Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет
имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ (ИУ)

КАФЕДРА ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ ЭВМ И ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ (ИУ7)

РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

К КУРСОВОЙ РАБОТЕ

НА ТЕМУ:

Разработка базы данных для проведения футбольных турниров

Студент группы **ИУ7-65Б** _____ **Ясинецкий Д. А.**

Руководитель курсовой работы _____ **Строганов Д. В.**

2023 г.

РЕФЕРАТ

Отчёт содержит 38 стр., 20 рис., 3 табл., 10 источн.

Курсовая работа представляет собой разработку базы данных для проведения футбольных турниров, а также приложения, предоставляющего интерфейс для доступа к базе данных.

В качестве системы управления базами данных используется PostgreSQL, которая подключается к приложению, реализованному на языке программирования C#.

Ключевые слова: база данных, PostgreSQL, C#, Windows Forms, система управления базами данных

ПЕРЕЧЕНЬ СОКРАЩЕНИЙ И ОБОЗНАЧЕНИЙ

В данной расчетно-пояснительной записке используются следующие сокращения и обозначения.

БД — база данных

СУБД — система управления базами данных

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

В настоящее время футбол является одним из самых популярных и массовых видов спорта во всем мире. Каждый год проводятся сотни футбольных турниров различного масштаба, включая международные соревнования, национальные лиги и местные чемпионаты [1]. Организация таких турниров требует учета множества факторов, начиная от регистрации команд до проведения матчей и отслеживания статистики.

Создание информационной системы позволит организаторам турниров эффективно управлять всеми аспектами проведения соревнований.

Целью курсового проекта является разработка базы данных для проведения футбольных турниров.

Задачи курсового проекта:

- провести анализ существующих баз данных на основе формализации данных;
- сформулировать описание пользователей проектируемого приложения для проведения футбольных турниров для доступа к базе данных;
- спроектировать сущности базы данных для проведения футбольных турниров;
- спроектировать ролевую модель на уровне базы данных для проведения футбольных турниров;
- выбрать средства реализации базы данных и приложения (в том числе выбор СУБД);
- разработать сущности базы данных для проведения футбольных турниров;
- провести исследование скорости рассчета турнирной таблицы от количества записей в базе данных через функцию базы данных и через функцию приложения.

1 Аналитический раздел

В данном разделе проведена формализация задачи и формализация данных. Был проведен обзор существующих решений, используемых для проведения футбольных турниров. Также были проанализированы различные модели баз данных, включая дореляционные, реляционные и постреляционные. Кроме того, были описаны типы пользователей, которые будут взаимодействовать с разработанной базой данных.

1.1 Формализация задачи

Необходимо спроектировать базу данных для хранения информации о пользователях, командах, странах, турнирах и матчах для проведения футбольных турниров. Требуется разработать Desktop-приложение, предоставляющее интерфейс для просмотра, добавления, изменения и удаления информации, хранящейся в базе данных. Необходимо реализовать три вида ролей — гость, авторизованный пользователь и администратор.

1.2 Анализ существующих решений

Среди решений, предоставляющих функционал для проведения футбольных турниров, были выделены 3 аналога:

– oball.ru [2];
– sportcup.org [3];
– tlab.pro [4].

Для сравнения были выбраны следующие критерии:

а) Возможность просмотра турниров других пользователей.
б) Возможность создания турнира с уже существующими командами.
в) Возможность просмотра информации о стране проведения турниров.

В таблице 1 представлено сравнение по вышеупомянутым критериям.

Таблица 1 – Существующие решения поставленной задачи

Критерий	**oball.ru**	**sportcup.org**	**tlab.pro**
1	+	+	+
2	+	-	-
3	-	-	-

Таким образом, среди представленных решений нет ни одного проекта, позволяющего просматривать информацию о стране проведения турниров. Кроме того, только в одном проекте есть возможность создания турнира с уже существующими командами. Разрабатываемое программное обеспечение будет предоставлять данный функционал.

1.3 Анализ моделей баз данных

База данных — это самодокументированное собрание интегрированных записей. Модель базы данных определяет логическую структуру базы данных. Существует три основных типа модели баз данных:

а) Дореляционные.

б) Реляционные.

в) Постреляционные.

К дореляционным моделям данных относятся иерархические и сетевые модели баз данных.

Информация в иерархической модели организована по принципу древовидной структуры, в виде отношений «предок-потомок» [5]. Каждая запись может иметь не более одной родительской записи и несколько подчиненных. Связи записей реализуются в виде физических указателей с одной записи на другую. Основной недостаток иерархической модели — невозможность реализовать отношения «многие-ко-многим», а также ситуации, когда запись имеет несколько предков.

Сетевая модель отличается от иерархической тем, что у потомка может быть любое количество предков.

Реляционная модель предназначена для хранения и организации точек данных с заданными отношениями для быстрого доступа. Данные в реляционных базах данных упорядочиваются в виде таблиц, которые содержат информацию о каждой сущности и представляют заданные заранее категории с помощью строк и столбцов [6]. Такое структурирование данных повышает эффективность и гибкость доступа. Кроме того, реляционная модель наиболее эффективно поддерживает целостность данных во всех приложениях и копиях (экземплярах) базы данных.

Постреляционная модель базы данных представляет собой расширенную реляционную модель, снимающую ограничение неделимости данных, хранящихся в записях таблиц. Постреляционная модель допускает наличие множественных значений в полях данных. Значения таких полей могут быть разделены на несколько подзначений, которые считаются отдельными записями в таблице. Главным недостатком этой модели является сложность решения проблемы обеспечения целостности и непротиворечивости хранимых данных.

В качестве модели базы данных была выбрана реляционная модель, поскольку она обеспечивает целостность данных, эффективность и гибкость доступа к данным, а также исключает дублирование данных.

1.4 Формализация данных

Разрабатываемая база данных должна содержать информацию о пользователях, командах, странах, турнирах и матчах. В таблице 2 представлены категории данных в БД и информация о них.

Таблица 2 – Категории данных в БД и информация о них

Категория	**Информация**
Пользователь	Id пользователя, логин, hash пароля, права доступа
Команда	Id команды, название, Id страны базирования
Страна	Id страны, название, конфедерация
Матч	Id матча, Id турнира, Id домашней команды, Id гостевой команды, голы домашней команды, голы гостевой команды
Турнир	Id турнира, название, Id пользователя (создателя), Id страны проведения
Связь команды и турнира	Id команды, Id турнира

На рисунке 1 представлена ER-диаграмма системы в нотации Чена.

