**Московский авиационный институт**

**(Национальный исследовательский университет)**

Факультет прикладной математики и информационных технологий Кафедра вычислительной математики и программирования

**Курсовая работа**

По курсу «Базы данных»

Тема: Статистика турниров по киберспорту в дисциплине

Counter Strike

Студенты: Андреев А. О.,

Попов И. П.,

Семин А. В.

Группа: М8О-306Б-20

Преподаватель: Сеченых П.А.

Оценка:

Москва, 2022

Оглавление

[**Постановка задачи** 3](#_Toc121087649)

[**База данных** 4](#_Toc121087650)

[**Разработка модели в ERWIN** 7](#_Toc121087651)

[Логическая модель: 7](#_Toc121087652)

[Физическая модель: 8](#_Toc121087653)

[**Запросы** 9](#_Toc121087654)

[**Разработка процедур на PL/SQL** 14](#_Toc121087655)

[**Список использованных источников** 18](#_Toc121087656)

# **Постановка задачи**

*Постановка задачи*: разработать и реализовать

Базу Данных по индивидуальному заданию в инструментальной среде разработчика PostgreSQL (или СУБД на выбор)

***ОСНОВНЫЕ ТРЕБОВАНИЯ***

Разрабатываемая база данных должна содержать не менее шести таблиц, не менее двух из которых находятся в отношении «многие-ко-многим».

Отчёт по курсовой работе должен включать следующие разделы:

**Таблицы и схема базы данных (ER-модели)**

* Необходимо описать включенные в БД таблицы.
* При описании каждой таблицы необходимо указывать для каждого поля тип данных, а также другие характеристики (длину поля, маску ввода, значение по умолчанию и т.п.), для некоторых полей выполнить проверку правильности ввода исходных данных.
* Определить первичные ключи и ключи связи.
* Демонстрационных записей в каждой таблице 10–20.
* В связующих таблицах – 20–40.

**Запросы**

* В отчёте при описании каждого запроса приводится:
* назначение запроса,
* SQL – текст,
* результат выборки для заданного заполнения БД.
* Написать SQL-запросы с использованием
* операторов JOIN (LEFT, RIGHT, CROSS, FULL)
* выражения CASE
* оператора GROUP BY с расширениями (ROLLUP, CUBE, GROUPING)

**Разработка процедур на PL/SQL**

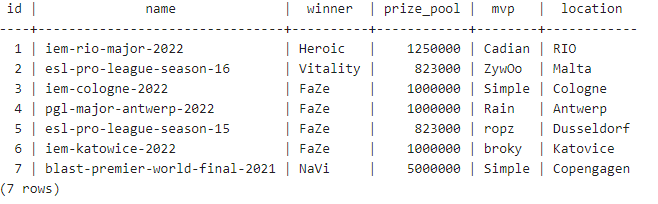
* Разработать процедуру, функцию и триггер на языке PL/SQL.
* Процедура/функция должна:
* принимать аргументы на вход;
* валидировать значения аргументов, бросать исключение, если значения некорректны;
* выполнять необходимое действие;
* если необходимо, выполнять пост-валидацию корректности данных после действия, в случае необходимости бросать исключение;
* обрабатывать исключения, возникающие во время выполнения.

**В отчёте при описании приводится:**

* назначение,
* SQL – текст,
* результат выполнения для заданного заполнения БД.

# **База данных**

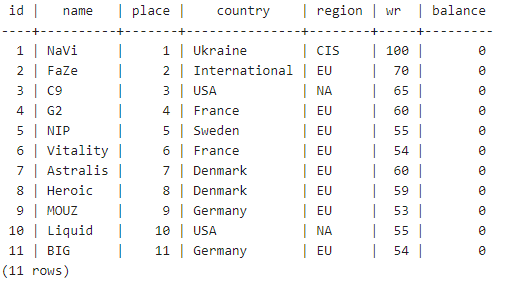
**Таблица Tournaments (турниры):**



Типы данных:

* Первичный ключ (serial)
* Название турнира (char)
* Победитель (char)
* Призовой фонд (integer)
* MVP (лучший игрок турнира по статистике) (char)
* Место проведения (char)

**Таблица Teams (команды):**



Типы данных:

