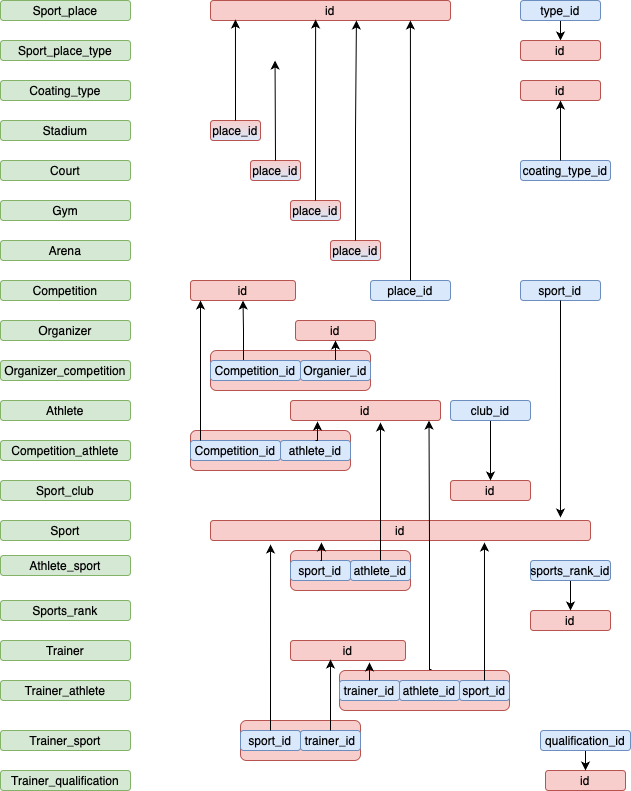
Все таблицы находятся в третьей нормальной форме, поскольку нетранзитивно зависят от первичного ключа.

## Реляционная модель данных

**Все данные представлены в таблице:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **№** | **Сущность** | **Атрибут** | **Тип** | **Ограничения типов** | **Pk, Fk** | **Ограничение на удаление** |
| 1 | Sport\_place | id | integer |  | Pk |  |
| name | Varchar(150) | Not null |  |  |
| type\_id | integer | Not null | Fk(Sport\_place\_type.id) | Restrict |
| address | Varchar(100) | Not null |  |  |
| Unique(name, address) | | | | |
| 2 | Sport\_place\_type | id | integer |  | Pk |  |
| name | Varchar(50) | Not null, Unique |  |  |
| 3 | Coating\_type | id | integer |  | Pk |  |
| name | Varchar(50) | Not null, Unique |  |  |
| 4 | Stadium | place\_id | integer |  | Pk, Fk(Sport\_place.id) | Cascade |
| capacity | integer | Not null, Check (capacity > 0) |  |  |
| 5 | Court | place\_id | integer |  | Pk, Fk(Sport\_place.id) | Cascade |
| coating\_type\_id | integer | Not null | Fk(Coating\_type.id) | Restrict |
| 6 | Gym | place\_id | integer |  | Pk, Fk(Sport\_place.id) | Cascade |
| simulators\_count | integer | Not null, Check (simulators\_count > 0) |  |  |
| 7 | Arena | place\_id | integer |  | Pk, Fk(Sport\_place.id) | Cascade |
| area | integer | Not null, Check(area > 0) |  |  |
| 8 | Competition | id | integer |  | Pk |  |
| name | Varchar(150) | Not null |  |  |
| sport\_id | integer | Not null | Fk(Sport.id) | Restrict |
| place\_id | integer | Not null | Fk(Sport\_place.id) | Restrict |
| competition\_date | date | Check(extract(year from competition\_date) > 1900), Not null |  |  |
| Unique(name, competition\_date, place\_id) | | | | |
| 9 | Organizer | id | integer |  | Pk |  |
| first\_name | Varchar(100) | Not null |  |  |
| last\_name | Varchar(100) | Not null |  |  |
| father\_name | Varchar(100) |  |  |  |
| 10 | Organizer\_competition | Organizer\_id | integer |  | Fk(Organizer.id) | Restrict |
| Competition\_id | integer |  | Fk(Competition.id) | Cascade |
|  |  | Pk(Organizer\_id, Competition\_id) | | | | |
| 11 | Athlete | id | integer |  | Pk |  |
| first\_name | Varchar(100) | Not null |  |  |
| last\_name | Varchar(100) | Not null |  |  |
| father\_name | Varchar(100) |  |  |  |
| club\_id | integer |  | Fk(Club.id) | Set null |
| 12 | Competition\_athlete | competition\_id | integer |  | Fk(Competition.id) | Restrict |
| athlete\_id | integer |  | Fk(Athlete.id) | Cascade |
| place | integer | Not null, Check(place > 0) |  |  |
|  |  | Pk(competition\_id, athlete\_id) | | | | |
| 13 | Sport\_club | id | integer |  | Pk |  |
| name | varchar(100) | Not null, Unique |  |  |
| creation\_date | date | Check(extract(year from creation\_date) > 1900) |  |  |
| 14 | Sport | id | integer |  | Pk |  |
| name | varchar(100) | Not null, Unique |  |  |
| description | varchar |  |  |  |
| 15 | Athlete\_sport | athlete\_id | integer |  | Fk(Athlete.id) | Cascade |
| sport\_id | integer |  | Fk(Sport.id) | Restrict |
| sports\_rank\_id | integer |  | Fk(Sports\_rank.id) | Restrict |
|  |  | Pk(athlete\_id, sport\_id) | | | | |
| 16 | Sports\_rank | id | integer |  | Pk |  |
| name | varchar(50) | Not null, Unique |  |  |
| 17 | Trainer | id | integer |  | Pk |  |
| first\_name | varchar(100) | Not null |  |  |
| last\_name | varchar(100) | Not null |  |  |
| father\_name | varchar(100) |  |  |  |
| 18 | Trainer\_athlete | trainer\_id | integer |  | Fk(Trainer.id) | Cascade |
| athlete\_id | integer |  | Fk(Athlete.id) | Cascade |
| sport\_id | integer |  | Fk(Sport.id) | Restrict |
|  |  | Pk(trainer\_id, athlete\_id, sport\_id) | | | | |
| 19 | Trainer\_sport | trainer\_id | integer |  | Fk(Trainer.id) | Cascade |
| sport\_id | integer |  | Fk(Sport.id) | Restrict |
| qualification\_id | integer |  | Fk(Trainer\_qualification) | Restrict |
|  |  | Pk(trainer\_id, sport\_id) | | | |  |
| 20 | Trainer\_qualification | id | integer |  | Pk |  |
| name | varchar(50) | Not null, Unique |  |  |