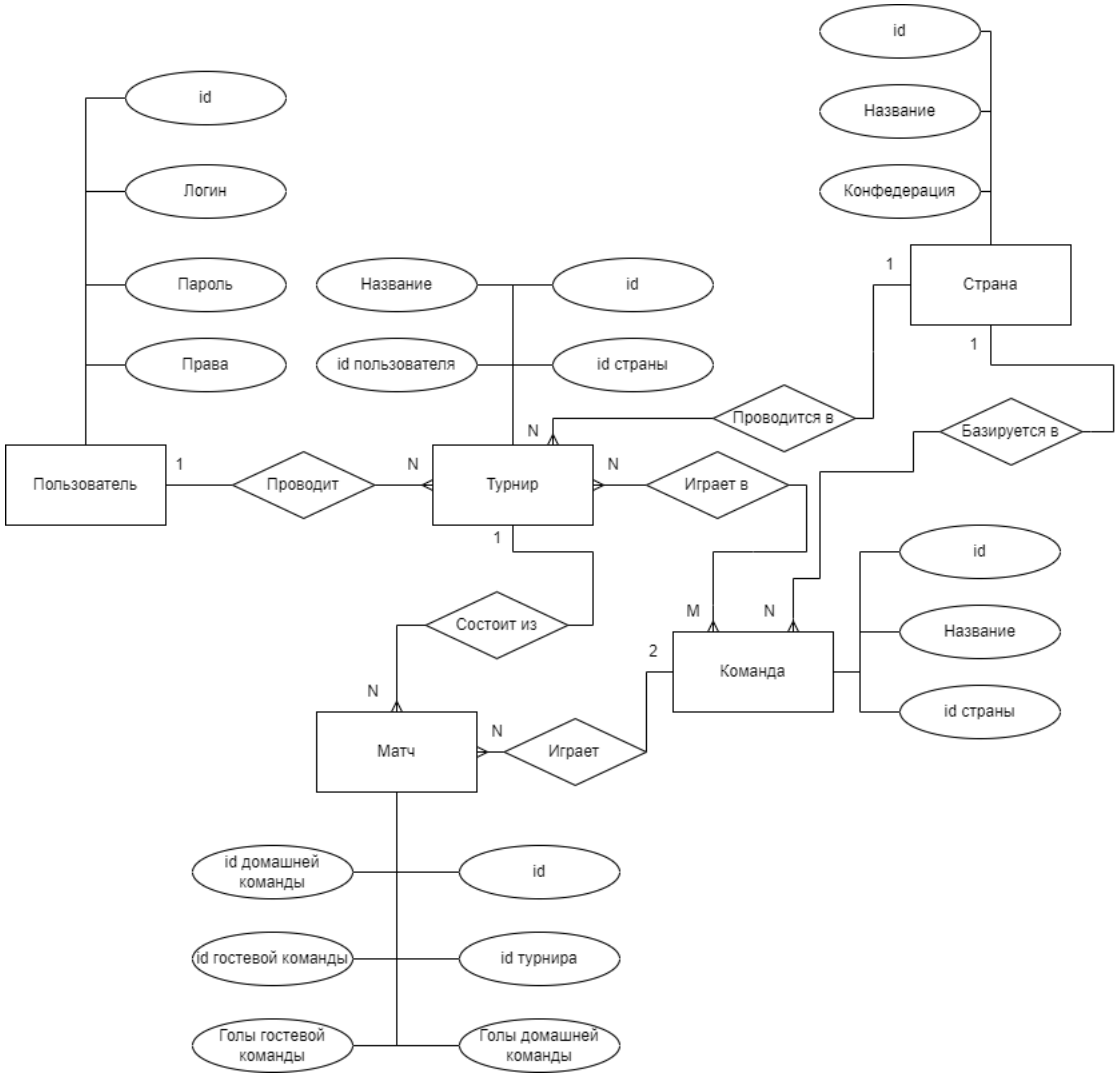


Рисунок 1 – ER-диаграмма системы в нотации Чена

1.5 Описание пользователей

Было выделено три категории пользователей — гость, авторизованный пользователь и администратор.

Гость может просматривать информацию о пользователях, командах, странах, турнирах (в том числе турнирную таблицу и результаты матчей), а также зарегистрироваться или авторизоваться.

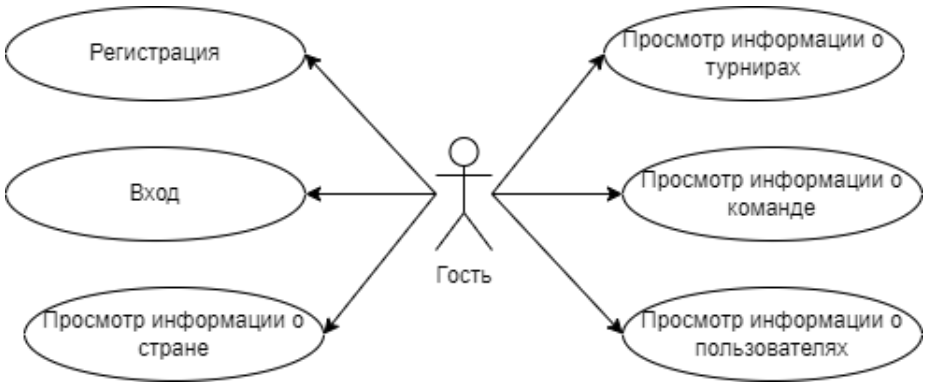


Рисунок 2 – Use-case диаграмма для гостя

Авторизованный пользователь, кроме просмотра информации, может создавать команды и турниры, а также редактировать информацию о матчах своих турниров.

Рисунок 3 – Use-case диаграмма для авторизованного пользователя

В качестве дополнительного функционала администратор может добавлять страны и изменять права пользователей.

Рисунок 4 – Use-case диаграмма для администратора

Вывод

В данном разделе была проведена формализация задачи, формализация данных, анализ существующих решений и анализ моделей баз данных и описаны типы пользователей. В качестве модели базы данных была выбрана реляционная модель.

2 Конструкторский раздел

В данном разделе спроектирована базы данных для проведения футбольных турниров. Была разработана схема базы данных, которая включает в себя таблицы, связи между ними и атрибуты. Каждая таблица в базе данных была описана подробно, описывая ее структуру, включая названия полей и их типы данных. Также были представлены алгоритм расчёта турнирной таблицы и ролевая модель.

2.1 Проектирование базы данных

На рисунке 5 представлена схема разрабатываемой базы данных.

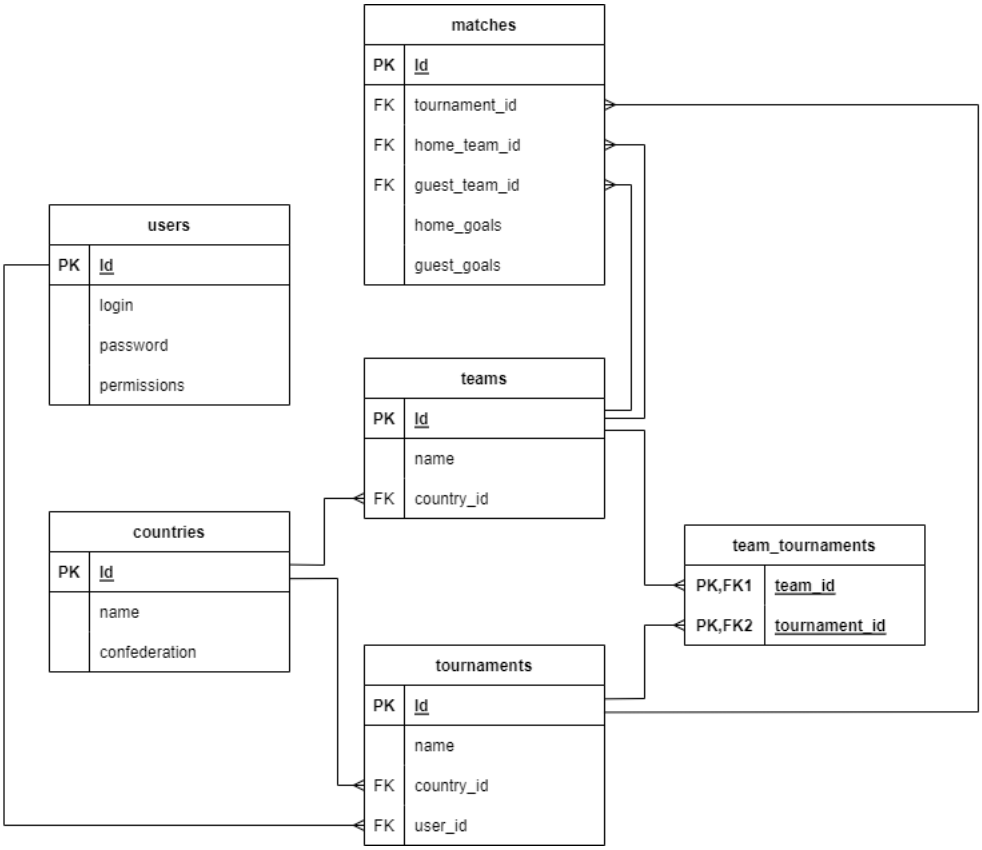


Рисунок 5 – Схема разрабатываемой базы данных

Таблица users хранит информацию о пользователях и содержит следующие поля:

– id — идентификатор пользователя, первичный ключ;
– login — логин пользользователя, символьный тип;
– password — hash пароля, символьный тип;
– permissions — права пользователя, символьный тип.

