Минобрнауки России

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Волгоградский государственный технический университет»

Факультет \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Направление (специальность)\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
Кафедра \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Дисциплина\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

|  |  |
| --- | --- |
|  | Утверждаю  Зав. кафедрой\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
|  | «\_\_\_\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_20 \_\_\_ г. |

**Задание**

**на курсовую работу (проект)**

Студент\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(фамилия, имя, отчество)

Группа\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. Тема: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Утверждена приказом от «\_\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_ 20\_\_\_ г. № \_\_\_\_\_\_\_\_\_

2. Срок представления работы (проекта) к защите «\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_20\_\_ г.

3. Содержание расчетно-пояснительной записки: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

4. Перечень графического материала: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

5. Дата выдачи задания «\_\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_20 \_\_\_ г.

Руководитель работы (проекта)\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

подпись, дата инициалы и фамилия

Задание принял к исполнению\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

подпись, дата инициалы и фамилия

**1 Лабораторная работа:**

**Ход выполнения работы**

* **Выбор темы:**

Я выбрал тему баскетбольный клуб, так как я люлблю этот спорт, а так же довольно неплохо в нем разбираюсь с точки зрения организации.

* **Требования к приложению:**
* **Аналетические**

Пользователь должен иметь возможность просматривать:

1. статистику игроков за сезон(очки за игру, подборы за игру и т..д)
2. статистику команды в целом и клуба(количество побед, поражений а так же прибыль от билетов и д.р)

* **Операционные**

По запросу пользователя приложение должно выводить:

1. Генерального менеджера, главного тренера, список всего персонала клуба, игроков,арену.
2. Контракты как игроков так и персонала, и расписание для команды на конкретный сезон(тренировки,игры и т.д).
3. Просматривать данные об игроках(история травм,данные(рост, вес, позиция и т.д)).
4. У пользователя должна быть возможность добавить/удалить/редактировать расписание сезона, игроков, персонала, статистика, личную информацию и т.д.

* **Список сущностей:**
* Баскетбольный клуб
* Игроки
* Сезонный Календарь
* Персонал
* Травма
* Контракты
* Арена
* Генеральный менеджер
* Главный тренер
* Статистика игрока
* Сезонная статистика
* Элемент статистики(общей)
* Элемент статистики(игрока)

**Список атрибутов:**

**Баскетбольный клуб:** Дата основания (Date),Рыночная стоимость (int), Лига (String), количество титулов чемпиона(int).

**Игроки:** Имя (String), Фамилия (String),отчество(String), Начало карьеры(date),позиция(String),возраст(int), рост(int), Размах рук(int), начало карьеры(Date) .

**Персонал:** Опыт работы(лет)(int), Имя (String), Фамилия (String), прошлая работа(string).

**Главный тренер:** тоже что и у персонала,процент побед(double), количество титулов(int), Кол-во тренированных команд(int).

**Генеральный менеджер:** тоже что и у персонала

**Сезонный календарь:** даты матчей(date), тренировки(date),номер сезона (int).

**Травма:** тяжесть травмы (enum), Время реабилитации (Date), дата получения последней травмы (Date), название травмы(String).

**Статистика игрока:** -

**Контракты:** размер оплаты труда(int), вид контракта(enum), защитные опции (String), Срок действия контракта (Date), дата подписания(Date).

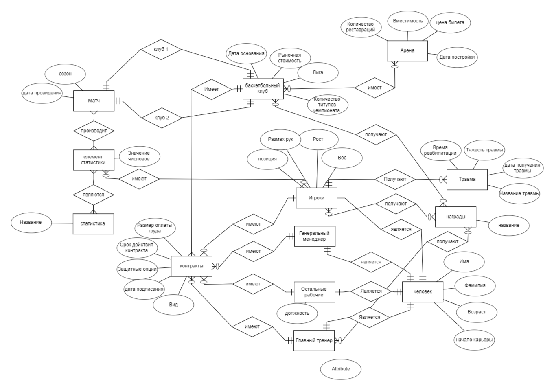
**Арена:** вместимость(int), цена билета(int), кол-во рестовраций(int), дата постройки(Date).

**Сезонная статистика: -**

**Элемент статистики(как игрока так и общая):** значение(double or int), название(String).

* **Связь сущностей:**
* Баскетбольным клубом управляет один генеральный менеджер и при этом он пренадлежит одному клубу.
* Баскетбольный клуб имеет много сезонных календарей и при этом каждый сезонный календарь пренадлежит одному клубу.
* Баскетбольный клуб имеет несколько арен(или одну) и при этом у адной арены может быть несколько клубов.
* Сезонный календарь формирует много или одну сезонную статистику и при этом у каждой статистики может быть один сезонный календарь.
* В сезонной статистике может быть много обшей статистики.
* Сезонная статистика имеет много статистик игроков и при этом каждый статистика игрока принадлежит к одному сезону.
* Генеральный менеджер выдает контракты и при этом у каждого контракта один ген. менеджер.
* Генеральный менеджер назначает много персонала и при этом каждый член персонала относится одному генеральному менеджеру.
* Главный тренер состоит в персонале и является единственным в своем роде.
* Игроки имеют много травм и при этом каждая травма принадлежит одному игроку.
* Игроки подписывают много контрактов и при этом каждый контракт принадлежит одному игроку.
* Игрок имеет много сезонной статистики и при этом каждая сезонная статистика принадлежит определенному игроку.