## Вертикальная диаграмма



## Представления

Для того, чтобы Обращаться к таблицам с подтипами как к одной полноценной сущности, были созданы представляения по каждому подтипу:

**Представление стадиона**

CREATE VIEW Stadium\_info AS

SELECT sp.id AS id,

sp.name AS name,

sp.type\_id AS type\_id,

sp.address AS address,

s.capacity AS capacity

FROM Sport\_place AS sp JOIN Stadium AS s

ON sp.id = s.place\_id

**Представление спортивного корта**

CREATE VIEW Court\_info AS

SELECT sp.id AS id,

sp.name AS name,

sp.type\_id AS type\_id,

sp.address AS address,

c.coating\_type\_id AS coating\_type\_id

FROM Sport\_place AS sp JOIN Court AS c

ON sp.id = c.place\_id

**Представление тренажерного зала**

CREATE VIEW Gym\_info AS

SELECT sp.id AS id,

sp.name AS name,

sp.type\_id AS type\_id,

sp.address AS address,

g.simulators\_count AS simulators\_count

FROM Sport\_place AS sp JOIN Gym AS g

ON sp.id = g.place\_id

**Представление спортивного манежа**

CREATE VIEW Arena\_info AS

SELECT sp.id AS id,

sp.name AS name,

sp.type\_id AS type\_id,

sp.address AS address,

a.area AS area

FROM Sport\_place AS sp JOIN Arena AS a

ON sp.id = a.place\_id

## Триггеры

Если спортсмен начал заниматься с тренером новым видом спорта, или этот спорт новый для тренера, то эти данные должны автоматически сохраниться в таблице с их квалификацией:

CREATE OR REPLACE FUNCTION trainer\_athlete\_insert\_trigger\_fnc()  
 RETURNS trigger AS  
$$  
BEGIN  
 IF NOT EXISTS  
 (SELECT 1 FROM Athlete\_sport AS a\_s  
 WHERE a\_s.athlete\_id = NEW.athlete\_id  
 AND a\_s.sport\_id = NEW.sport\_id)  
 THEN  
 INSERT INTO Athlete\_sport (athlete\_id, sport\_id)  
 VALUES(NEW.athlete\_id, NEW.sport\_id);  
 END IF;  
 IF NOT EXISTS  
 (SELECT 1 FROM Trainer\_sport AS ts  
 WHERE ts.trainer\_id = NEW.trainer\_id  
 AND ts.sport\_id = NEW.sport\_id)  
 THEN  
 INSERT INTO Trainer\_sport (trainer\_id, sport\_id)  
 VALUES(NEW.trainer\_id, NEW.sport\_id);  
 END IF;  
 RETURN NEW;  
END  
$$  
 LANGUAGE 'plpgsql';

CREATE OR REPLACE TRIGGER trainer\_athlete\_insert\_trigger  
 AFTER INSERT ON Trainer\_athlete  
 FOR EACH ROW  
EXECUTE PROCEDURE trainer\_athlete\_insert\_trigger\_fnc();

Для добавления информации в сущности с подтипами как в полноценные объекты были созданы следующие триггеры на представления:

**Триггер для добавления в тренажерный зал**

CREATE OR REPLACE FUNCTION gym\_insert\_trigger\_fnc()  
 RETURNS trigger AS  
$$  
DECLARE  
 sport\_place\_id integer;  
BEGIN  
 INSERT INTO Sport\_place(name, type\_id, address)  
 VALUES(NEW.name, 1, NEW.address);  
 SELECT sp.id INTO sport\_place\_id FROM Sport\_place AS sp  
 WHERE sp.name = NEW.name AND sp.address = NEW.address;  
 INSERT INTO gym(place\_id, simulators\_count)  
 VALUES(sport\_place\_id, NEW.simulators\_count);  
 RETURN NEW;  
END  
$$  
 LANGUAGE 'plpgsql';