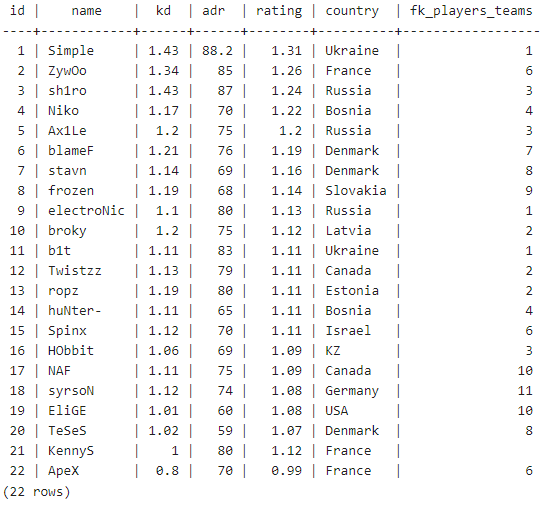
* Первичный ключ (serial)
* Название команды (char)
* Место в мировом рейтинге (integer)
* Тренер команды (char)
* Страна (char)
* Регион (char)
* Win rate (процент побед) (double)
* Balance (integer)

**Таблица Tournaments\_Teams:**

Типы данных:

* Первичный ключ
* Внешний ключ команды
* Внешний ключ турнира

**Таблица Players (игроки):**

****

Типы данных:

* Первичный ключ (serial)
* Имя игрока (char)
* KD (double)
* ADR (double)
* rating (double)
* Страна (char)
* Внешний ключ команды(integer)

**Таблица Players\_Backup (буфер в который отправляются старые данные об игроках после внесения в них изменений):**

Типы данных:

* Первичный ключ (serial)
* Имя игрока (char)
* KD (double)
* ADR (double)
* rating (double)
* Страна (char)
* Внешний ключ команды(integer)

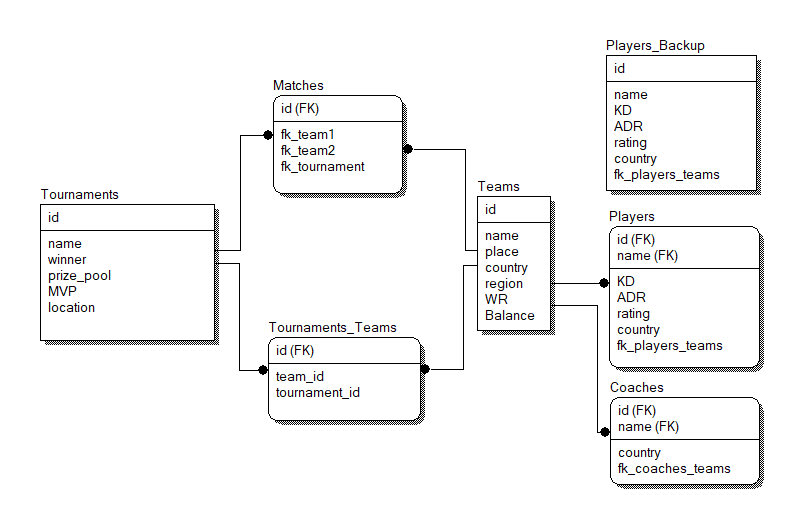
**Таблица Coaches(тренеры):**

Типы данных:

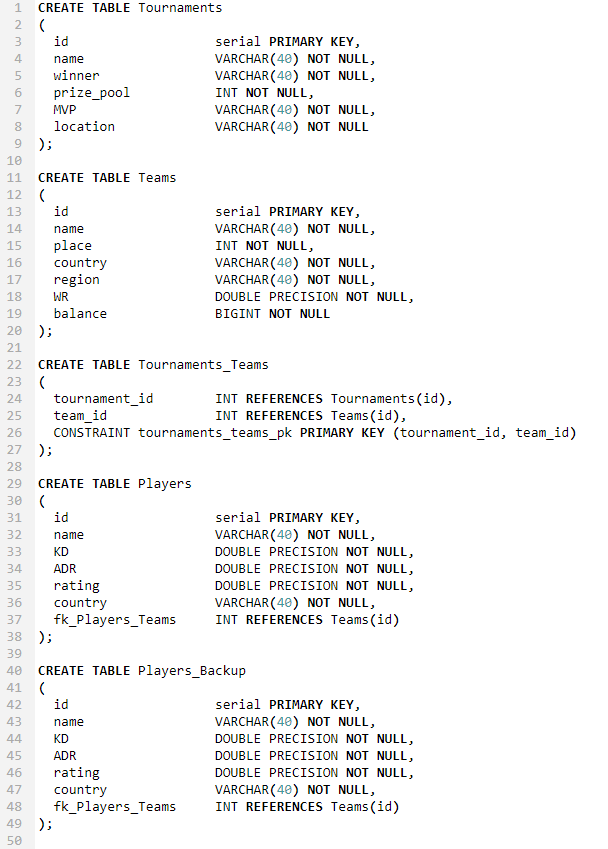
* Первичный ключ (serial)
* Имя тренера (char)
* Страна (char)
* Внешний ключ команды(integer)