Построить ER- диаграмму

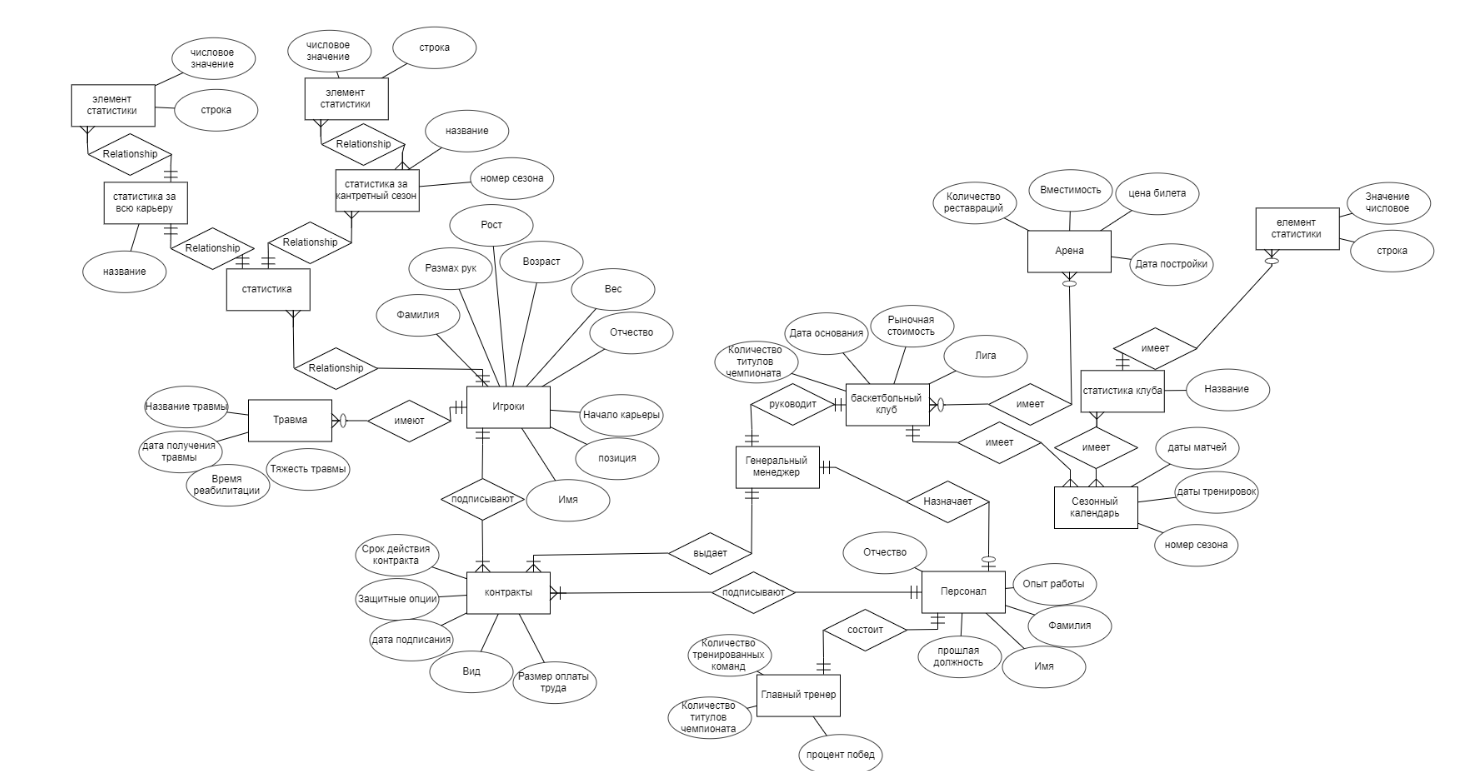


**2 Лабораторная работа:**

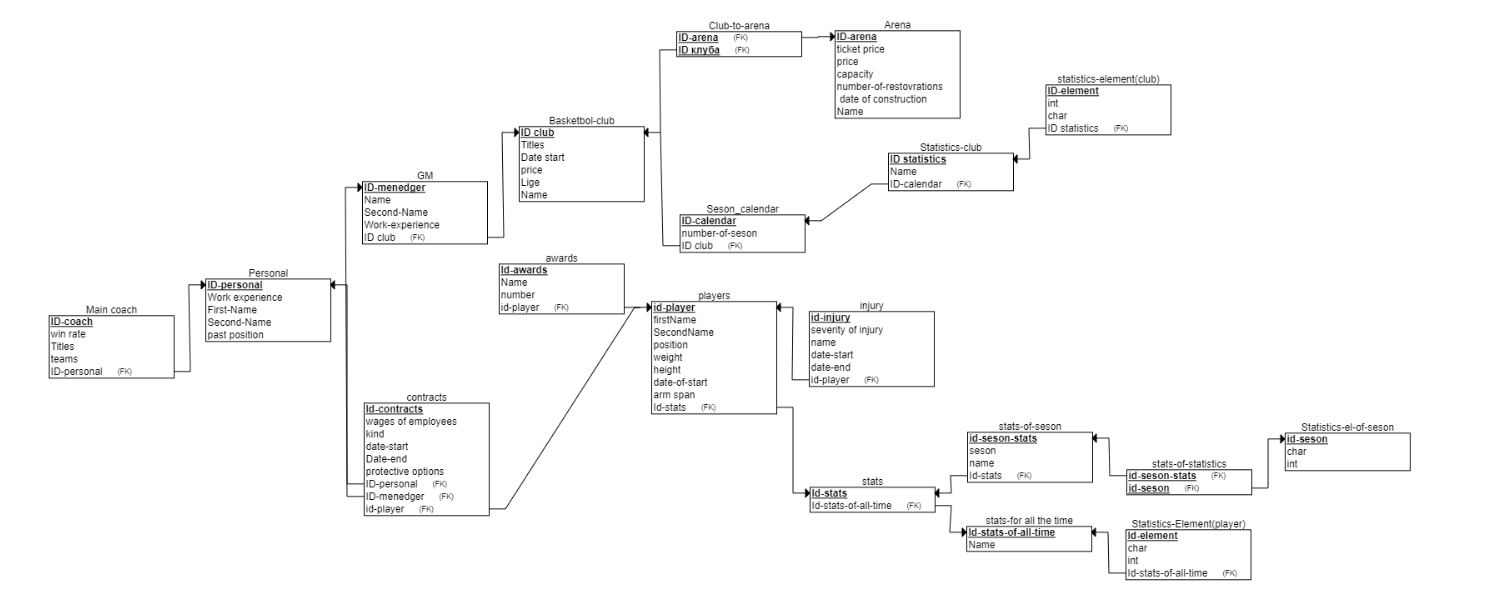
**Ход Выполнения работы**

1. по своей ER-диаграмме построить реляционную схему:

**Исходная ER – диаграмма:**



**Построенная на ее основе реляционная схема:**

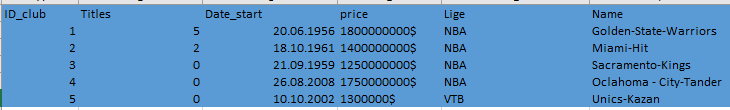


1. **провести нормализацию**

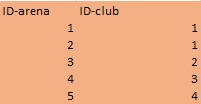
Построенная реляционная схема соответствует 1-3 нормальным формам

1. Придумать данные для каждой таблицы (минимум по 2)

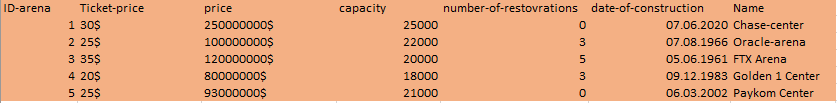
Basketball -club



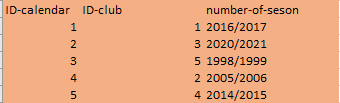
Club-to-arena



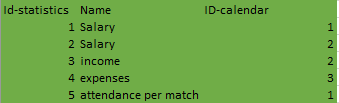
Arena



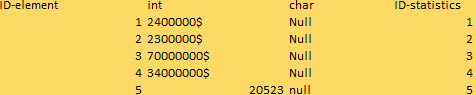
Season-calendar



Statistics-club



Statistics-element(club)