CREATE OR REPLACE TRIGGER gym\_insert\_trigger  
 INSTEAD OF INSERT ON Gym\_info  
 FOR EACH ROW  
EXECUTE PROCEDURE gym\_insert\_trigger\_fnc();

**Триггер для добавления в спортивный манеж**

CREATE OR REPLACE FUNCTION arena\_insert\_trigger\_fnc()  
 RETURNS trigger AS  
$$  
DECLARE  
 sport\_place\_id integer;  
BEGIN  
 INSERT INTO Sport\_place(name, type\_id, address)  
 VALUES(NEW.name, 2, NEW.address);  
 SELECT sp.id INTO sport\_place\_id FROM Sport\_place AS sp  
 WHERE sp.name = NEW.name AND sp.address = NEW.address;  
 INSERT INTO arena(place\_id, area)  
 VALUES(sport\_place\_id, NEW.area);  
 RETURN NEW;  
END  
$$  
 LANGUAGE 'plpgsql';

CREATE OR REPLACE TRIGGER arena\_insert\_trigger  
 INSTEAD OF INSERT ON Arena\_info  
 FOR EACH ROW  
EXECUTE PROCEDURE arena\_insert\_trigger\_fnc();

**Триггер для добавления в стадион**

CREATE OR REPLACE FUNCTION stadium\_insert\_trigger\_fnc()  
 RETURNS trigger AS  
$$  
DECLARE  
 sport\_place\_id integer;  
BEGIN  
 INSERT INTO Sport\_place(name, type\_id, address)  
 VALUES(NEW.name, 3, NEW.address);  
 SELECT sp.id INTO sport\_place\_id FROM Sport\_place AS sp  
 WHERE sp.name = NEW.name AND sp.address = NEW.address;  
 INSERT INTO stadium(place\_id, capacity)  
 VALUES(sport\_place\_id, NEW.capacity);  
 RETURN NEW;  
END  
$$  
 LANGUAGE 'plpgsql';

CREATE OR REPLACE TRIGGER stadium\_insert\_trigger  
 INSTEAD OF INSERT ON Stadium\_info  
 FOR EACH ROW  
EXECUTE PROCEDURE stadium\_insert\_trigger\_fnc();

**Триггер для добавления в спортивный корт**

CREATE OR REPLACE FUNCTION court\_insert\_trigger\_fnc()  
 RETURNS trigger AS  
$$  
DECLARE  
 sport\_place\_id integer;  
BEGIN  
 INSERT INTO Sport\_place(name, type\_id, address)  
 VALUES(NEW.name, 4, NEW.address);  
 SELECT sp.id INTO sport\_place\_id FROM Sport\_place AS sp  
 WHERE sp.name = NEW.name AND sp.address = NEW.address;  
 INSERT INTO court(place\_id, coating\_type\_id)  
 VALUES(sport\_place\_id, NEW.coating\_type\_id);  
 RETURN NEW;  
END  
$$  
 LANGUAGE 'plpgsql';

CREATE OR REPLACE TRIGGER court\_insert\_trigger  
 INSTEAD OF INSERT ON Court\_info  
 FOR EACH ROW  
EXECUTE PROCEDURE court\_insert\_trigger\_fnc();

**Триггеры для удаления спортивных сооружений**

CREATE OR REPLACE FUNCTION sport\_place\_delete\_trigger\_fnc()  
 RETURNS trigger AS  
$$  
BEGIN  
 DELETE FROM Sport\_place WHERE id = OLD.id;  
 RETURN OLD;  
END  
$$  
 LANGUAGE 'plpgsql';  
  
CREATE OR REPLACE TRIGGER gym\_delete\_trigger  
 INSTEAD OF DELETE ON Gym\_info  
 FOR EACH ROW  
EXECUTE PROCEDURE sport\_place\_delete\_trigger\_fnc();  
  
CREATE OR REPLACE TRIGGER arena\_delete\_trigger  
 INSTEAD OF DELETE ON Arena\_info  
 FOR EACH ROW  
EXECUTE PROCEDURE sport\_place\_delete\_trigger\_fnc();  
  
CREATE OR REPLACE TRIGGER stadium\_delete\_trigger  
 INSTEAD OF DELETE ON Stadium\_info  
 FOR EACH ROW  
EXECUTE PROCEDURE sport\_place\_delete\_trigger\_fnc();  
  
CREATE OR REPLACE TRIGGER court\_delete\_trigger  
 INSTEAD OF DELETE ON Court\_info  
 FOR EACH ROW  
EXECUTE PROCEDURE sport\_place\_delete\_trigger\_fnc();

**Триггер для обновления тренажерного зала**

CREATE OR REPLACE FUNCTION gym\_update\_trigger\_fnc()  
 RETURNS trigger AS  
$$  
BEGIN  
 UPDATE Sport\_place SET name = NEW.name, address = NEW.address  
 WHERE id = OLD.id;  
 UPDATE Gym SET simulators\_count = NEW.simulators\_count  
 WHERE place\_id = OLD.id;  
 RETURN NEW;  
END  
$$  
 LANGUAGE 'plpgsql';  
CREATE OR REPLACE TRIGGER gym\_update\_trigger  
 INSTEAD OF UPDATE ON Gym\_info  
 FOR EACH ROW  
EXECUTE PROCEDURE gym\_update\_trigger\_fnc();