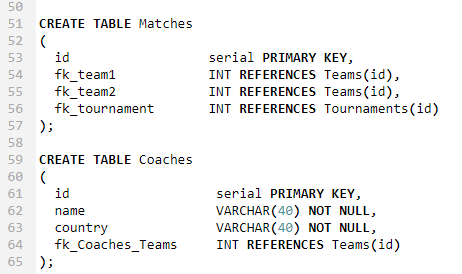
# **Разработка модели в ERWIN**

## Логическая модель:



## Физическая модель:





# **Запросы**

**JEFT JOIN (У игрока KennyS нет команды в данный момент)**

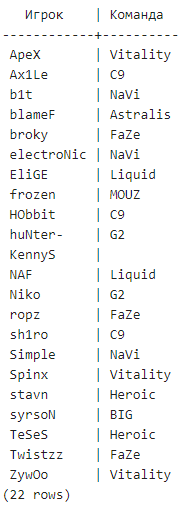
**1. Для кажого игрока вывести его нынешнюю команду**

SELECT Players.name AS "Игрок", Teams.name AS "Команда"

FROM Players LEFT JOIN Teams

ON Players.fk\_Players\_Teams = Teams.id

ORDER BY Players.name;



**RIGHT JOIN (В топе мирового рейтинга нет игроков из команды NIP)**

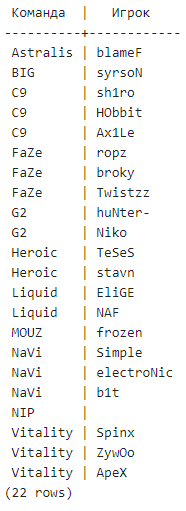
**2. Для кажой команды вывести ее игроков, которые находятся в топе мирового рейтинга**

SELECT Teams.name AS "Команда", Players.name AS "Игрок"

FROM Teams LEFT JOIN Players

ON Teams.id = Players.fk\_Players\_Teams

ORDER BY Teams.name;



**FULL JOIN**

**3. Все матчи указанного турнира**

SELECT Matches.id AS "Матч", a.name AS "Команда 1", b.name AS "Команда 2"

FROM Matches

FULL JOIN Teams AS a

ON Matches.fk\_team1 = a.id

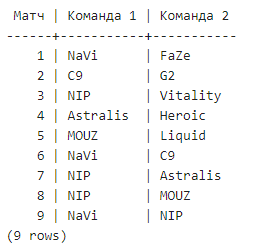
FULL JOIN Teams AS b

ON Matches.fk\_team2 = b.id

FULL JOIN Tournaments

ON Matches.fk\_tournament = Tournaments.id

WHERE Tournaments.name = 'iem-rio-major-2022';

****

**CROSS JOIN**

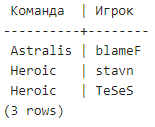
**4. Все игроки и тернеры представляющие Данию**

SELECT Teams.name AS "Команда", Players.name AS "Игрок"

FROM Teams

CROSS JOIN Players

WHERE Teams.country = 'Denmark' AND Players.fk\_Players\_Teams = Teams.id;

****

**CASE**

**5. Оценка интересности матча в зависимости от играющих команд**

SELECT Matches.id AS "Матч", a.name AS "Команда 1", b.name AS "Команда 2",

CASE WHEN a.id <= 3 AND b.id <= 3 THEN '5'

WHEN a.id <= 5 AND b.id <= 5 THEN '4'

WHEN a.id <= 7 AND b.id <= 7 THEN '3'

WHEN a.id <= 10 AND b.id <= 10 THEN '2'

ELSE '1'

END

FROM Matches

FULL JOIN Teams AS a

ON Matches.fk\_team1 = a.id

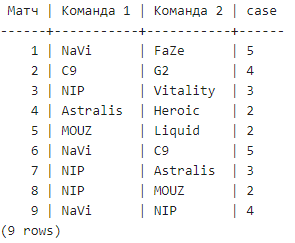
FULL JOIN Teams AS b

ON Matches.fk\_team2 = b.id

FULL JOIN Tournaments

ON Matches.fk\_tournament = Tournaments.id

WHERE Tournaments.name = 'iem-rio-major-2022';