Таблица teams хранит информацию о командах и содержит следующие поля:

- id — идентификатор команды, первичный ключ;
- name — название команды, символьный тип;
- country_id — идентификатор страны базирования команды, внешний ключ на поле id таблицы countries.

Таблица countries хранит информацию о странах и содержит следующие поля:

- id — идентификатор страны, первичный ключ;
- name — название страны, символьный тип;
- confederation — конфедерация страны, символьный тип.

Таблица tournaments хранит информацию о турнирах и содержит следующие поля:

- id — идентификатор турнира, первичный ключ;
- name — название турнира, символьный тип;
- country_id — идентификатор страны проведения турнира, внешний ключ на поле id таблицы countries;
- user_id — идентификатор создателя турнира, внешний ключ на поле id таблицы users.

Таблица matches хранит информацию о матчах и содержит следующие поля:

- id — идентификатор матча, первичный ключ;
- tournament_id — идентификатор турнира, внешний ключ на поле id таблицы tournaments;
- home_team_id — идентификатор домашней команды, внешний ключ на поле id таблицы teams.
- guest_team_id — идентификатор гостевой команды, внешний ключ на поле id таблицы teams.
- home_goals — количество голов домашней команды, целое число;
- guest_goals — количество голов гостевой команды, целое число.

Таблица team_tournaments хранит информацию о связях команда-турнир и содержит следующие поля:

- team_id — идентификатор команды, внешний ключ на поле id таблицы teams, часть первичного ключа;
- tournament_id — идентификатор турнира, внешний ключ на поле id таблицы tournament, часть первичного ключа.

2.2 Алгоритм расчёта турнирной таблицы

При создании базы данных была определена функция get_table для расчёта турнирной таблицы. Возвращаемая таблица содержит количество матчей, побед, ничьих, поражений, забитых голов, пропущенных голов и очков для каждой команды турнира. Кроме того, таблица сортируется по убыванию по количеству очков. При равенстве очков более высокое место займёт команда с большей разницей голов. В случае равного количества очков и разницы голов выше окажется команда с большим количеством побед.

На рисунке 6 представлена схема алгоритма расчёта турнирной таблицы.

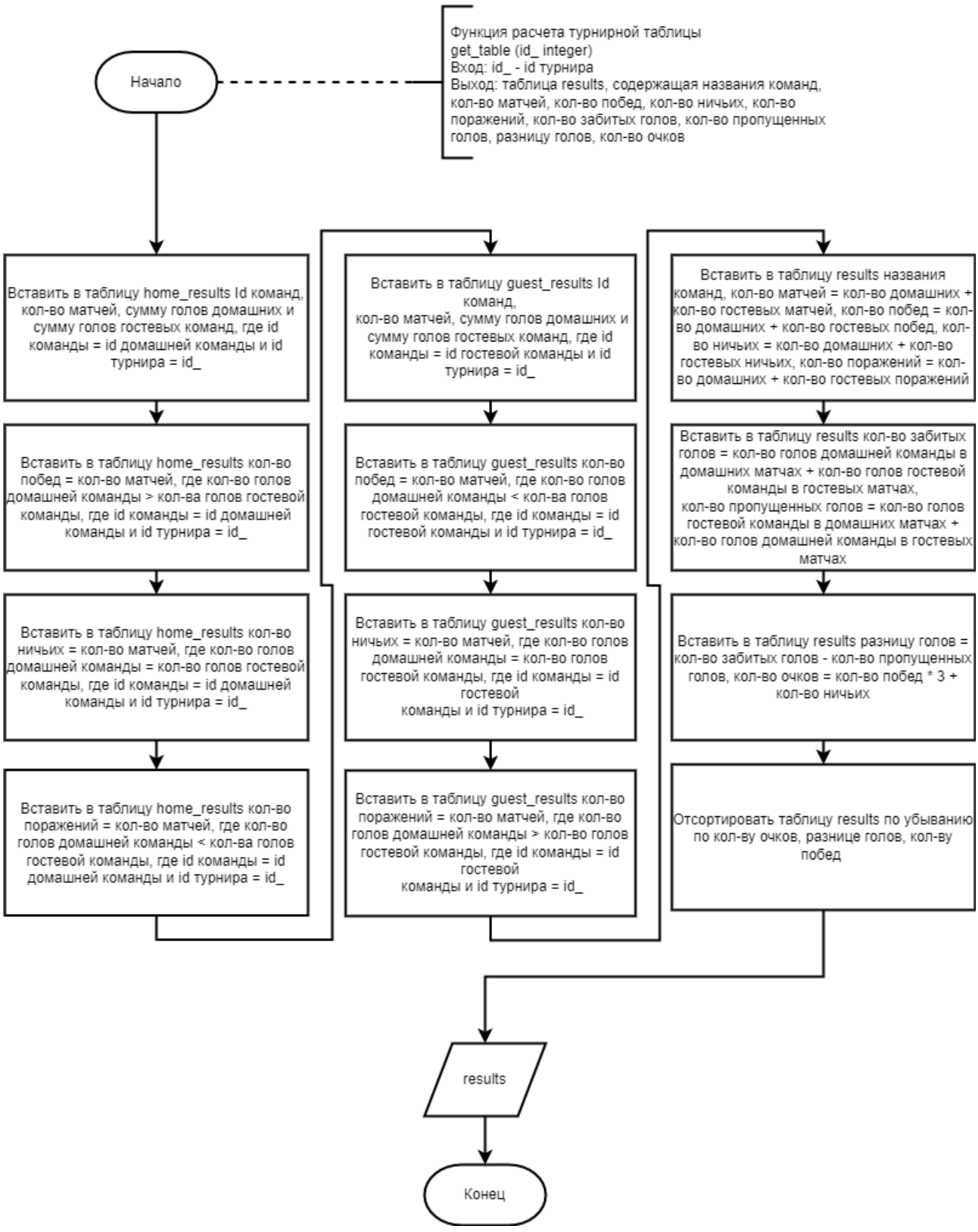


Рисунок 6 – Схема алгоритма расчёта турнирной таблицы

2.3 Создание ролевой модели

Для каждой категории пользователей, описанных ранее, была создана отдельная роль в базе данных.

а) Гость имеет права на просмотр всех таблиц. Кроме того, гость имеет права на добавление записей в таблице users для возможности регистрации и авторизации. Другими правами гость не наделён.

б) Авторизованный пользователь также имеет права на просмотр всех таблиц. У него есть возможность создания команд, турниров и матчей, поэтому авторизованный пользователь наделён правами на добавление записей в таблицы teams, tournaments, team_tournaments и matches. Кроме того, он может удалять турниры, а также изменять и удалять матчи, поэтому имеет права на удаление записей из таблиц tournaments, matches и team_tournaments и права на обновление записей в таблице matches.

в) Администратор имеет права на просмотр, удаление, добавление и обновление записей во всех таблицах.