**Триггер для обновления спортивного манежа**

CREATE OR REPLACE FUNCTION arena\_update\_trigger\_fnc()  
 RETURNS trigger AS  
$$  
BEGIN  
 UPDATE Sport\_place SET name = NEW.name, address = NEW.address  
 WHERE id = OLD.id;  
 UPDATE Arena SET area = NEW.area  
 WHERE place\_id = OLD.id;  
 RETURN NEW;  
END  
$$  
 LANGUAGE 'plpgsql';  
CREATE OR REPLACE TRIGGER arena\_update\_trigger  
 INSTEAD OF UPDATE ON Arena\_info  
 FOR EACH ROW  
EXECUTE PROCEDURE arena\_update\_trigger\_fnc();

**Триггер для обновления стадиона**

CREATE OR REPLACE FUNCTION stadium\_update\_trigger\_fnc()  
 RETURNS trigger AS  
$$  
BEGIN  
 UPDATE Sport\_place SET name = NEW.name, address = NEW.address  
 WHERE id = OLD.id;  
 UPDATE Stadium SET capacity = NEW.capacity  
 WHERE place\_id = OLD.id;  
 RETURN NEW;  
END  
$$  
 LANGUAGE 'plpgsql';  
CREATE OR REPLACE TRIGGER stadium\_update\_trigger  
 INSTEAD OF UPDATE ON Stadium\_info  
 FOR EACH ROW  
EXECUTE PROCEDURE stadium\_update\_trigger\_fnc();

**Триггер для обновления спортивного корта**

CREATE OR REPLACE FUNCTION court\_update\_trigger\_fnc()  
 RETURNS trigger AS  
$$  
BEGIN  
 UPDATE Sport\_place AS sp SET name = NEW.name, address = NEW.address  
 WHERE sp.id = OLD.id;  
 UPDATE Court SET coating\_type\_id = NEW.coating\_type\_id  
 WHERE place\_id = OLD.id;  
 RETURN NEW;  
END  
$$  
 LANGUAGE 'plpgsql';  
CREATE OR REPLACE TRIGGER court\_update\_trigger  
 INSTEAD OF UPDATE ON Court\_info  
 FOR EACH ROW  
EXECUTE PROCEDURE court\_update\_trigger\_fnc();

## Запросы

**1.Получить перечень спортивных сооружений указанного типа в целом или удовлетворяющих заданным характеристикам (например, стадионы, вмещающие не менее указанного числа зрителей)**

SELECT si.id AS id,

si.name AS name,

spt.name AS place\_type,

si.address AS address,

si.capacity AS capacity

FROM Stadium\_info AS si JOIN Sport\_place\_type AS spt

ON si.type\_id = spt.id;

SELECT si.id AS id,

si.name AS name,

spt.name AS place\_type,

si.address AS address,

si.capacity AS capacity

FROM Stadium\_info AS si JOIN Sport\_place\_type AS spt

ON si.type\_id = spt.id

WHERE si.capacity > ?;

SELECT ci.id AS id,

ci.name AS name,

spt.name AS place\_type,

ci.address AS address,

ci.coating\_type\_id AS coating\_type

FROM Court\_info AS ci JOIN Sport\_place\_type AS spt

ON ci.type\_id = spt.id;

SELECT ci.id AS id,

ci.name AS name,

spt.name AS place\_type,

ci.address AS address,

ci.coating\_type\_id AS coating\_type

FROM Court\_info AS ci JOIN Sport\_place\_type AS spt

ON ci.type\_id = spt.id

WHERE ci.coating\_type = ?;

SELECT gi.id AS id,

gi.name AS name,

spt.name AS place\_type,

gi.address AS address,

gi. simulators\_count AS simulators\_count

FROM Gym\_info AS gi JOIN Sport\_place\_type AS spt

ON gi.type\_id = spt.id;

SELECT gi.id AS id,

gi.name AS name,

spt.name AS place\_type,

gi.address AS address,

gi. simulators\_count AS simulators\_count

FROM Gym\_info AS gi JOIN Sport\_place\_type AS spt

ON gi.type\_id = spt.id

WHERE gi.simulators\_count > ?;

SELECT ai.id AS id,

ai.name AS name,

spt.name AS place\_type,

ai.address AS address,

ai.area AS area

FROM Arena\_info AS ai JOIN Sport\_place\_type AS spt

ON ai.type\_id = spt.id;

SELECT ai.id AS id,

ai.name AS name,

spt.name AS place\_type,

ai.address AS address,

ai. area AS area

FROM Arena\_info AS ai JOIN Sport\_place\_type AS spt

ON ai.type\_id = spt.id

WHERE ai.area > ?;

**2.Получить список спортсменов, занимающихся указанным видом спорта в целом либо не ниже определенного разряда**

SELECT a.id AS id,

a.first\_name AS first\_name,

a.last\_name AS last\_name,

a.father\_name AS father\_name,

s.name AS sport,

sr.name AS sports\_rank

FROM Athlete AS a JOIN Athlete\_sport AS a\_s

ON a.id = a\_s.athlete\_id LEFT JOIN Sports\_rank AS sr

ON a\_s.sports\_rank\_id = sr.id

WHERE a\_s.sport\_id = ?