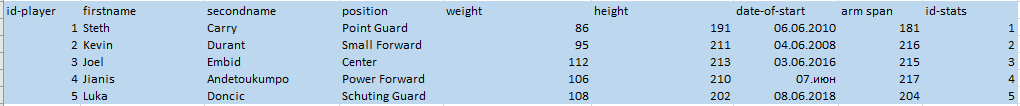
GM



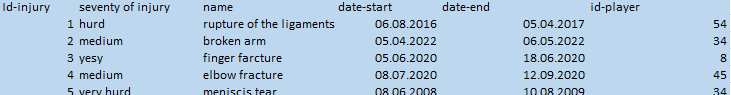
Personal



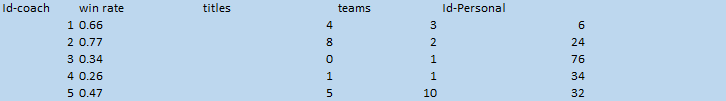
Players



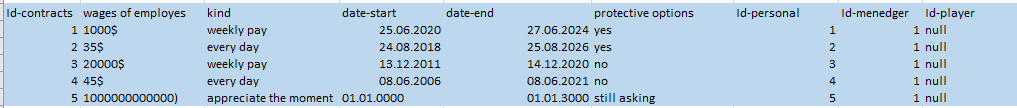
injures



Main-coach



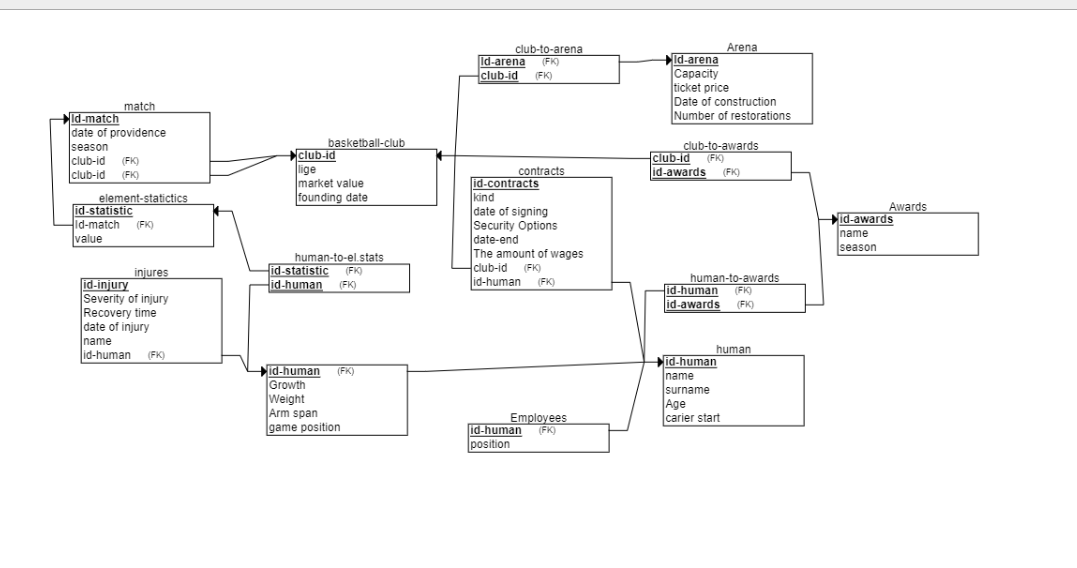
Contracts



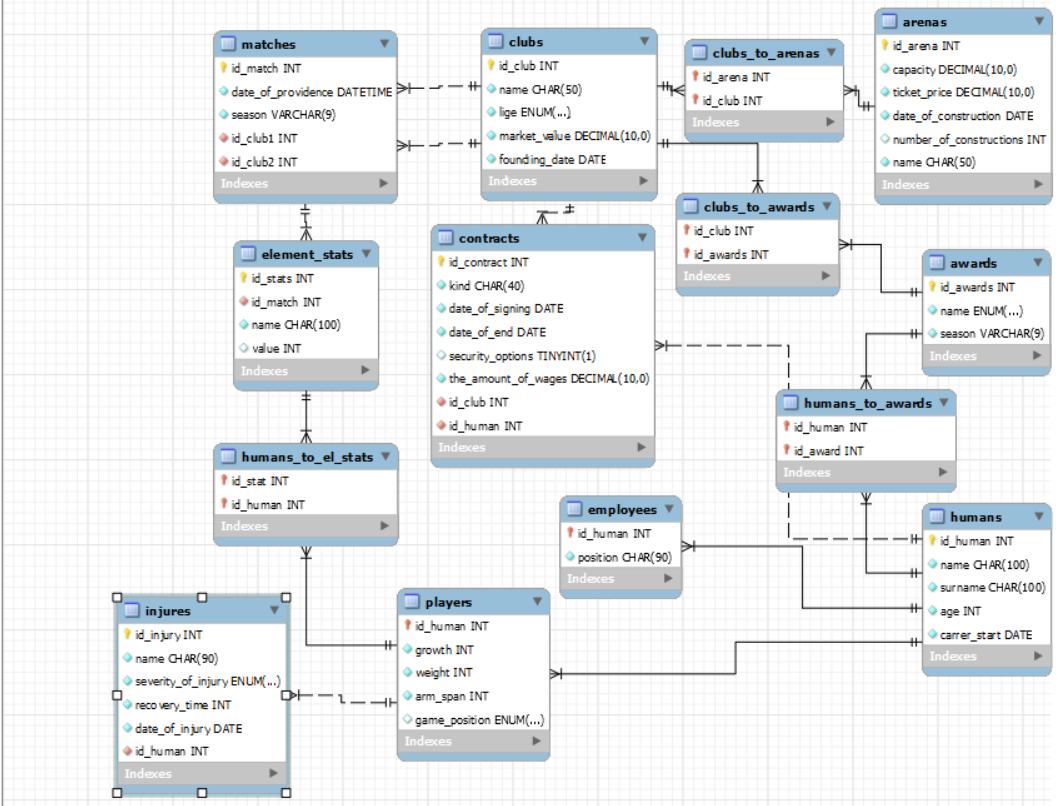
**3 лабораторная работа:**

**Часть 1.**

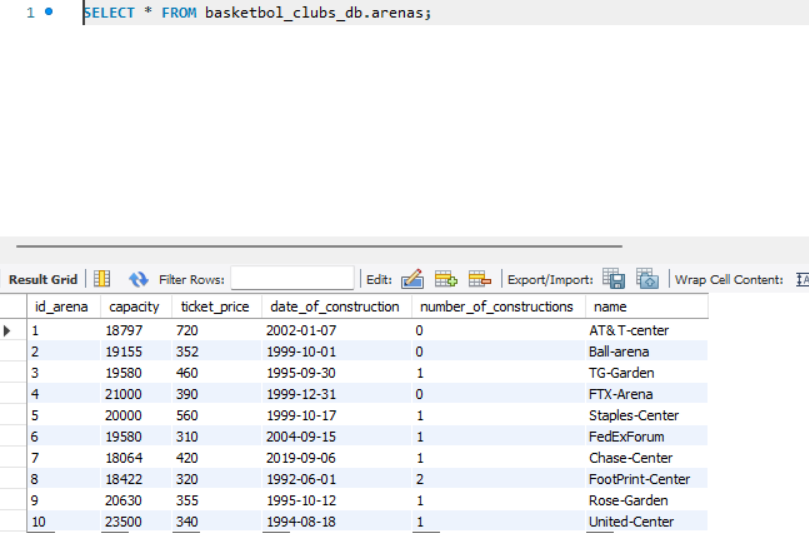
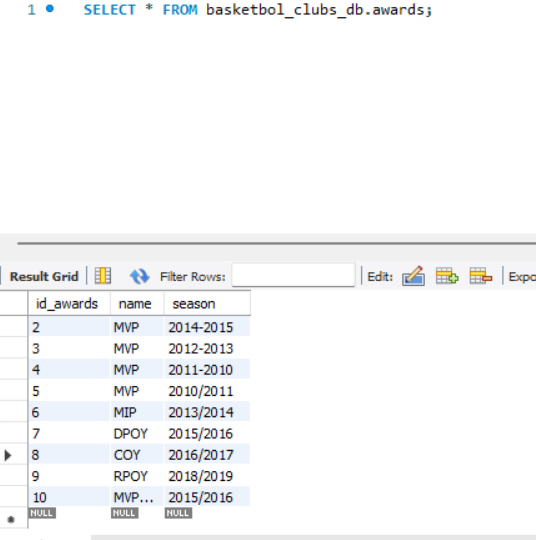
**Скриншот схемы, разработанной на 2 лабораторной работе.**



**Схема, сгенерированная на основе созданной БД в MySQL Workbench.**



**Наполненные таблицы:**



**2 часть . Создать бд однагрупника**

create database Schools\_db;

use Schools\_db;

CREATE TABLE schools (

id INT NOT NULL PRIMARY KEY AUTO\_INCREMENT,

`Name` CHAR(20) NOT NULL,

HeadMaster CHAR(80) NOT NULL,

District CHAR(80) NOT NULL,

Adress CHAR(80) NOT NULL

);

CREATE TABLE positions (

id INT NOT NULL PRIMARY KEY AUTO\_INCREMENT,

Positions CHAR(80) NOT NULL

);

CREATE TABLE humans (

id INT NOT NULL PRIMARY KEY AUTO\_INCREMENT,

`Name` CHAR(80) NOT NULL,

PhoneNumber CHAR(11) NOT NULL,

SecondName CHAR(80) NOT NULL,

Patronymic CHAR(80) NOT NULL,

CovidStatus BOOLEAN,

Mail CHAR(100),

Birthday DATE NOT NULL

);

CREATE TABLE workers (

id\_Human INT NOT NULL,

Adress CHAR(80),

FOREIGN KEY (id\_Human)

REFERENCES humans (id)

);

CREATE TABLE parents (

id\_Human INT NOT NULL,

Job CHAR(80),

AdditionalMail CHAR(100),

AdditionalPhoneNumber CHAR(11),

FOREIGN KEY (id\_Human)

REFERENCES humans (id)

);

CREATE TABLE nameOfObject (

id INT NOT NULL PRIMARY KEY AUTO\_INCREMENT,

titles CHAR(100) NOT NULL

);

CREATE TABLE Subjects (

id INT NOT NULL PRIMARY KEY AUTO\_INCREMENT,

Grade ENUM('0', '1', '2', '3', '4', '5'),

ProgrammFile CHAR(100) NOT NULL,

id\_NameOfObject INT NOT NULL,

FOREIGN KEY (id\_NameOfObject)

REFERENCES nameOfObject (id)

);

CREATE TABLE contracts (

id INT NOT NULL PRIMARY KEY AUTO\_INCREMENT,

Salary INT NOT NULL,

id\_Worker INT NOT NULL,

id\_Position INT NOT NULL,

id\_School INT NOT NULL,

FOREIGN KEY (id\_Worker)

REFERENCES workers (id\_Human),

FOREIGN KEY (id\_Position)

REFERENCES positions (id),

FOREIGN KEY (id\_School)

REFERENCES schools (id)

);

CREATE TABLE subjectsToContracts (

id\_Subject INT NOT NULL,

id\_Cotract INT NOT NULL,

FOREIGN KEY (id\_Subject)

REFERENCES Subjects (id),

FOREIGN KEY (id\_Cotract)

REFERENCES contracts (id)

);

CREATE TABLE classrooms (

id INT NOT NULL PRIMARY KEY AUTO\_INCREMENT,

Floor INT NOT NULL,

`Number` INT NOT NULL,

AmountOfSeats INT NOT NULL,

id\_School INT NOT NULL,

id\_NameOfSubject INT NOT NULL,

FOREIGN KEY (id\_School)

REFERENCES schools (id),

FOREIGN KEY (id\_NameOfSubject)

REFERENCES nameOfObject (id)

);

CREATE TABLE classes (

id INT NOT NULL PRIMARY KEY AUTO\_INCREMENT,

Letter CHAR(80) NOT NULL,

BeginYear DATE NOT NULL,

id\_School INT NOT NULL,

id\_Cotract INT NOT NULL,

FOREIGN KEY (id\_School)

REFERENCES schools (id),

FOREIGN KEY (id\_Cotract)

REFERENCES contracts (id)

);

CREATE TABLE students (

id\_Human INT NOT NULL PRIMARY KEY,

id\_Class INT NOT NULL,

FOREIGN KEY (id\_Human)

REFERENCES humans (id),

FOREIGN KEY (id\_Class)

REFERENCES classes (id)

);

CREATE TABLE studentsToParents (

id\_Parent INT NOT NULL,

id\_Student INT NOT NULL,

FOREIGN KEY (id\_Parent)

REFERENCES parents (id\_Human),

FOREIGN KEY (id\_Student)

REFERENCES students (id\_Human)

);

CREATE TABLE lessons (

id INT NOT NULL PRIMARY KEY AUTO\_INCREMENT,

`Day` DATE NOT NULL,

LessonNumber ENUM('1', '2', '3', '4', '5', '6', '7', '8', '9', '10', '11', '12'),

id\_Classroom INT NOT NULL,

id\_Subject INT NOT NULL,

id\_Class INT NOT NULL,

id\_Contract INT NOT NULL,

FOREIGN KEY (id\_Classroom)

REFERENCES classrooms (id),

FOREIGN KEY (id\_Subject)

REFERENCES subjects (id),

FOREIGN KEY (id\_Class)

REFERENCES classes (id),

FOREIGN KEY (id\_Contract)

REFERENCES contracts (id)

);

CREATE TABLE ratings (

id INT NOT NULL PRIMARY KEY AUTO\_INCREMENT,

Mark ENUM('0', '1', '2', '3', '4', '5') NOT NULL,

Note CHAR(200),

id\_Student INT NOT NULL,

FOREIGN KEY (id\_Student)

REFERENCES students (id\_Human)

);

rename table nameofobject to nameOfSubjects;

rename table workers to Worker;

rename table Worker to workers;

ALTER TABLE `schools\_db`.`workers`

ADD PRIMARY KEY (`id\_Human`);

;

ALTER TABLE `schools\_db`.`subjectstocontracts`

ADD PRIMARY KEY (`id\_Subject`, `id\_Cotract`);

;

ALTER TABLE `schools\_db`.`subjectstocontracts`

DROP PRIMARY KEY,

ADD PRIMARY KEY (`id\_Subject`);

;

ALTER TABLE `schools\_db`.`subjects`

ADD COLUMN `date` VARCHAR(45) NULL AFTER `id\_NameOfObject`;

ALTER TABLE `schools\_db`.`subjects`

DROP COLUMN `date`;