****

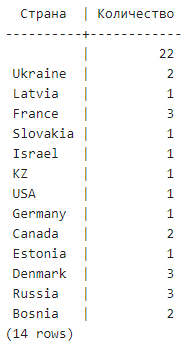
**ROLLUP**

**6. Подсчет количества игроков из каждой страны и их общего числа**

SELECT country AS "Страна", COUNT(\*) AS "Количество"

FROM Players

GROUP BY ROLLUP(country);

****

**GROUPING SETS**

**7. Подсчет количества игроков по отдельности и в командах**

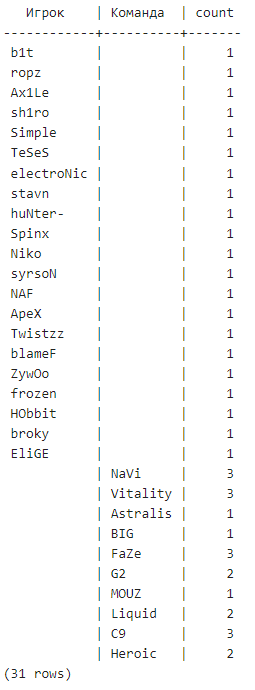
SELECT Players.name AS "Игрок", Teams.name AS "Команда", count(\*)

FROM Players

INNER JOIN Teams

ON Players.fk\_Players\_Teams = Teams.id

GROUP BY GROUPING SETS(Players.name, Teams.name);

****

**CUBE**

**8. Суммарный рейтинг команд, исходя из среднего рейтинга ее игроков**

SELECT Teams.name AS "Команда", Players.name AS "Игрок", avg(Players.rating) AS "Рейтинг"

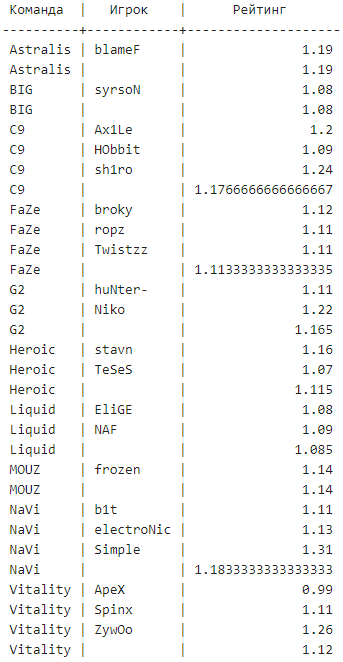
FROM Teams

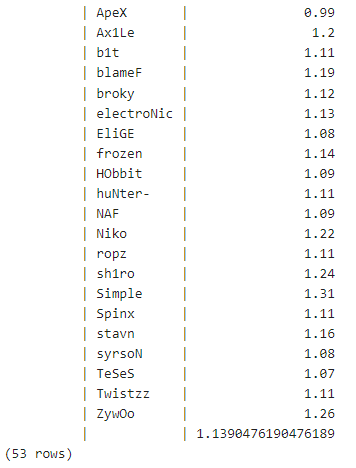
INNER JOIN Players

ON Players.fk\_Players\_Teams = Teams.id

GROUP BY CUBE(Teams.name, Players.name)

ORDER BY Teams.name, Players.name;

****

****

# **Разработка процедур на PL/SQL**

**Функция выводит всех игроков указанной команды**

CREATE OR REPLACE FUNCTION team\_composition(team\_name VARCHAR)

returns table (

name VARCHAR(40),

KD DOUBLE PRECISION,

ADR DOUBLE PRECISION,

rating DOUBLE PRECISION,

country VARCHAR(40),

team VARCHAR(40)

)

LANGUAGE PLPGSQL

AS $$

DECLARE

error\_text TEXT;

BEGIN

IF NOT EXISTS(SELECT FROM Teams WHERE Teams.name = team\_name) THEN

-- IF team\_name NOT IN(SELECT Teams.name FROM Teams) THEN

RAISE EXCEPTION 'Team name specified with an error: %', team\_name;

END IF;

RETURN QUERY

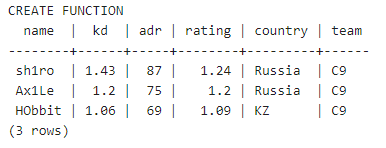
SELECT Players.name, Players.KD, Players.ADR, Players.rating, Players.country, Teams.name