SELECT a.id AS id,

a.first\_name AS first\_name,

a.last\_name AS last\_name,

a.father\_name AS father\_name,

s.name AS sport,

sr.name AS sports\_rank

FROM Athlete AS a JOIN Athlete\_sport AS a\_s

ON a.id = a\_s.athlete\_id JOIN Sports\_rank AS sr

ON a\_s.sports\_rank\_id = sr.id

WHERE a\_s.sport\_id = ? AND sr.id > ?

**3.Получить список спортсменов, тренирующихся у некого тренера в целом либо не ниже определенного разряда**

SELECT a.id AS id,

a.first\_name AS first\_name,

a.last\_name AS last\_name,

a.father\_name AS father\_name,

s.name AS sport,

sr.name AS sports\_rank,

t.first\_name AS first\_name,

t.last\_name AS last\_name,

t.father\_name AS father\_name

FROM Athlete AS a JOIN Trainer\_athlete AS ta

ON a.id = ta.athlete\_id JOIN Trainer AS t

ON ta.trainer\_id = t.id JOIN Athlete\_sport AS a\_s

ON ta.athlete\_id = a\_s.athlete\_id AND ta.sport\_id = a\_s.sport\_id

LEFT JOIN Sports\_rank AS sr ON a\_s.sports\_rank\_id = sr.id

WHERE t.id = ?

SELECT a.id AS id,

a.first\_name AS first\_name,

a.last\_name AS last\_name,

a.father\_name AS father\_name,

s.name AS sport,

sr.name AS sports\_rank,

t.first\_name AS first\_name,

t.last\_name AS last\_name,

t.father\_name AS father\_name

FROM Athlete AS a JOIN Trainer\_athlete AS ta

ON a.id = ta.athlete\_id JOIN Trainer AS t

ON ta.trainer\_id = t.id JOIN Athlete\_sport AS a\_s

ON ta.athlete\_id = a\_s.athlete\_id AND ta.sport\_id = a\_s.sport\_id

JOIN Sports\_rank AS sr ON a\_s.sports\_rank\_id = sr.id

WHERE t.id = ? AND sr.id > ?

**4.Получить список спортсменов, занимающихся более чем одним видом спорта с указанием этих видов спорта**

SELECT a.id AS id,

a.first\_name AS first\_name,

a.last\_name AS last\_name,

a.father\_name AS father\_name,

s.name AS sport,

FROM Athlete AS a JOIN Athlete\_sport AS a\_s

ON a.id = a\_s.athlete\_id JOIN Sport AS s

ON a\_s.sport\_id = s.id JOIN

(SELECT ats.athlete\_id AS ats\_id, COUNT(\*) AS cnt

FROM Athlete\_sport AS ats

GROUP BY ats.athlete\_id) AS counts

ON a\_s.athlete\_id = counts.ats\_id

WHERE counts.cnt > ?

**5.Получить список тренеров указанного спортсмена**

SELECT t.id AS id,

t.first\_name AS first\_name,

t.last\_name AS last\_name,

t.father\_name AS father\_name

s.name AS sport\_name

FROM Trainer AS t JOIN Trainer\_athlete AS ta

ON t.id = ta.trainer\_id JOIN Athlete AS a

ON a.id = ta.atlete\_id JOIN Sport AS s

ON s.id = ta.sport\_id

WHERE a.id = ?

**6.Получить перечень соревнований, проведенных в течение заданного периода времени в целом либо указанным организатором**

SELECT c.id AS id,

c.name AS name,

c.sport\_id AS sport\_id,

c.place\_id AS place\_id,

c.competition\_date AS competition\_date

FROM Competition AS c

WHERE c.competition\_date BETWEEN ? AND ?

SELECT c.id AS id,

c.name AS name,

c.sport\_id AS sport\_id,

c.place\_id AS place\_id,

c.competition\_date AS competition\_date

FROM Competition AS c JOIN Organizer\_competition AS oc

ON c.id = oc.competition\_id

WHERE c.competition\_date BETWEEN ? AND ?

AND oc.organizer\_id = ?

**7.Получить список призеров указанного соревнования**

SELECT a.id AS id,

a.first\_name AS first\_name,

a.last\_name AS last\_name,

a.father\_name AS father\_name,

ca.place AS place

FROM Athlete AS a JOIN Competition\_athlete AS ca

ON a.id = ca.athlete\_id JOIN Competition AS c

ON c.id = ca.competition\_id

WHERE c.id = ? AND ca.place >=1 AND ca.place <= 3

ORDER BY ca.place

**8.Получить перечень соревнований, проведенных в указанном спортивном сооружении в целом либо по определенному виду спорта**

SELECT c.id AS id,

c.name AS name,

s.name AS sport,

c.place\_id AS place\_id,

c.competition\_date AS competition\_date

FROM Competition AS c JOIN Sport AS s

ON c.sport\_id = s.id

WHERE c.place\_id = ?

SELECT c.id AS id,

c.name AS name,

s.name AS sport,

c.place\_id AS place\_id,

c.competition\_date AS competition\_date

FROM Competition AS c JOIN Sport AS s

ON c.sport\_id = s.id

WHERE c.place\_id = ? AND s.id = ?