ALTER TABLE `schools\_db`.`positions`

ADD CONSTRAINT `fk1`

FOREIGN KEY (`id`)

REFERENCES `schools\_db`.`classes` (`id`)

ON DELETE NO ACTION

ON UPDATE NO ACTION;

ALTER TABLE `schools\_db`.`positions`

DROP FOREIGN KEY `fk1`;

ALTER TABLE `schools\_db`.`positions`

CHANGE COLUMN `Positions` `Positions` CHAR(80) NULL ;

ALTER TABLE `schools\_db`.`positions`

CHANGE COLUMN `Positions` `Positions` CHAR(80) NOT NULL ;

ALTER TABLE `schools\_db`.`subjects`

DROP FOREIGN KEY `subjects\_ibfk\_1`;

ALTER TABLE `schools\_db`.`subjects`

CHANGE COLUMN `id\_NameOfObject` `id\_NameOfSubject` INT NOT NULL ;

ALTER TABLE `schools\_db`.`subjects`

ADD CONSTRAINT `subjects\_ibfk\_1`

FOREIGN KEY (`id\_NameOfSubject`)

REFERENCES `schools\_db`.`nameofsubjects` (`id`);

use schools\_db;

INSERT INTO `schools\_db`.`humans` (`id`, `Name`, `PhoneNumber`, `SecondName`, `Patronymic`, `CovidStatus`, `Mail`, `Birthday`)

VALUES ('1', 'Valera', '89044296481', 'Petrov', 'Sergeevich', '1', 'TreyYangMVP13rus@gmail.com', '2002.10.20');

INSERT INTO `schools\_db`.`positions` (`Positions`)

VALUES ('TEACHER');

INSERT INTO `schools\_db`.`humans` (`id`, `Name`, `PhoneNumber`, `SecondName`, `Patronymic`, `CovidStatus`, `Mail`, `Birthday`)

VALUES ('2', 'Valera', '89044296481', 'Petrov', 'Sergeevich', '1', 'TreyYangMVP13rus@gmail.com', '2002.10.20'),

('3', 'Valera', '89044296481', 'Ivan', 'Sergeevich', '1', 'TreyYangMVP13rus@gmail.com', '2002.10.20'),

('4', 'Valera', '89044296481', 'Grichs', 'Sergeevich', '1', 'TreyYangMVP13rus@gmail.com', '2002.10.20'),

('5', 'Valera', '89044296481', 'Petr', 'Sergeevich', '1', 'TreyYangMVP13rus@gmail.com', '2002.10.20'),

('6', 'Valera', '89044296481', 'Roven', 'Sergeevich', '1', 'TreyYangMVP13rus@gmail.com', '2002.10.20'),

('7', 'Valera', '89044296481', 'Petrov', 'Sergeevich', '1', 'TreyYaP13rus@gmail.com', '2002.10.20'),

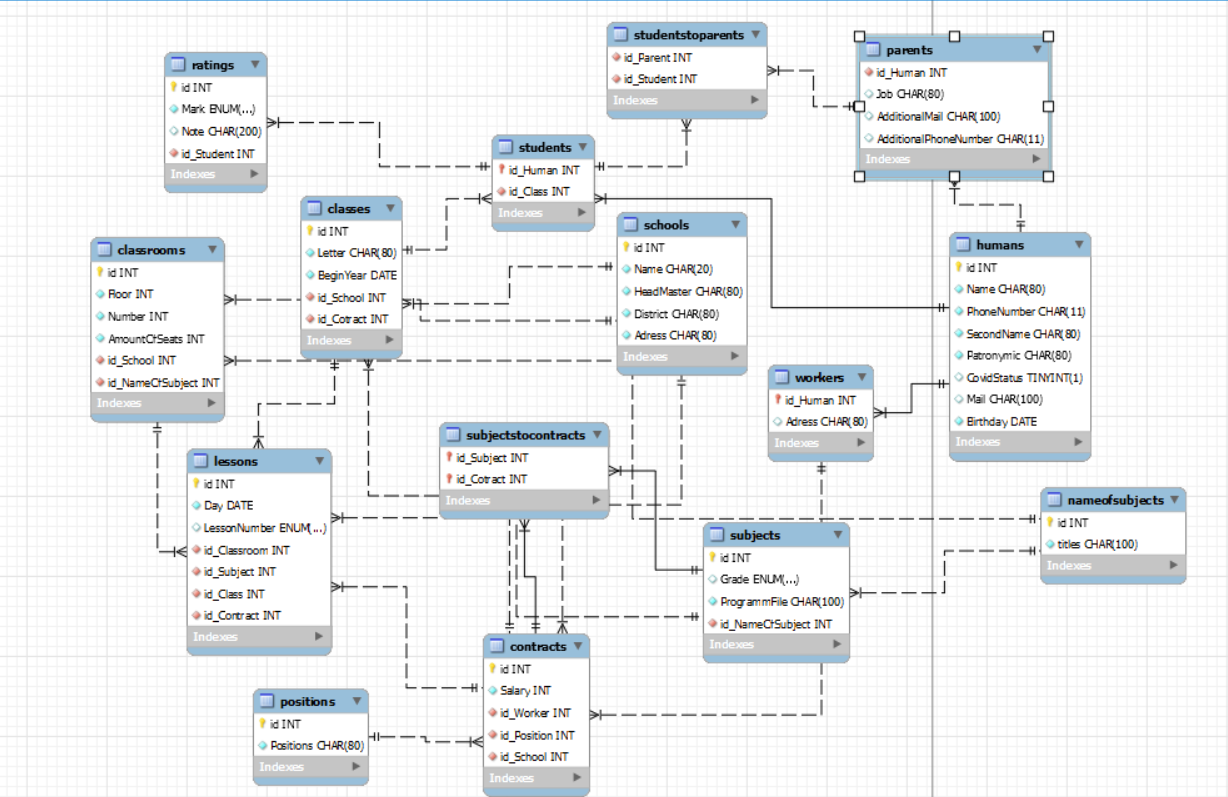
('8', 'Valera', '89044296481', 'Petrov', 'Sergeevich', '1', 'YangMVP13rus@gmail.com', '2002.10.20'),

('9', 'Valera', '89044296481', 'Petrov', 'Sergeevich', '1', 'YangMVP13rus@gmail.com', '2002.10.20'),

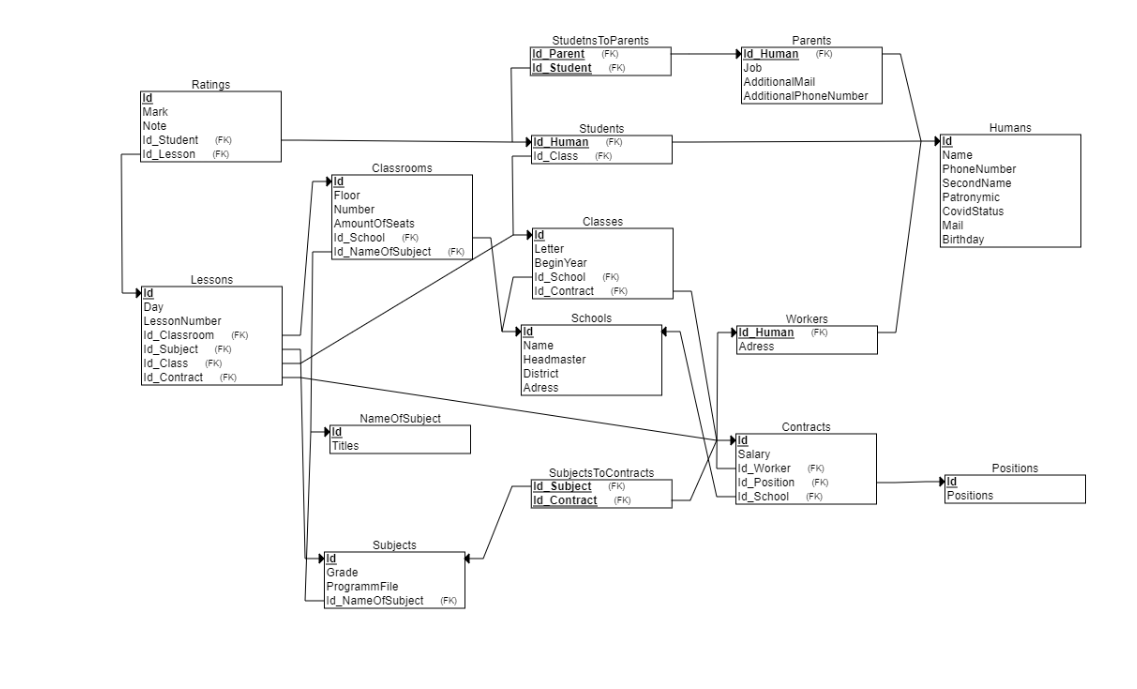
('10', 'Valera', '89044296481', 'Petrov', 'Sergeevich', '1', 'TreyYangMVP13rus@gmail.com', '2002.10.20')

;

**Схема, сгенерированная на основе созданной БД в MySQL Workbench.**



**Реляционная схема Плотникова Тимофея ИВТ 260:**



**4 Лабораторная работа:**

1. **UPDATE в разных таблицах, с WHERE, можно условно, например, изменить заранее созданные некорректные данные (5 шт.)**

UPDATE arenas

SET

date\_of\_construction = '2001-12-3'

WHERE

(id\_arena = '5');

UPDATE contracts

SET

kind = 'back',

date\_of\_signing = '2012-12-21'