Вывод

В данном разделе была спроектирована база данных, в результате чего была приведена схема разрабатываемой базы данных и описаны поля всех таблиц. Также был приведён алгоритм расчёта турнирной таблицы и описаны права доступа для каждой категории пользователей.

3 Технологический раздел

В данном разделе выбраны средства реализации, в том числе определенный язык программирования и фреймворк, учитывая их поддержку работы с базами данных. Также была выбрана конкретная СУБД, представлены реализации создания баз данных, функции расчёта турнирной таблицы и ролевой модели. Кроме того, был представлен интерфейс приложения, предоставляющий возможность взаимодействия с базой данных, включая просмотр и редактирование данных.

3.1 Средства реализации

Для написания приложения был выбран язык программирования C#, поскольку:

- C# объектно-ориентирован, что позволяет создавать классы, объекты и методы, упрощая моделирование объектов базы данных и их взаимосвязей;
- C# имеет встроенный механизм LINQ [7], который предоставляет объектно-ориентированный синтаксис для упрощения запросов к базе данных;
- C# является частью .NET framework [8], который предоставляет широкий набор библиотек и инструментов для разработки приложений, работающими с базами данных.

В качестве среды разработки была выбрана Visual Studio, так как:

- данная среда разработки является бесплатной;
- предоставляет удобный интерфейс для работы с Windows Forms [9] — инструментом для создания графического интерфейса.

3.2 Выбор СУБД

В качестве СУБД была выбрана PostgreSQL. Данный выбор обусловлен следующими причинами:

- PostgreSQL — СУБД с открытым исходным кодом, что означает, что она свободно доступна для использования и модификации;
- данная СУБД обеспечивает детальный контроль доступа, позволяя определять роли пользователей, разрешения и ограничения доступа на различных уровнях;

– PostgreSQL позволяет расширять его функциональность с помощью пользовательских расширений и определяемых пользователем функций.

3.3 Реализация создания таблиц базы данных

В листинге 1 представлено создание таблиц users, countries и tournaments.

Листинг 1 – Создание таблиц users, countries, tournaments

```
create table if not exists users
(
    id int not null primary key,
    login varchar(64) not null,
    password varchar(64) not null,
    permissions varchar(64) not null
);

create table if not exists countries
(
    id int not null primary key,
    name varchar(64) not null,
    confederation varchar(64) not null
);

create table if not exists tournaments
(
    id int not null primary key,
    name varchar(64) not null,
    country_id int,
    foreign key (country_id) references countries(id) on delete set
       null,
    user_id int not null,
    foreign key (user_id) references users(id) on delete cascade
);
```

В листинге 2 представлено создание таблиц teams, matches и team_tournaments.

Листинг 2 – Создание таблиц teams, matches, team_tournaments

```
create table if not exists teams
(
    id int not null primary key,
    name varchar(64) not null,
    country_id int,
    foreign key (country_id) references countries(id) on delete set
       null
);

create table if not exists matches
(
    id int not null primary key,
    tournament_id int not null,
    foreign key (tournament_id) references tournaments(id) on delete
       cascade,
    home_team_id int not null,
    foreign key (home_team_id) references teams(id) on delete cascade
       ,
    guest_team_id int not null,
    foreign key (guest_team_id) references teams(id) on delete
       cascade,
    home_goals int not null,
    guest_goals int not null
);

create table if not exists team_tournaments
(
    team_id int not null,
    foreign key (team_id) references teams(id) on delete cascade,
    tournament_id int not null,
    foreign key (tournament_id) references tournaments(id) on delete
       cascade,
    primary key(team_id, tournament_id)
);
```

3.4 Реализация функции расчёта турнирной таблицы

В листингах 3 и 4 представлена функция расчёта турнирной таблицы.

Листинг 3 – Функция расчёта турнирной таблицы (часть 1)

```
create or replace function get_table(id_ integer)
returns table(name varchar(64), matches integer, wins integer, draws
   integer, loses integer, gs integer, gc integer, points integer)
as $$
begin
  drop table if exists tour_clubs;
  create temp table if not exists tour_clubs(id integer, name varchar
     (64));
    insert into tour_clubs(id, name)
  select c.id, c.name
  from teams c join team_tournaments tt on c.id = tt.team_id
    join tournaments t on tt.tournament_id = t.id
  where t.id = id_;

  drop table if exists home_results;
  create temp table if not exists home_results(id integer, matches
     integer, wins integer, draws integer, loses integer, gs integer,
      gc integer);
  insert into home_results(id, matches, wins, draws, loses, gs, gc)
  select c.id, count(*), sum(case when m.home_goals > m.guest_goals
     then 1 else 0 end),
    sum(case when m.home_goals = m.guest_goals then 1 else 0 end),
    sum(case when m.home_goals < m.guest_goals then 1 else 0 end),
    sum(m.home_goals), sum(m.guest_goals)

  from tour_clubs c join matches m on c.id = m.home_team_id
  where m.tournament_id = id_
  group by c.id;

  drop table if exists guest_results;
    create temp table if not exists guest_results(id integer, matches
        integer, wins integer, draws integer, loses integer, gs
       integer, gc integer);
  insert into guest_results(id, matches, wins, draws, loses, gs, gc)
  select c.id, count(*),
    sum(case when m.home_goals < m.guest_goals then 1 else 0 end),
    sum(case when m.home_goals = m.guest_goals then 1 else 0 end),
    sum(case when m.home_goals > m.guest_goals then 1 else 0 end),
    sum(m.guest_goals), sum(m.home_goals)
```

Листинг 4 – Функция расчёта турнирной таблицы (часть 2)

```
  from tour_clubs c join matches m on c.id = m.guest_team_id
  where m.tournament_id = id_
  group by c.id;

  drop table if exists results;
    create temp table if not exists results(name varchar(64), matches
        integer, wins integer, draws integer, loses integer, gs
       integer, gc integer, diff integer, points integer);
  insert into results(name, matches, wins, draws, loses, gs, gc, diff
     , points)
  select c.name, h.matches + g.matches, h.wins + g.wins, h.draws + g.
     draws, h.loses + g.loses,
    h.gs + g.gs, h.gc + g.gc,
    h.gs + g.gs - h.gc - g.gc,
    (h.wins + g.wins) * 3 + h.draws + g.draws
  from tour_clubs c join home_results h on c.id = h.id
    join guest_results g on c.id = g.id;
  return query
  select r.name, r.matches, r.wins, r.draws, r.loses, r.gs, r.gc, r.
     points
  from results r
  order by r.points desc, r.diff desc, r.wins desc;

end
$$ language PLPGSQL
```

3.5 Реализация ролевой модели

В листингах 5 и 6 представлены создание ролей и наделение их правами доступа.

Листинг 5 – Создание ролей и наделение их правами доступа (часть 1)

```
CREATE ROLE admin WITH
    CONNECTION LIMIT -1
    LOGIN
    PASSWORD 'admin';

GRANT ALL PRIVILEGES
    ON ALL TABLES IN SCHEMA public
    TO admin;
```

Листинг 6 – Создание ролей и наделение их правами доступа (часть 2)

```
CREATE ROLE _user WITH
    CONNECTION LIMIT -1
    LOGIN
    PASSWORD 'user';

GRANT SELECT
    ON ALL TABLES IN SCHEMA public
    TO _user;

GRANT INSERT
    ON public."teams",
       public."tournaments",
       public."team_tournaments",
       public."matches"
    TO _user;

GRANT DELETE
    ON public."tournaments",
       public."team_tournaments",
       public."matches"
    TO _user;

GRANT UPDATE
    ON public."matches"
    TO _user;

CREATE ROLE guest WITH
    CONNECTION LIMIT -1
    LOGIN
    PASSWORD 'guest';

GRANT SELECT
    ON ALL TABLES IN SCHEMA public
    TO guest;

GRANT INSERT
    ON public."users"
    TO guest;
```

3.6 Интерфейс приложения

При запуске приложение открывается меню, продемонстрированное на рисунке 7. При этом изначально пользователь является неавторизованным (гостем).