**9.Получить перечень спортивных клубов и число спортсменов этих клубов, участвовавших в спортивных соревнованиях в течение заданного интервала времени**

SELECT o.first\_name AS first\_name,  
 o.last\_name AS last\_name,  
 o.father\_name AS father\_name,  
 organizations.organization\_count AS organization\_count  
FROM Organizer AS o LEFT JOIN  
(SELECT io.id AS iid,  
count(\*) AS organization\_count  
FROM Organizer AS io JOIN Organizer\_competition AS oc  
ON io.id = oc.organizer\_id JOIN Competition AS c  
ON c.id = oc.competition\_id  
WHERE c.competition\_date BETWEEN ? AND ?  
GROUP BY io.id) AS organizations  
ON o.id = organizations.iid

**10.Получить список тренеров по определенному виду спорта**

SELECT a.id AS id,

t.first\_name AS first\_name,

t.last\_name AS last\_name,

t.father\_name AS father\_name,

s.name AS sport,

tq.name AS qualification

FROM Trainer AS t JOIN Trainer\_sport AS ts

ON t.id = ts.trainer\_id LEFT JOIN Trainer\_qualification AS tq

ON ts.qualification\_id = tq.id

WHERE ts.sport\_id = ?

**11.Получить список спортсменов, не участвовавших ни в каких соревнованиях в течение определенного периода времени**

SELECT a.id AS id,

a.first\_name AS first\_name,

a.last\_name AS last\_name,

a.father\_name AS father\_name

FROM Athlete AS a

WHERE a.id NOT IN (SELECT \* FROM

Competition\_athlete AS ca JOIN Competition AS c

ON c.id = ca.competition\_id

WHERE c.competition\_date BETWEEN ? AND ?) AS check

**12.Получить список организаторов соревнований и число проведенных ими соревнований в течение определенного периода времени**

SELECT o.id AS id,

o.first\_name AS first\_name,

o.last\_name AS last\_name,

o.father\_name AS father\_name

organizations.organization\_count AS organization\_count

FROM Organizer AS o LEFT JOIN

(SELECT io.id AS iid,

count(\*) AS organization\_count

FROM Organizer AS io JOIN Organizer\_competition AS oc

ON o.id = oc.organizer\_id JOIN Competition AS c

ON c.id = c.competition\_id

WHERE c.competition\_date BETWEEN ? AND ?) AS organizations

ON o.id = organizations.iid

**13.Получить перечень спортивных сооружений и даты проведения на них соревнований в течение определенного периода времени**

SELECT sp.id AS id,

sp.name AS name,

spt.name AS type,

sp.address AS address

c.name AS competition\_name

c.competition\_date AS competition\_date

FROM Sport\_place AS sp JOIN Sport\_place\_type AS spt

ON sp.type\_id = spt.id JOIN Competition AS c

ON sp.id = c.place\_id

WHERE c.competition\_date BETWEEN ? AND ?

# Клиентское приложение

## Описание

Клиентская часть представляет собой веб-приложение, состоящее из монолитной системы Java Spring и Thymeleaf. Для взаимодействия с базой данных использовался Spring Data JDBC. Коннект с базой происходит на старте приложения.

Скрипты миграци находятся в папке **db/migration**

Скрипты заполнения находятся в папке **db/testData**

Запуск описан в **README.md**

## Запросы

Страницы с выполненными запросами находятся на вкладках Объектов, которые мы хотим найти (например, есть запрос «найти атлетов по тренеру» - для этого мы переходим на вкладку «athletes», а на этой странице будет поле «find athletes by trainer». Вводим нужного тренера и получаем ответ)

Аналитические запросы были модифицированы под условия синтаксиса фреймворка и имеют следующий вид:

**1.Получить перечень спортивных сооружений указанного типа в целом или удовлетворяющих заданным характеристикам (например, стадионы, вмещающие не менее указанного числа зрителей)**

SELECT si.id AS id,  
 si.name AS name,  
 si.type\_id AS type\_id,  
 si.address AS address,  
 si.capacity AS capacity  
FROM Stadium\_info AS si

SELECT si.id AS id,  
 si.name AS name,  
 si.address AS address,  
 si.capacity AS capacity  
FROM Stadium\_info AS si

WHERE Stadium\_info.capacity >= :capacity

SELECT ci.id AS id,  
 ci.name AS name,  
 ci.address AS address,  
 ct.name AS coating\_type  
FROM Court\_info AS ci JOIN Coating\_type AS ct  
ON ci.coating\_type\_id = ct.id

SELECT ci.id AS id,  
 ci.name AS name,  
 ci.address AS address,  
 ct.name AS coating\_type  
FROM Court\_info AS ci JOIN Coating\_type AS ct  
ON ci.coating\_type\_id = ct.id  
WHERE ci.coating\_type\_id = :coatingTypeId

SELECT gi.id AS id,  
 gi.name AS name,  
 gi.type\_id AS type\_id,  
 gi.address AS address,  
 gi. simulators\_count AS simulators\_count  
FROM Gym\_info AS gi

SELECT gi.id AS id,  
 gi.name AS name,  
 gi.type\_id AS type\_id,  
 gi.address AS address,  
 gi.simulators\_count AS simulators\_count  
FROM Gym\_info AS gi  
WHERE gi.simulators\_count >= :simulatorsCount;