FROM Players, Teams

WHERE Teams.id = Players.fk\_Players\_Teams AND Teams.name = team\_name;

END;

$$;

SELECT \* FROM team\_composition('C9');

**Процедура переводит призовой фонд турнира на баланс победившей команде**

SELECT Teams.name AS "Команда", Teams.balance AS "Баланс" FROM Teams WHERE Teams.name = 'Heroic';

SELECT Tournaments.name AS "Команда", Tournaments.prize\_pool AS "Призовой фонд", Tournaments.winner AS "Победитель"

FROM Tournaments

WHERE Tournaments.name = 'iem-rio-major-2022';

CREATE OR REPLACE PROCEDURE prizepool\_giveaway(tournament\_name VARCHAR)

LANGUAGE PLPGSQL

AS $$

DECLARE

old\_team\_balance BIGINT;

new\_team\_balance BIGINT;

team\_winner VARCHAR;

team\_prize BIGINT;

error\_text TEXT;

BEGIN

IF NOT EXISTS(SELECT FROM Tournaments WHERE Tournaments.name = tournament\_name) THEN

RAISE EXCEPTION 'Tournament name specified with an error: %', tournament\_name;

END IF;

SELECT winner INTO team\_winner FROM Tournaments WHERE Tournaments.name = tournament\_name;

SELECT prize\_pool INTO team\_prize FROM Tournaments WHERE Tournaments.name = tournament\_name;

SELECT balance INTO old\_team\_balance FROM Teams WHERE Teams.name = team\_winner;

UPDATE Teams SET balance = balance + team\_prize WHERE Teams.name = team\_winner;

SELECT balance INTO new\_team\_balance FROM Teams WHERE Teams.name = team\_winner;

IF (new\_team\_balance - old\_team\_balance <> team\_prize) THEN

RAISE EXCEPTION 'The transaction was wrong';

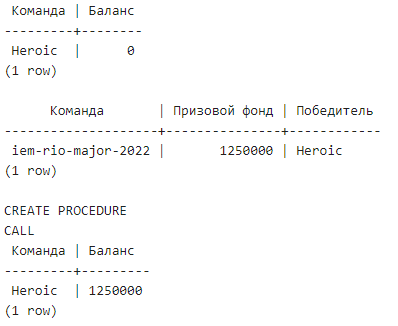
END IF;

END;

$$;

CALL prizepool\_giveaway('iem-rio-major-2022');

SELECT Teams.name AS "Команда", Teams.balance AS "Баланс" FROM Teams WHERE Teams.name = 'Heroic';



**Если рейтинг игрока изменяется, то копия данных об игроке (со старым рейтингом) добавляется в таблицу Players\_Backup**

CREATE OR REPLACE FUNCTION log\_rating\_changes()

RETURNS TRIGGER

LANGUAGE PLPGSQL

AS $$

BEGIN

IF NEW.rating <> OLD.rating THEN

INSERT INTO Players\_Backup(name, KD, ADR, rating, country, fk\_Players\_Teams)

VALUES(OLD.name, OLD.KD, OLD.ADR, OLD.rating, OLD.country, OLD.fk\_Players\_Teams);

END IF;

RETURN NEW;

END;

$$;

CREATE TRIGGER player\_rating\_changes

BEFORE UPDATE

ON Players

FOR EACH ROW

EXECUTE PROCEDURE log\_rating\_changes();

SELECT Players.name AS "Игрок", Players.rating AS "Рейтинг"

FROM Players WHERE Players.name = 'Simple';

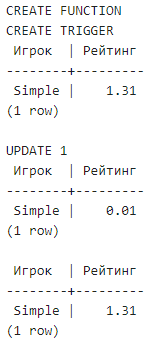
UPDATE Players SET rating = 0.01 WHERE name = 'Simple';

SELECT Players.name AS "Игрок", Players.rating AS "Рейтинг"

FROM Players WHERE Players.name = 'Simple';

SELECT Players\_Backup.name AS "Игрок", Players\_Backup.rating AS "Рейтинг"

FROM Players\_Backup WHERE Players\_Backup.name = 'Simple';



# **Список использованных источников**

1. Документация PostgreSQL https://www.postgresql.org/docs
2. PostgreSQL с примерами [https://www.postgresqltutorial.com/postgresql-plpgsql](https://vk.com/away.php?to=https%3A%2F%2Fwww.postgresqltutorial.com%2Fpostgresql-plpgsql&cc_key=)