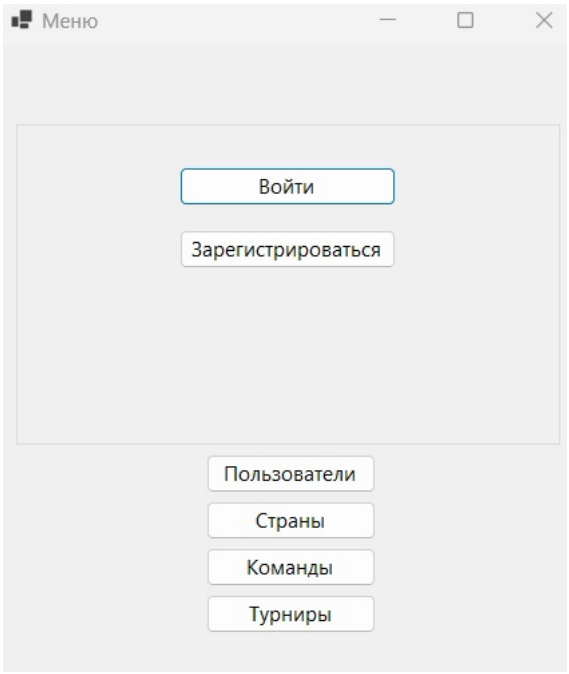


Рисунок 7 – Меню приложения

На рисунках 8 и 9 представлены окно входа в аккаунт и окно регистрации соответственно.

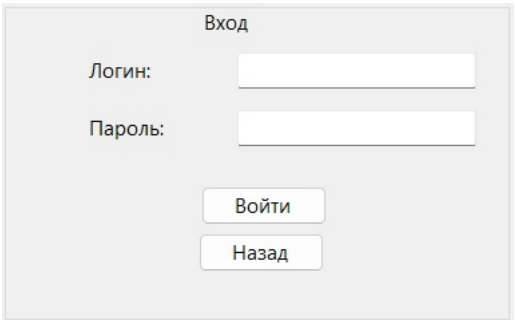


Рисунок 8 – Окно входа

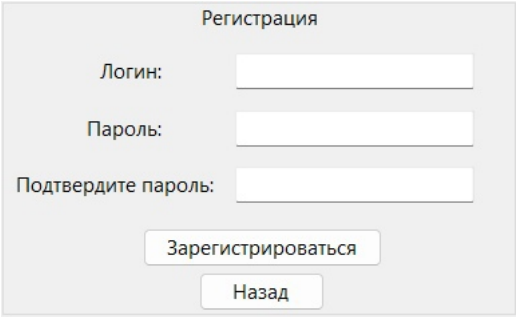


Рисунок 9 – Окно регистрации

Далее окна будут показаны в том виде, в котором они открываются для администратора, для отображения всего функционала приложения.

На рисунке 10 представлено окно просмотра пользователей.

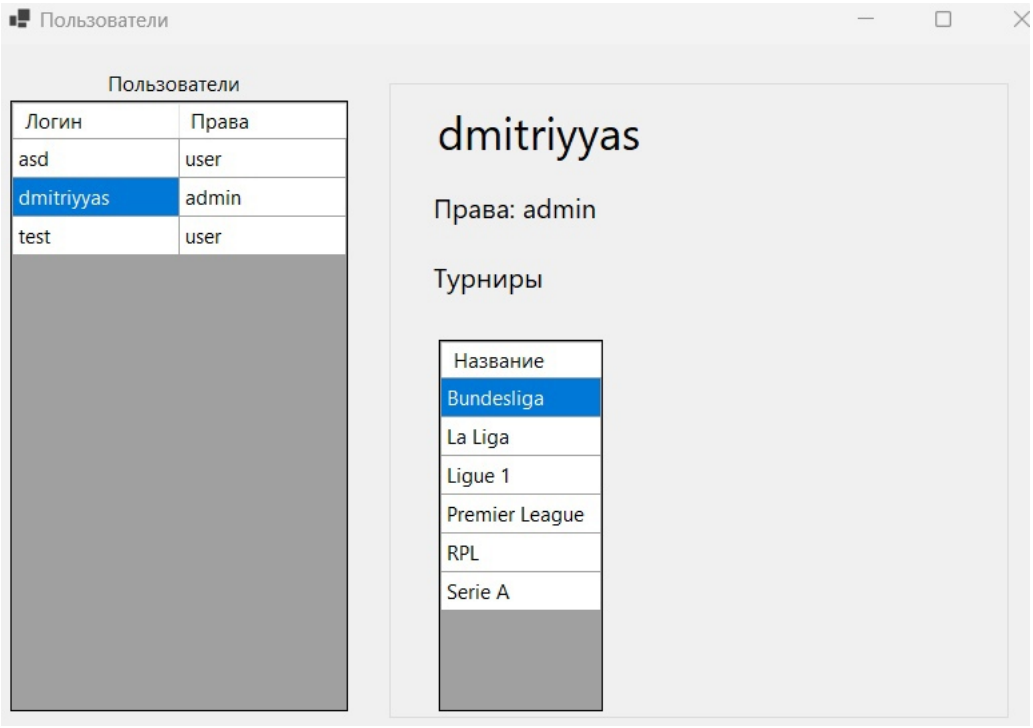


Рисунок 10 – Окно просмотра пользователей

На рисунке 11 представлено окно просмотра стран. В случае, если пользователь не обладает правами администратора, кнопка «Создать страну» не будет активна.

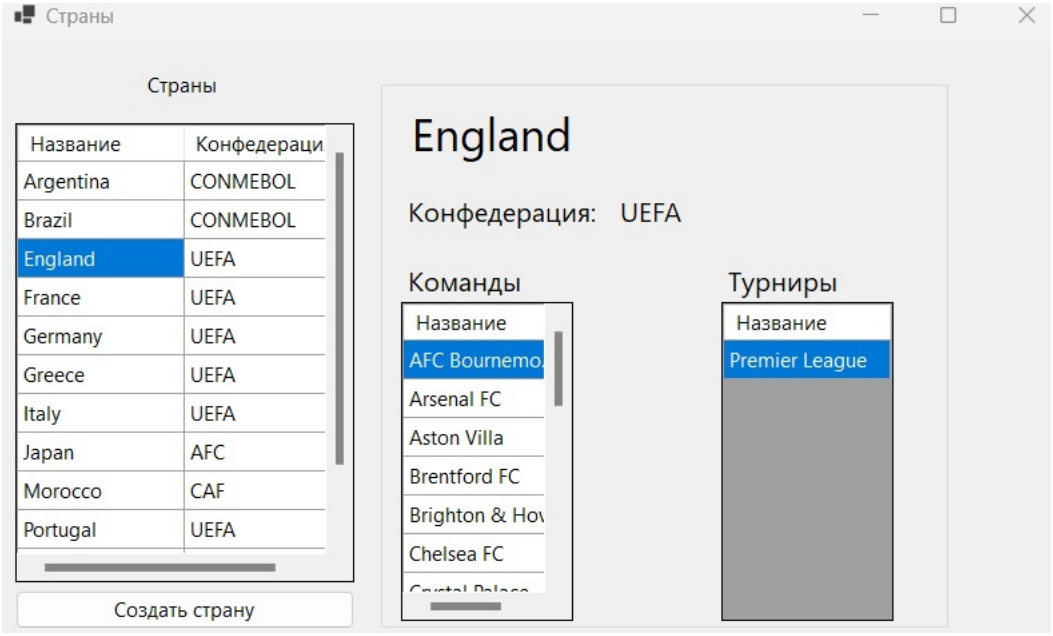


Рисунок 11 – Окно просмотра стран

На рисунке 12 представлено окно создания страны.