WHERE

(id\_contract = '4');

UPDATE contracts

SET

kind = 'front',

security\_options = '1',

id\_club = '3'

WHERE

(id\_contract = '8');

UPDATE players

SET

growth = '2012',

weight = '97'

WHERE

(id\_human = '1');

UPDATE players

SET

arm\_span = '208',

game\_position = 'SF'

WHERE

(id\_human = '4');

1. **DELETE в разных таблицах, с WHERE, можно условно, например, удалить заранее созданные некорректные данные (5 шт.)**

DELETE FROM humans

WHERE

(id\_human = '11');

DELETE FROM humans

WHERE

(id\_human = '12');

DELETE FROM contracts

WHERE

(id\_contract = '11');

DELETE FROM contracts

WHERE

(id\_contract = '12');

DELETE FROM humans

WHERE

(id\_human = '11');

1. **SELECT, DISTINCT, WHERE, AND/OR/NOT, IN, BETWEEN, различная работа с датами и числами, преобразование данных, IS NULL, AS для таблиц и столбцов и др. в различных вариациях (15 шт. +)**

SELECT

\*

FROM

humans

WHERE

age BETWEEN 25 AND 40;

SELECT

\*

FROM

contracts

WHERE

date\_of\_signing BETWEEN '2015-01-01' AND '2020-01-01';

SELECT

\*

FROM

awards

WHERE

`name` = 'MVP' OR `name` = 'DPOY';

SELECT

\*

FROM

injures

WHERE

date\_of\_injury NOT BETWEEN '2009-01-01' AND '2016-01-01';

SELECT

\*

FROM

arenas

WHERE

ticket\_price > 350

AND number\_of\_constructions >= 1;

SELECT

\*

FROM

awards

WHERE

`name` = 'MVP'

AND (season = '2015-2016'

OR season = '2017-2018');

select\*from employees

where not `position` = 'trener' and not `position` = 'GM';

SELECT

\*

FROM

players

WHERE

growth BETWEEN '195' AND '206'

AND game\_position = 'SF';

SELECT

\*

FROM

humans

WHERE

`name` LIKE 'K%'

AND carrer\_start BETWEEN '2010-01-01' AND '2014-01-01';

select\*from humans

where carrer\_start between '2006-01-01' and '2011-01-01';

SELECT

\*

FROM

contracts;

SELECT

\*

FROM

players;

SELECT

\*

FROM

humans;

SELECT

\*

FROM

contracts

WHERE

date\_of\_signing

AND date\_of\_end BETWEEN '2004-01-01' AND '2016-01-01';

SELECT

\*

FROM

injures;

SELECT

\*

FROM

injures

WHERE

recovery\_time BETWEEN '60' AND '140';

1. **LIKE и другая работа со строками (5-7 шт.+)**

SELECT

`name`, founding\_date

FROM

clubs

WHERE

`name` LIKE 'B%';

SELECT

`name`, surname, age

FROM

humans

WHERE

surname LIKE 'D%' AND age < 50;

SELECT

\*

FROM

arenas

WHERE

`name` REGEXP 'center'

AND capacity < '20000';

SELECT

\*

FROM

injures

WHERE

`name` REGEXP '(п|П)ерелом';

SELECT

\*

FROM

injures

WHERE

`name` REGEXP 'ankl';

1. **SELECT INTO или INSERT SELECT, что поддерживается СУБД (2-3 шт.). Для использования запроса INSERT SELECT вначале можно создать новую тестовую таблицу или несколько, в которые будут скопированы данные из существующих таблиц с помощью данного запроса. Код создания таблиц также приложить в лабораторную работу**

CREATE TABLE copy\_injures (

id INT NOT NULL AUTO\_INCREMENT PRIMARY KEY,

`name` CHAR(100) NOT NULL,

severity\_of\_injury ENUM('high', 'very-high', 'low', 'very-low', 'medium', 'indefinitely') NOT NULL,

recovery\_time INT NOT NULL,

date\_of\_injury DATE NOT NULL,

id\_human INT NOT NULL

)

insert into copy\_injures

select \* from injures

where `name` like "ankl%";

insert into copy\_injures

select \* from injures

where severity\_of\_injury = 'high';

insert into copy\_injures

select \*from injures

where recovery\_time < '20';

SELECT

\*

FROM

copy\_injures;

1. **JOIN: INNER, OUTER (LEFT, RIGHT, FULL), CROSS, NATURAL, разных, в различных вариациях, несколько запросов с более, чем одним JOIN (15 шт.+)**

SELECT

\*

FROM

humans a

JOIN

players b ON a.id\_human = b.id\_human;

SELECT

\*

FROM

humans

NATURAL JOIN

players;

SELECT

\*

FROM

humans a

RIGHT JOIN

employees b ON a.id\_human = b.id\_human;

SELECT

\*

FROM

humans a

LEFT JOIN

contracts b ON a.id\_human = b.id\_human;

SELECT

\*

FROM

humans a

INNER JOIN

contracts b ON a.id\_human = b.id\_human;

SELECT

\*

FROM

clubs a

CROSS JOIN

contracts b ON a.id\_club = b.id\_club;

SELECT

\*

FROM

injures

NATURAL JOIN

players;

SELECT

\*

FROM

humans

NATURAL JOIN

humans\_to\_awards;

SELECT

\*

FROM

awards a

RIGHT JOIN

clubs\_to\_awards b ON a.id\_awards = b.id\_awards;

use `basketbol\_clubs\_db`;

SELECT

\*

FROM

arenas

NATURAL JOIN

clubs\_to\_arenas;

SELECT

\*

FROM

clubs a

JOIN

`matches` b ON a.id\_club = b.id\_club1;

SELECT

\*

FROM

humans a

RIGHT JOIN

injures b ON a.id\_human = b.id\_human;

SELECT

\*

FROM

humans

NATURAL JOIN

players;

SELECT

\*

FROM

players

NATURAL JOIN

`humans\_to\_el\_stats`;

SELECT

\*

FROM

players a

JOIN

`humans\_to\_el\_stats` b ON a.id\_human = b.id\_human

JOIN

`element\_stats` c ON b.id\_stat = c.id\_stats;

SELECT

\*

FROM

`humans` a

JOIN

`players` b ON a.id\_human = b.id\_human

JOIN

`injures` c ON b.id\_human = c.id\_human;

1. **GROUP BY**(некоторые с **HAVING**), с **LIMIT, ORDER BY**(**ASC**|**DESC**) вместе с **COUNT**, **MAX**, **MIN, SUM, AVG** в различных вариациях, можно по отдельности (**15 шт.+**)

select

game\_position,count(\*)

from players

group by game\_position;

select

growth,max(weight)

from players

group by weight;

select

severity\_of\_injury,avg(recovery\_time)

from `injures`

group by severity\_of\_injury;

select

kind,avg(the\_amount\_of\_wages)

from contracts

group by kind;

select

avg(age),`name`

from humans

group by `name`;

select `name`,avg(`value`)

from element\_stats

group by `name`;

select `name`,max(`value`)

from element\_stats

group by `name`;

select `name`,min(`value`)

from element\_stats

group by `name`;

select `name`,count(`value`)

from element\_stats

group by `name`;

select `name`,count(`name`)

from awards

group by `name`;

select `name`,avg(season)

from awards

group by `name`;

select \*

from players

order by weight desc;

select \*

from players

order by arm\_span;

select \*

from awards

group by `name`

having id\_awards between '5' and '10';

select

kind,avg(the\_amount\_of\_wages)

from contracts

group by kind

having kind = 'back' or kind = 'front';

select \*

from players

limit 6;

1. **UNION**, **EXCEPT**, **INTERSECT**, что поддерживается СУБД (**3-5 шт.**)

select \*

from players

union

select \*

from humans;

select \*

from humans

where `name` like 'K%'

union

select \*

from humans

where surname like 'M%';

select \*

from players

where growth > 195

union

select \*

from players

where arm\_span < 210;

select \*

from injures

where `name` like 'ankl%'

union

select \*

from injures

where severity\_of\_injury = 'medium';

1. Вложенные **SELECT**с **GROUP BY,** **ALL, ANY, EXISTS**(**3-5 шт.**)

select \*

from humans

where exists ( select \* from players where id\_human = humans.id\_human and game\_position = 'SF');

select `name`,surname

from humans

where id\_human = any(select id\_human from employees where position = 'Coach');

select \*

from injures

where id\_human > all(select id\_human from players where game\_position = 'PG');

select \*

from arenas

group by (select count(`name`) from arenas);

1. **GROUP\_CONCAT** и другие разнообразные функции SQL (**2-3 шт.**)

select id\_human,concat(`name`,' ',surname),length(concat(`name`,' ',surname))

from humans;

select id\_club,`name`,left(`name`,5)

from clubs;

select repeat(`name`,3)

from arenas;

1. Запросы с **WITH**(**2-3 шт.**)

-

1. Запросы со строковыми функциями СУБД, с функциями работы с датами временем (форматированием дат), с арифметическими функциями (**5-7 шт.**)

select avg(`value`)

from element\_stats;

select max(`date\_of\_construction`)

from arenas;

select lower(`name`)

from awards;

select concat(`name`,' ',surname),id\_human,age

from humans;

select count(recovery\_time)

from injures;

select \*

from humans

where (`name` like concat('%','ev','%') or surname like concat('%','om','%'));