SELECT ai.id AS id,  
 ai.name AS name,  
 ai.type\_id AS type\_id,  
 ai.address AS address,  
 ai.area AS area  
FROM Arena\_info AS ai

SELECT ai.id AS id,  
 ai.name AS name,  
 ai.type\_id AS type\_id,  
 ai.address AS address,  
 ai.area AS area  
FROM Arena\_info AS ai  
WHERE ai.area >= :area

**2.Получить список спортсменов, занимающихся указанным видом спорта в целом либо не ниже определенного разряда**

SELECT a.first\_name AS first\_name,  
 a.last\_name AS last\_name,  
 sr.name AS sports\_rank  
FROM Athlete AS a JOIN Athlete\_sport AS a\_s  
ON a.id = a\_s.athlete\_id LEFT JOIN Sports\_rank AS sr  
ON a\_s.sports\_rank\_id = sr.id  
WHERE a\_s.sport\_id = :sportId

SELECT a.first\_name AS first\_name,  
 a.last\_name AS last\_name,  
 sr.name AS sports\_rank  
FROM Athlete AS a JOIN Athlete\_sport AS a\_s  
ON a.id = a\_s.athlete\_id JOIN Sports\_rank AS sr  
ON a\_s.sports\_rank\_id = sr.id  
WHERE a\_s.sport\_id = :sportId AND sr.id >= :rankId

**3.Получить список спортсменов, тренирующихся у некого тренера в целом либо не ниже определенного разряда**

SELECT a.id AS id,  
 a.first\_name AS first\_name,  
 a.last\_name AS last\_name,  
 a.father\_name AS father\_name  
FROM Athlete AS a JOIN Trainer\_athlete AS ta  
ON a.id = ta.athlete\_id  
WHERE ta.trainer\_id = :trainerId  
GROUP BY a.id

SELECT s.name AS sport\_name,  
 sr.name AS rank\_name  
FROM Athlete AS a JOIN Trainer\_athlete AS ta  
ON a.id = ta.athlete\_id JOIN Trainer AS t  
ON ta.trainer\_id = t.id JOIN Athlete\_sport AS a\_s  
ON ta.athlete\_id = a\_s.athlete\_id AND ta.sport\_id = a\_s.sport\_id  
JOIN Sport AS s ON a\_s.sport\_id = s.id  
LEFT JOIN Sports\_rank AS sr ON a\_s.sports\_rank\_id = sr.id  
WHERE t.id = :trainerId AND a.id=:athleteId

SELECT a.id AS id,  
 a.first\_name AS first\_name,  
 a.last\_name AS last\_name,  
 a.father\_name AS father\_name  
FROM Athlete AS a JOIN Trainer\_athlete AS ta  
ON a.id = ta.athlete\_id JOIN Athlete\_sport AS a\_s  
ON ta.athlete\_id = a\_s.athlete\_id AND ta.sport\_id = a\_s.sport\_id  
JOIN Sports\_rank AS sr ON a\_s.sports\_rank\_id = sr.id  
WHERE ta.trainer\_id = :trainerId AND sr.id >= :rankId  
GROUP BY a.id

SELECT s.name AS sport\_name,  
 sr.name AS rank\_name  
FROM Athlete AS a JOIN Trainer\_athlete AS ta  
ON a.id = ta.athlete\_id JOIN Trainer AS t  
ON ta.trainer\_id = t.id JOIN Athlete\_sport AS a\_s  
ON ta.athlete\_id = a\_s.athlete\_id AND ta.sport\_id = a\_s.sport\_id  
JOIN Sport AS s ON a\_s.sport\_id = s.id  
JOIN Sports\_rank AS sr ON a\_s.sports\_rank\_id = sr.id  
WHERE t.id = :trainerId AND a.id=:athleteId AND sr.id >= :rankId

**4.Получить список спортсменов, занимающихся более чем одним видом спорта с указанием этих видов спорта**

SELECT a.first\_name AS first\_name,  
 a.last\_name AS last\_name,  
 sr.name AS sports\_rank  
FROM Athlete AS a JOIN Athlete\_sport AS a\_s  
ON a.id = a\_s.athlete\_id JOIN Sports\_rank AS sr  
ON a\_s.sports\_rank\_id = sr.id  
WHERE a\_s.sport\_id = :sportId AND sr.id >= :rankId

SELECT s.id AS id,  
 s.name AS name,  
 s.description AS description  
FROM Sport AS s JOIN Athlete\_sport AS a\_s  
ON s.id = a\_s.sport\_id  
WHERE a\_s.athlete\_id = :athleteId

**5.Получить список тренеров указанного спортсмена**

SELECT t.id AS id,  
 t.first\_name AS first\_name,  
 t.last\_name AS last\_name,  
 t.father\_name AS father\_name  
FROM Trainer AS t JOIN Trainer\_athlete AS ta  
ON t.id = ta.trainer\_id  
WHERE ta.athlete\_id = :athleteId  
GROUP BY t.id

SELECT s.id AS id,  
 s.description AS description,  
 s.name AS name  
FROM Trainer AS t JOIN Trainer\_athlete AS ta  
ON t.id = ta.trainer\_id JOIN Athlete AS a  
ON a.id = ta.athlete\_id JOIN Sport AS s  
ON s.id = ta.sport\_id  
WHERE a.id = :athleteId AND t.id = :trainerId