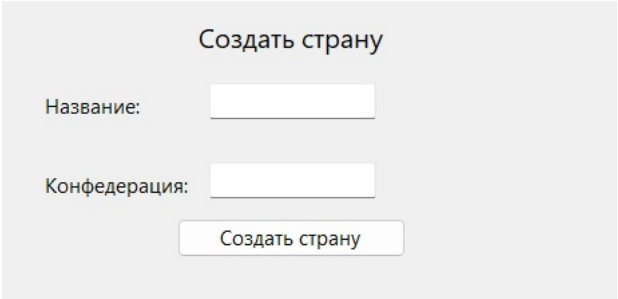


Рисунок 12 – Окно создания страны

На рисунке 13 представлено окно просмотра команд. В случае, если пользователь не авторизован, кнопка «Создать команду» не будет активна.

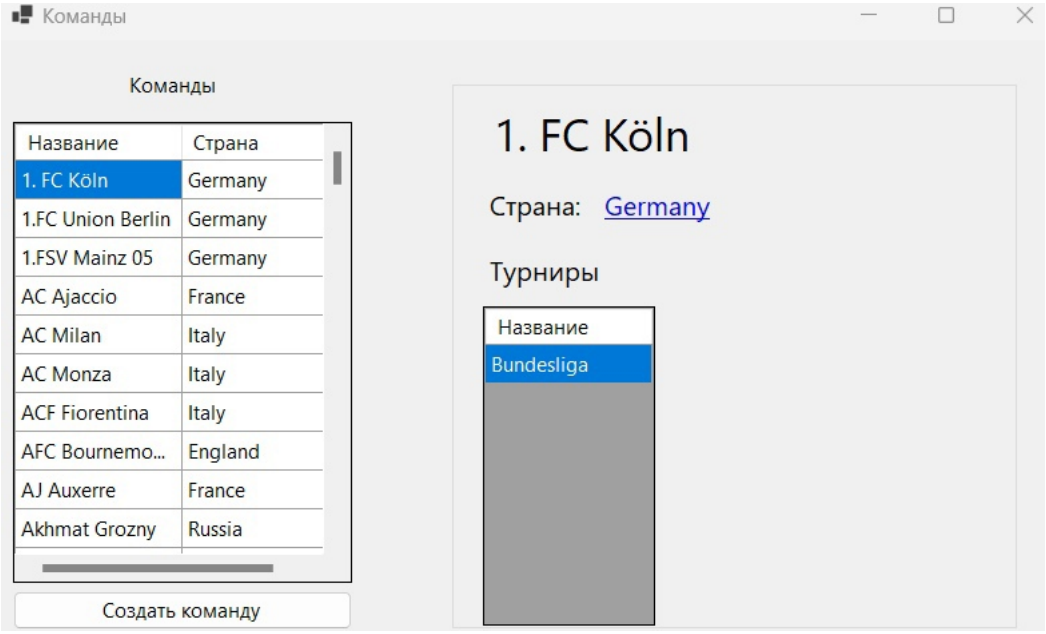


Рисунок 13 – Окно просмотра команд

На рисунке 14 представлено окно создания команды.

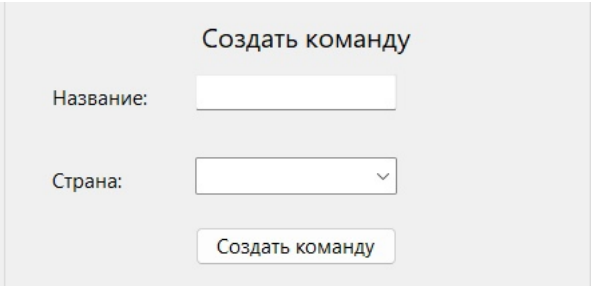


Рисунок 14 – Окно создания команды

На рисунке 15 представлено окно просмотра турниров. В случае, если пользователь не авторизован, кнопка «Создать турнир» не будет активна, а если пользователь просматривает чужой турнир, то кнопка «Удалить турнир» не будет активна.

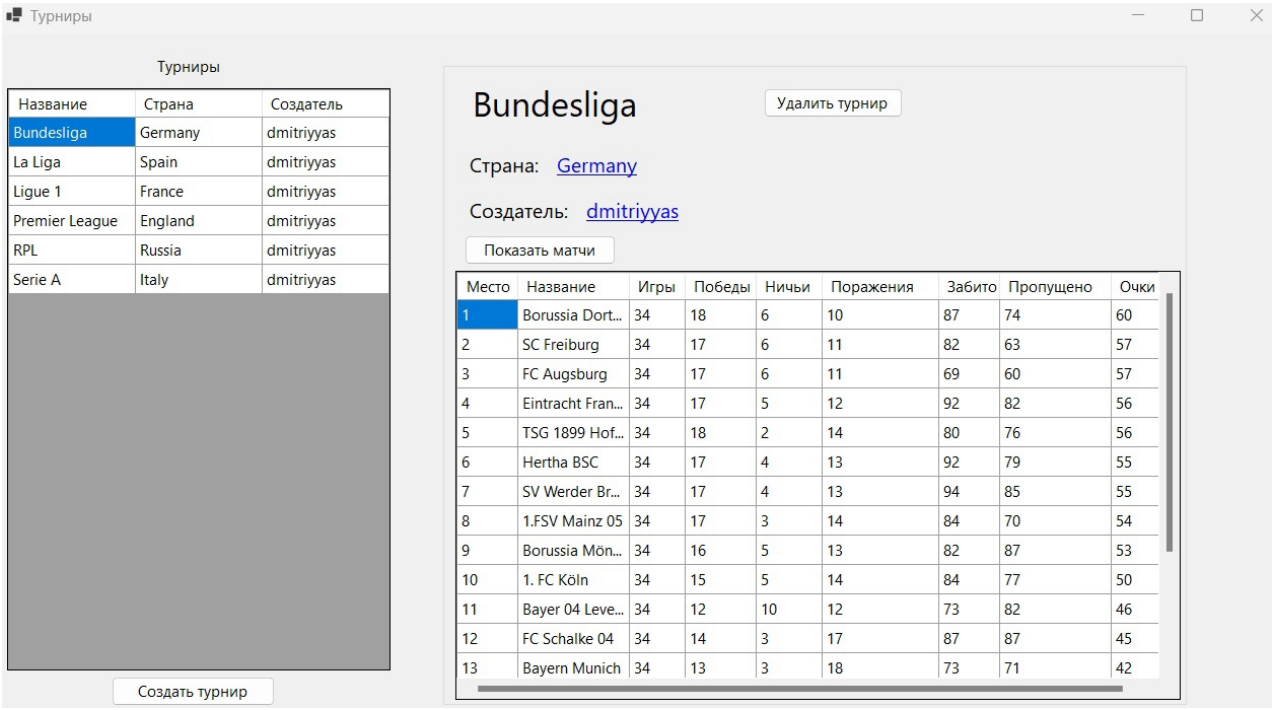


Рисунок 15 – Окно просмотра турниров

На рисунке 16 представлено окно создания турнира.