1. Сложные запросы, входящие в большинство групп выше, т.е. **SELECT** ... **JOIN**... **JOIN** ... **WHERE**... **GROUP BY** ... **ORDER BY** ... **LIMIT** ...; (**5-7 шт. +**), можно написать больше вместо простых.

select \*

from players

where weight > (select avg(weight) from players);

select \*

from humans

where age > all(

select \*

from humans

where age);

select \*

from

(select id\_human,game\_position,`value`,name\_stat

from

( select id\_stat,id\_human,`value`,name\_stat

from `element\_stats` a

join `humans\_to\_el\_stats` b on a.id\_stats = b.id\_stat) a

natural join players) a

natural join humans;

select \*

from contracts

natural join humans

group by `kind`;

select\*

from (select \*

from employees

natural join humans) a

join contracts b on a.id\_human = b.id\_human

group by security\_options;

select\*

from (select \*

from employees

natural join humans) a

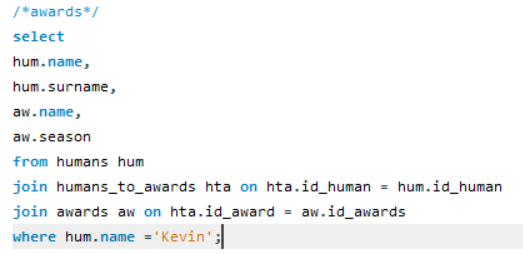
join contracts b on a.id\_human = b.id\_human

limit 3;

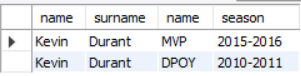
**5 Лабораторная работа:**

1. **Индексы**
2. Данный запрос выводит список наград у определненного игрока.

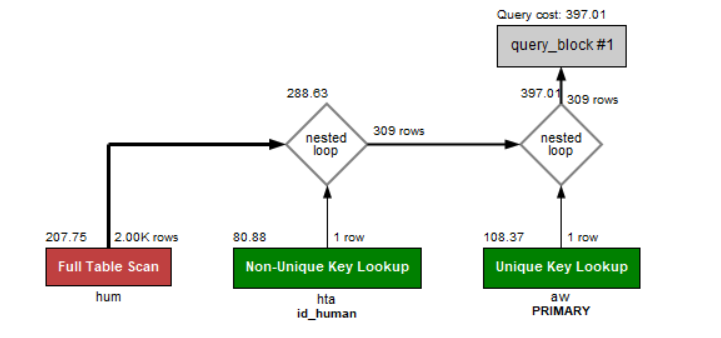
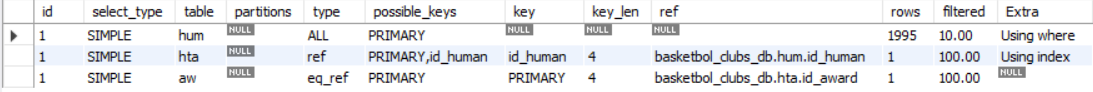
Запрос



Результат



explain

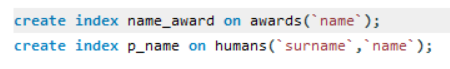


Время выполнения запроса.

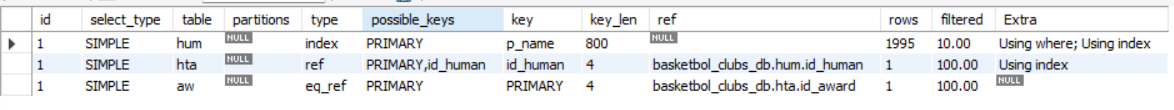
Пояснение первого индекса: Всего у наград может быть 6 имен, добавив индекс на имя награды я сокращю поиск в 6 раз. У наград нету друго атрибута, был бы более уникален и позволил бы ускорить поиск.



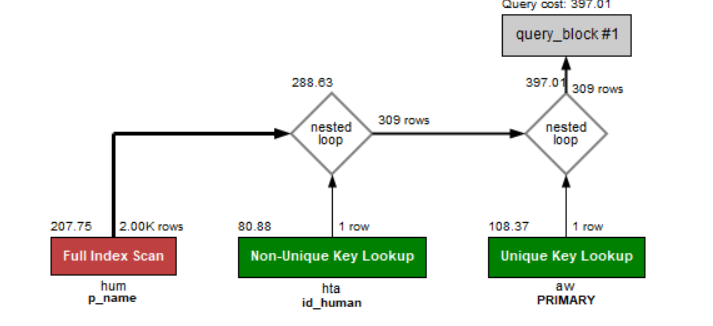
Пояснение второго индекса: составной индекс на фамилию, а потом имя.



Explain с индексами



Visual explain



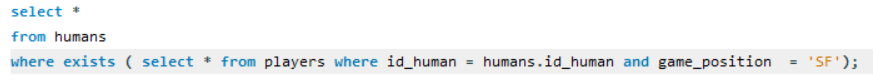
Время выполнения запроса



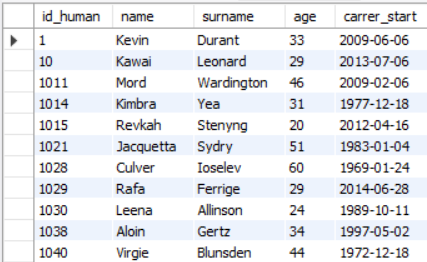
Вывод: В результате добавления индексов время выполнения уменьшилось.

2) Данный запрос выводит всех игроков с позицией “SF”

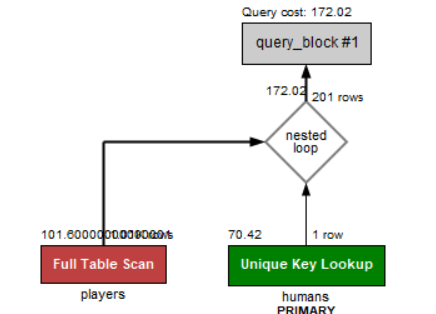
Запрос



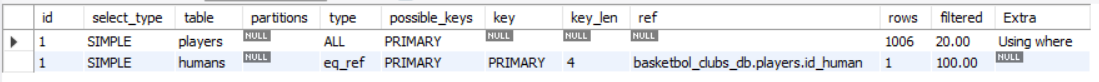
результат



Visual explain



Explain



Время выполнения

Вложенный селект



Джоин

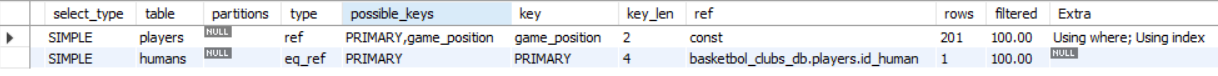


Индекс

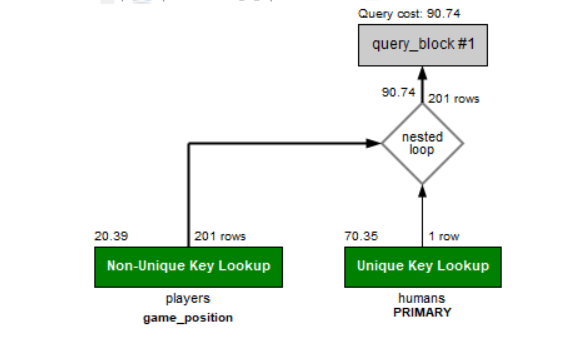
Пояснение: Дынный индекс мной добавлен поскольку игровых позиций у меня всего 5, поэтому при добавлении индекса на позицию игрока мы уменьшаем поиск в 5 раз.



Explain с индексами



Visual explain

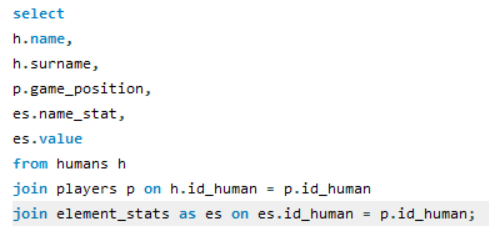


Время выполнения запроса

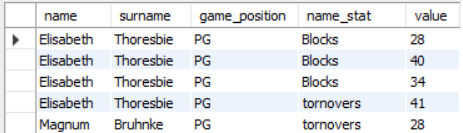


Вывод: В результате добавления индексов время выполнения уменьшилось.

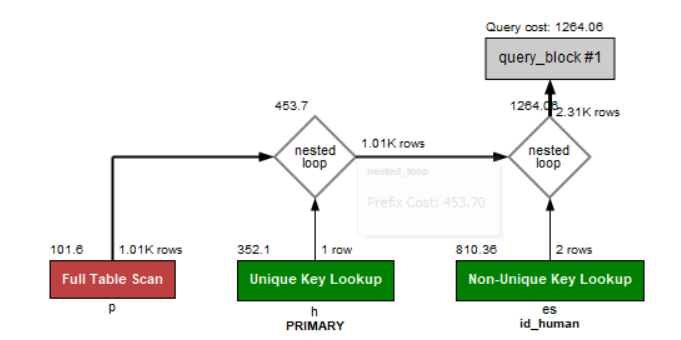
3) Данный запрос выводит статистику по игрокам.