**6.Получить перечень соревнований, проведенных в течение заданного периода времени в целом либо указанным организатором**

SELECT c.id AS id,  
 c.name AS name,  
 c.sport\_id AS sport\_id,  
 c.place\_id AS place\_id,  
 c.competition\_date AS competition\_date  
FROM Competition AS c  
WHERE c.competition\_date BETWEEN :dateFrom AND :dateTo

SELECT c.id AS id,  
 c.name AS name,  
 c.sport\_id AS sport\_id,  
 c.place\_id AS place\_id,  
 c.competition\_date AS competition\_date  
FROM Competition AS c JOIN Organizer\_competition AS oc  
ON c.id = oc.competition\_id  
WHERE c.competition\_date BETWEEN :dateFrom AND :dateTo AND oc.organizer\_id = :OrganizerId

**7.Получить список призеров указанного соревнования**

SELECT a.first\_name AS first\_name,  
 a.last\_name AS last\_name,  
 a.father\_name AS father\_name,  
 ca.place AS place  
FROM Athlete AS a JOIN Competition\_athlete AS ca  
ON a.id = ca.athlete\_id JOIN Competition AS c  
ON c.id = ca.competition\_id  
WHERE c.id = :competitionId AND ca.place >=1 AND ca.place <= 3  
ORDER BY ca.place

**8.Получить перечень соревнований, проведенных в указанном спортивном сооружении в целом либо по определенному виду спорта**

SELECT c.id AS id,  
 c.name AS name,  
 c.place\_id AS place\_id,  
 c.competition\_date AS competition\_date  
FROM Competition AS c  
WHERE c.place\_id = :placeId

SELECT c.id AS id,  
 c.name AS name,  
 c.place\_id AS place\_id,  
 c.competition\_date AS competition\_date  
FROM Competition AS c JOIN Sport AS s  
ON c.sport\_id = s.id  
WHERE c.place\_id = :placeId AND s.id = :sportId

**9.Получить перечень спортивных клубов и число спортсменов этих клубов, участвовавших в спортивных соревнованиях в течение заданного интервала времени**

SELECT sc.name AS name,  
 members.member\_count AS member\_count  
FROM Sport\_club AS sc LEFT JOIN  
(SELECT isc.id AS iid,  
 count(\*) AS member\_count  
FROM Sport\_club AS isc JOIN Athlete AS a  
ON isc.id = a.club\_id JOIN Competition\_athlete AS ca  
ON a.id = ca.athlete\_id JOIN Competition c  
ON c.id = ca.competition\_id  
WHERE c.competition\_date BETWEEN :from AND :to  
GROUP BY isc.id) AS members  
ON sc.id = members.iid

**10.Получить список тренеров по определенному виду спорта**

SELECT t.first\_name AS first\_name,  
 t.last\_name AS last\_name,  
 t.father\_name AS father\_name,  
 tq.name AS qualification  
FROM Trainer AS t JOIN Trainer\_sport AS ts  
ON t.id = ts.trainer\_id LEFT JOIN Trainer\_qualification AS tq  
ON ts.qualification\_id = tq.id  
WHERE ts.sport\_id = :sportId

**11.Получить список спортсменов, не участвовавших ни в каких соревнованиях в течение определенного периода времени**

SELECT a.id AS id,  
 a.first\_name AS first\_name,  
 a.last\_name AS last\_name,  
 a.father\_name AS father\_name  
FROM Athlete AS a  
WHERE a.id NOT IN (SELECT ca.athlete\_id FROM  
Competition\_athlete AS ca JOIN Competition AS c  
ON c.id = ca.competition\_id  
WHERE c.competition\_date BETWEEN :from AND :to)

**12.Получить список организаторов соревнований и число проведенных ими соревнований в течение определенного периода времени**

SELECT o.first\_name AS first\_name,  
 o.last\_name AS last\_name,  
 o.father\_name AS father\_name,  
 organizations.organization\_count AS organization\_count  
FROM Organizer AS o LEFT JOIN  
(SELECT io.id AS iid,  
 count(\*) AS organization\_count  
FROM Organizer AS io JOIN Organizer\_competition AS oc  
ON io.id = oc.organizer\_id JOIN Competition AS c  
ON c.id = oc.competition\_id  
WHERE c.competition\_date BETWEEN :dateFrom AND :dateTo  
GROUP BY io.id) AS organizations  
ON o.id = organizations.iid

**13.Получить перечень спортивных сооружений и даты проведения на них соревнований в течение определенного периода времени**

SELECT sp.id AS id,  
 sp.name AS name,  
 sp.type\_id AS type\_id,  
 sp.address AS address  
FROM Sport\_place AS sp JOIN Competition AS c  
ON sp.id = c.place\_id  
WHERE c.competition\_date BETWEEN :from AND :to

SELECT c.id AS id,  
 c.name AS name,  
 c.place\_id AS place\_id,  
 c.competition\_date AS competition\_date  
FROM Competition AS c  
WHERE c.place\_id = :placeId

Также на сайте мы можем увидеть списки всех объектов, на которые заходим, и есть полный редактор для спортсменов.