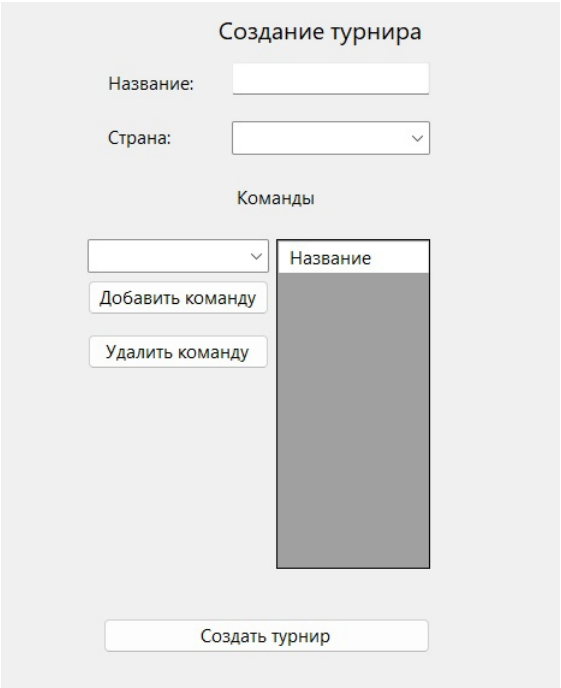
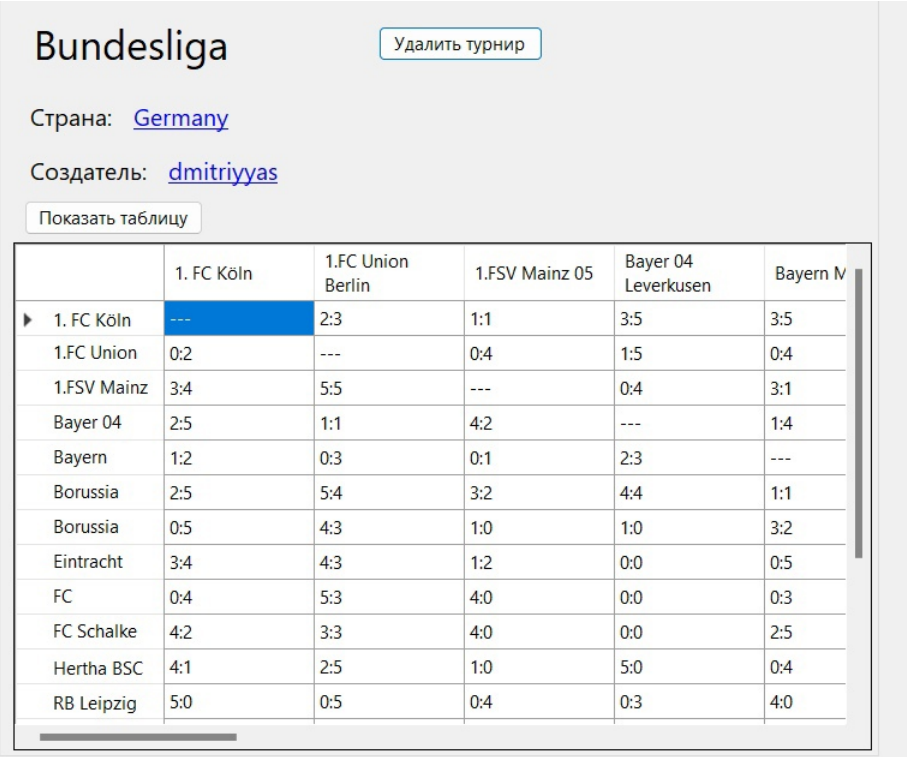


Рисунок 16 – Окно создания турнира

На рисунке 17 представлено окно просмотра турниров после нажатия на кнопку «Показать матчи».



	1. FC Köln	1.FC Union Berlin	1.FSV Mainz 05	Bayer 04 Leverkusen	Bayern M
1. FC Köln	---	2:3	1:1	3:5	3:5
1.FC Union	0:2	---	0:4	1:5	0:4
1.FSV Mainz	3:4	5:5	---	0:4	3:1
Bayer 04	2:5	1:1	4:2	---	1:4
Bayern	1:2	0:3	0:1	2:3	---
Borussia	2:5	5:4	3:2	4:4	1:1
Borussia	0:5	4:3	1:0	1:0	3:2
Eintracht	3:4	4:3	1:2	0:0	0:5
FC	0:4	5:3	4:0	0:0	0:3
FC Schalke	4:2	3:3	4:0	0:0	2:5
Hertha BSC	4:1	2:5	1:0	5:0	0:4
RB Leipzig	5:0	0:5	0:4	0:3	4:0

Рисунок 17 – Окно просмотра матчей турнира

На рисунке 18 представлено окно просмотра матча. В случае, если пользователь просматривает матч чужого турнира, кнопки «Сохранить матч» и «Удалить матч» не будут активны.

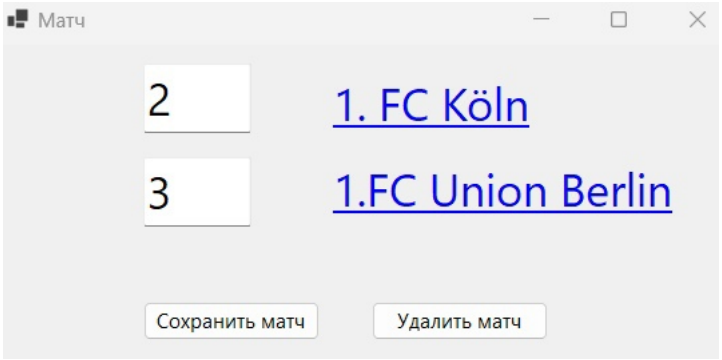


Рисунок 18 – Окно просмотра матча

На рисунке 19 представлено окно создания матча. В случае, если пользователь просматривает матч чужого турнира, кнопка «Создать матч» не будет активна.

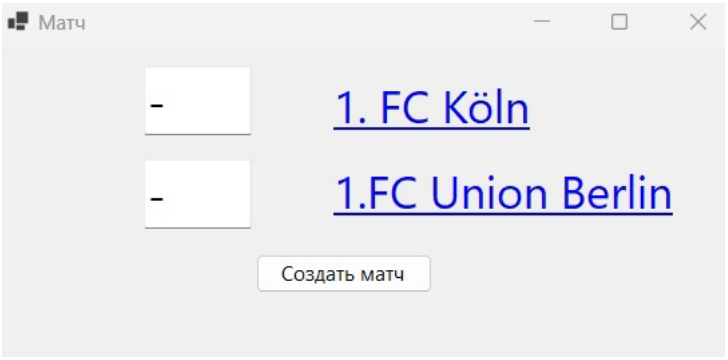


Рисунок 19 – Окно создания матча

Вывод

В данном разделе были выбраны средства реализации — язык программирования C# и среда разработки Visual Studio. В качестве СУБД была выбрана PostgreSQL. Также были приведены средства реализации и интерфейс пользователя.

4 Исследовательский раздел

В данном разделе была представлена постановка эксперимента, направленного на сравнение времени расчета турнирной таблицы в зависимости от двух факторов: количества команд, участвующих в турнире, и места проведения расчета, то есть на стороне базы данных или на стороне приложения. Для проведения эксперимента были подготовлены тестовые данные, включающая турниры с разным количеством команд и результаты матчей.

4.1 Описание эксперимента

В эксперименте сравнивается время расчёта турнирной таблицы в зависимости от количества команд в турнире и от используемой функции. Для каждого количества команд и каждой функции замеры производятся 10 раз, а затем вычисляется среднее арифметическое.

В листингах 7 — 9 приведена реализация функции расчёта турнирной таблицы на стороне приложения.