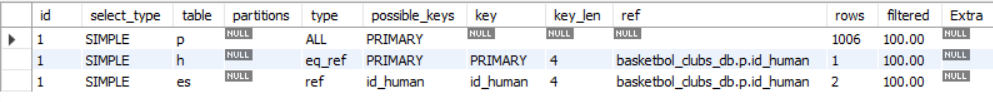
результат



Visual explain



Explain



Время выполнения

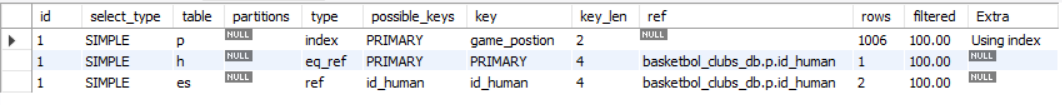


Индекс

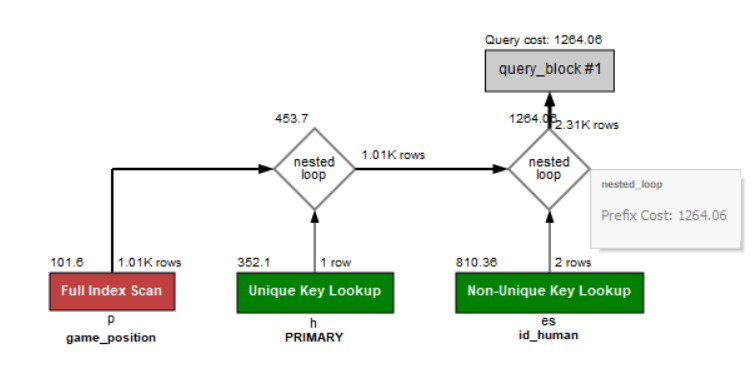
Пояснение: Дынный индекс мной добавлен поскольку игровых позиций у меня всего 5, поэтому при добавлении индекса на позицию игрока мы уменьшаем поиск в 5 раз.



Explain с индексами



Visual explain

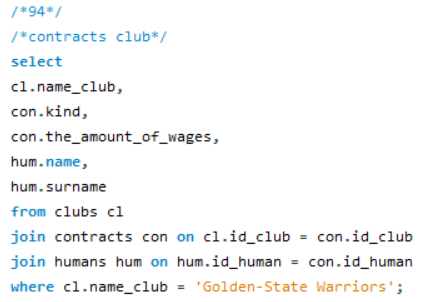


Время выполнения запроса

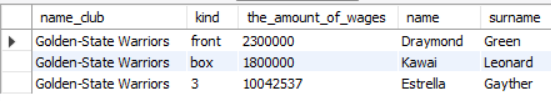


Вывод: В результате добавления индексов время выполнения уменьшилось.

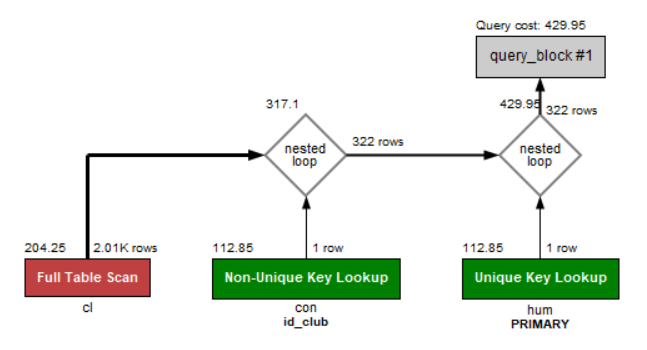
4) Данный запрос выводит список контрактов определенного клуба.



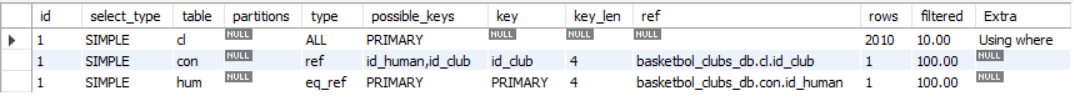
результат



Visual explain



Explain



Время выполнения

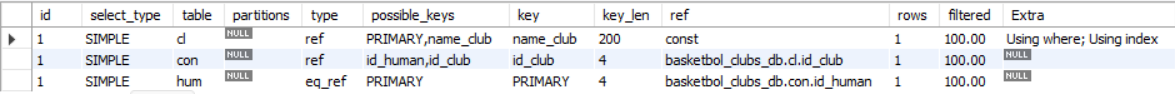


Индекс

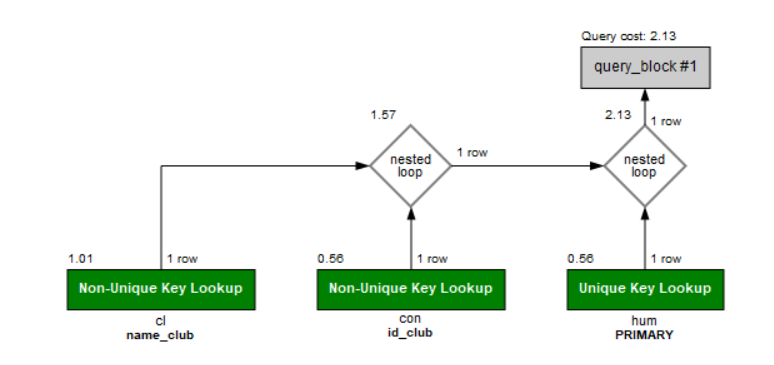
Пояснение: Дынный индекс по имени клуба.



Explain с индексами



Visual explain

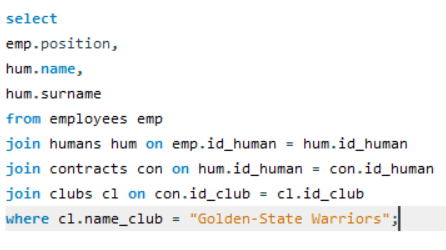


Время выполнения запроса



Вывод: В результате добавления индексов время выполнения уменьшилось.

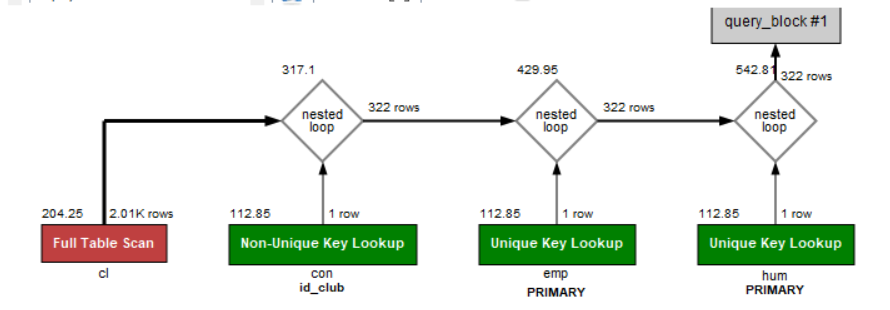
5) Данный запрос выводит список работников клуба.



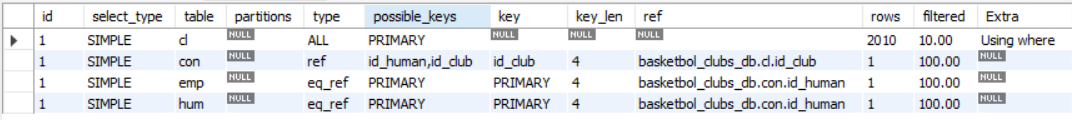
результат



Visual explain



Explain



Время выполнения

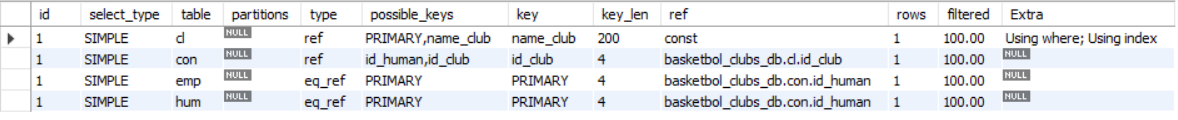


Индекс

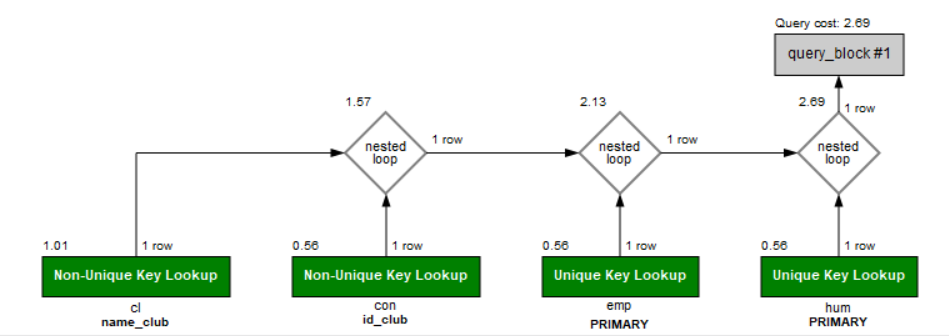
Пояснение: Дынный индекс по имени клуба.



Explain с индексами



Visual explain



Время выполнения запроса

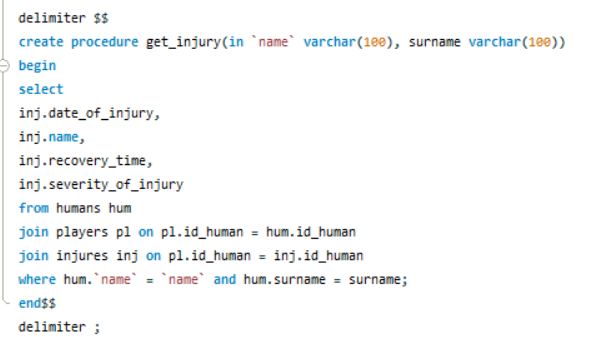


Вывод: В результате добавления индексов время выполнения уменьшилось.

**2) Процедуры и функции**

1. Процедура которая выводит травмы определенного игрока

Процедура:



Результат:

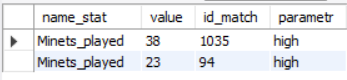


2)Процедура выводит заданый елемент статистики по определенному игроку и класифицирует показатели по значению.

Процедура:

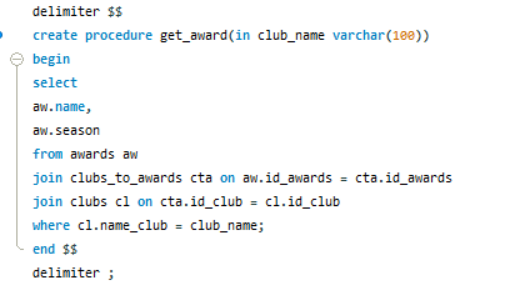


Реультат:



3) Проецуда выводит все награды по определенному клубу

Процедура:

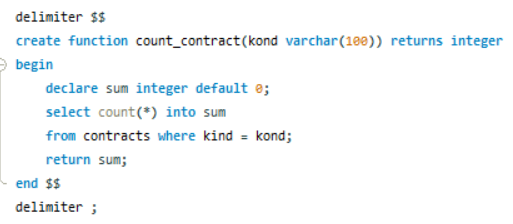


Результат:



4) Данная функция считает количество контрактов по типу контракта

Функция :

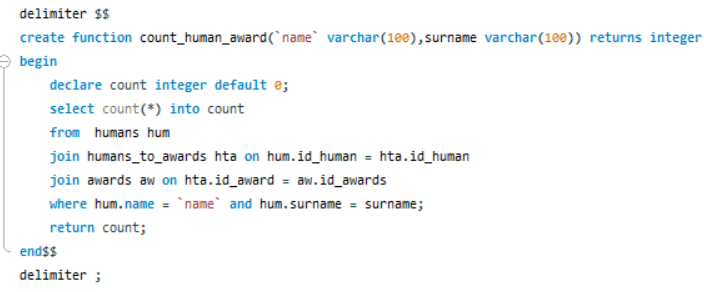


Результат :



5) Данная функция считает количество наград у человека

Функция:

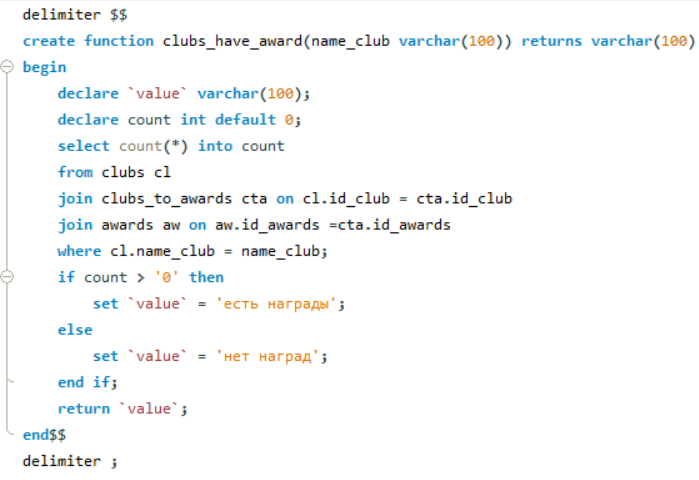


Результат:

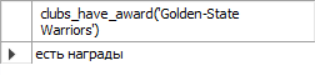


6)Данная функция показывает есть награды или нет.

Функция:



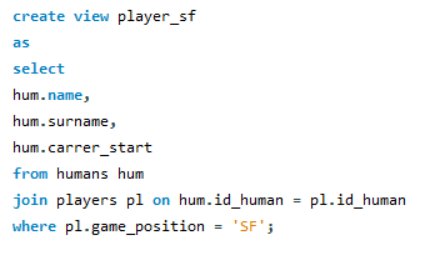
Результат:



**3) Представления**

1. Данное пердставление выводит всех игроков с позицией SF

Представление:

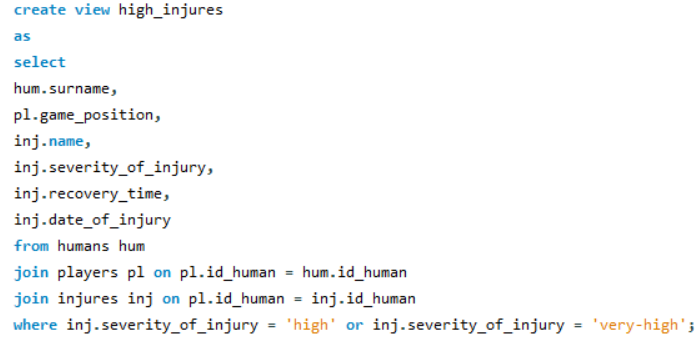


Результат:

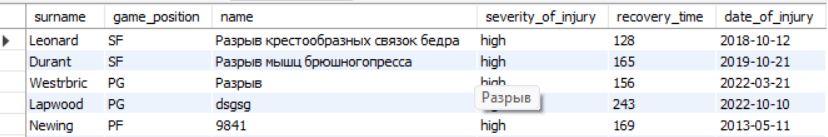


2) Представление выводит только тяжелые травмы у игроков

Представление:

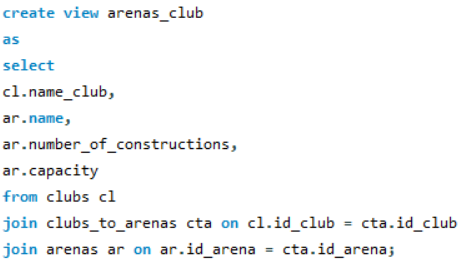


Результат:

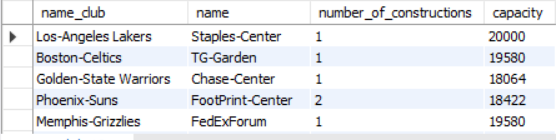


3) Данное представление выводит арены и клубы, которым эти арены пренадлежат.

Представление:



Результат:



**4) Тригеры**

Данный триггер срабатывает при добавлении в таблицу clubs новой строки и создает в таблице clubs\_ copy добавленную строку.

Тригер:

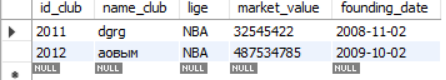


Результат:

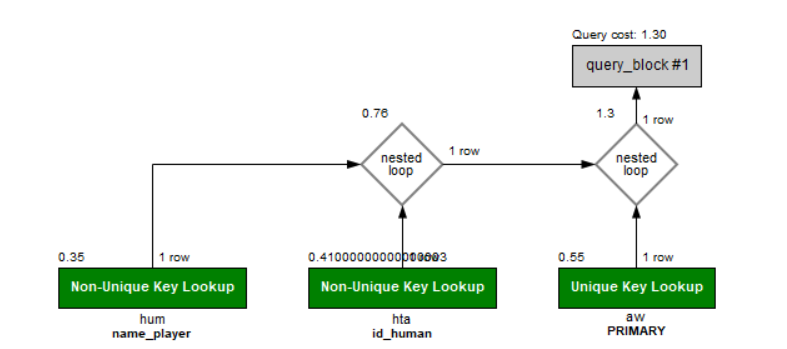
Добавляю в таблицу новые данные



Они сами копируются в другую таблицу



1. Cелект заменить на джоин и проверить результаты и Заменить первый индекс.



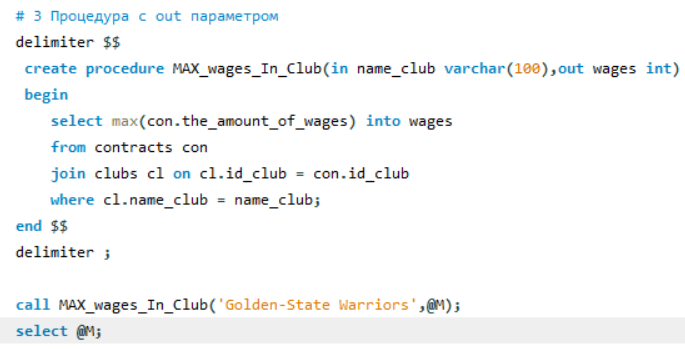
Вложенный селект



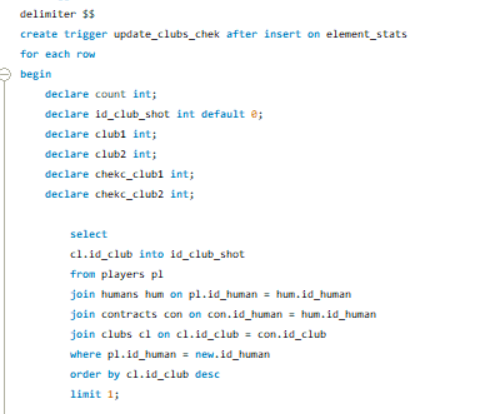
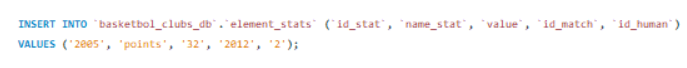
Джоин



1. Процедуру с out параметром. Процедура выводит максимальную зарплату в по определенному клубу.



1. Зделать триггер который обнавляет счет в матче при изменении статистики игрока.



**6-7 Лабораторная работа – курсовая работа:**