Листинг 7 – Функция расчёта таблицы на стороне приложения (часть 1)

```
public IEnumerable<TeamStatistics> getTournamentTable(int
   tournamentId)
{
    List<TeamStatistics> table = new List<TeamStatistics>();
    IEnumerable<Team> teams = getTournamentTeams(tournamentId);
    foreach(var team in teams)
    {
        TeamStatistics statistics = new TeamStatistics(team.Name);
        IEnumerable<Match> matches = _matchRepository.getByHostTeam(
           team.Id, tournamentId);
        foreach(var match in matches)
        {
            statistics.Matches += 1;
            statistics.GoalsScored += match.HomeGoals;
            statistics.GoalsConceded += match.GuestGoals;
            if (match.HomeGoals > match.GuestGoals)
            {
                statistics.Wins += 1;
                statistics.Points += 3;
            }
            else if (match.HomeGoals == match.GuestGoals)
```

Листинг 8 – Функция расчёта таблицы на стороне приложения (часть 2)

```
            {
                statistics.Draws += 1;
                statistics.Points += 1;
            }
            else
            {
                statistics.Loses += 1;
            }
        }
        matches = _matchRepository.getByGuestTeam(team.Id,
           tournamentId);
        foreach (var match in matches)
        {
            statistics.Matches += 1;
            statistics.GoalsScored += match.GuestGoals;
            statistics.GoalsConceded += match.HomeGoals;
            if (match.HomeGoals < match.GuestGoals)
            {
                statistics.Wins += 1;
                statistics.Points += 3;
            }
            else if (match.HomeGoals == match.GuestGoals)
            {
                statistics.Draws += 1;
                statistics.Points += 1;
            }
            else
            {
                statistics.Loses += 1;
            }
        }

        table.Add(statistics);
    }
    table.Sort((y, x) =>
    {
        if (x.Points != y.Points)
        {
            return x.Points.CompareTo(y.Points);
        }
```

Листинг 9 – Функция расчёта таблицы на стороне приложения (часть 3)

```
        int xDiff = x.GoalsScored - x.GoalsConceded;
        int yDiff = y.GoalsScored - y.GoalsConceded;
        if (xDiff != yDiff)
        {
            return xDiff.CompareTo(yDiff);
        }

        if (x.Wins!= y.Wins)
        {
            return x.Wins.CompareTo(y.Wins);
        }

        return 0;
    });
    return table;
}
```

Реализация функции расчёта турнирной таблицы на стороне базы данных приведена в пункте 3.4.

4.2 Результаты эксперимента

Ниже приведены технические характеристики устройства, на котором выполнялось тестирование.

– Операционная система: Windows 11 Home.
– Оперативная память: 8 Гб.
– Процессор: 11th Gen Intel(R) Core(TM) i5-1135G7 @ 2.40ГГц [10].

Во время тестирования компьютер был нагружен только встроенными приложениями окружения, а также непосредственно системой тестирования.

В таблице 3 приведены результаты эксперимента. Время в таблице измеряется в миллисекундах.

Таблица 3 – Результаты сравнения времени расчёта турнирной таблицы в зависимости от количества команд и используемой функции

Количество команд	**Функция на стороне БД**	**Функция на стороне приложения**
10	11	101
20	12	110
30	12	222
40	14	320
50	16	475
60	17	547
70	19	632
80	21	746
90	23	850
100	25	958

На рисунке 20 приведена зависимость времени расчёта турнирной таблицы в зависимости от количества команд и используемой функции.

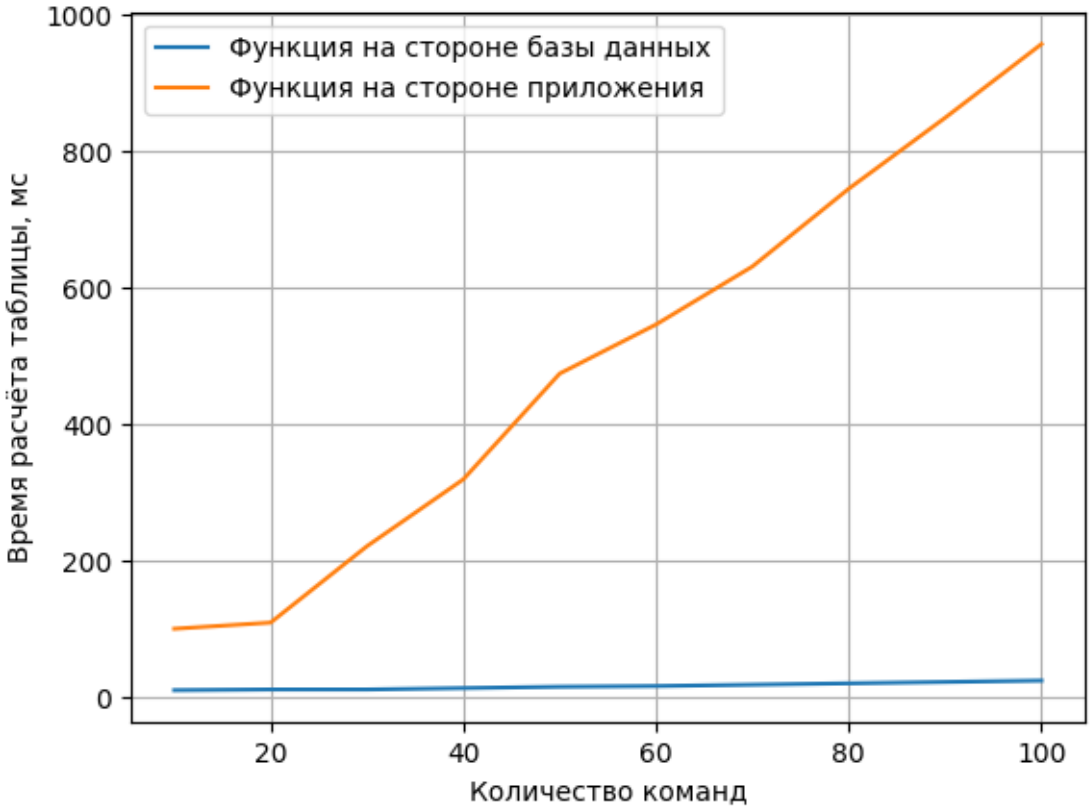


Рисунок 20 – Зависимость времени расчёта турнирной таблицы в зависимости от количества команд и используемой функции

Вывод

В результате эксперимента было установлено, что время расчёта турнирной таблицы линейно зависит от количества команд, участвующих в турнире, вне зависимости от того, какая функция используется — функция на стороне базы данных или на стороне приложения. Также выяснилось, что время расчёта таблицы при использовании функции на стороне приложения в $38,32$ раз дольше, чем при использовании функции на стороне базы данных при количестве команд, равном 100.

ЗАКЛЮЧЕНИЕ

В ходе выполнения курсовой работы была разработана база данных для проведения футбольных турниров, а также разработано приложение, предоставляющее интерфейс для работы с базой данных. Был проведен анализ существующих баз данных на основе формализации данных и сделан выбор в пользу реляционной БД.

Исследование показало, что время расчёта турнирной таблицы зависит линейно от количества команд, участвующих в турнире.

Поставленная цель была достигнута.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. European leagues & cups [Электронный ресурс]. — Режим доступа: https://www.transfermarkt.com/wettbewerbe/europa (дата обращения: 23.04.2023).
2. Создание таблиц для спортивных игр [Электронный ресурс]. — Режим доступа: https://oball.ru/ru/ (дата обращения: 23.04.2023).
3. Турниры онлайн [Электронный ресурс]. — Режим доступа: https://sportcup.org/ru/ (дата обращения: 23.04.2023).
4. Генератор турнирных сеток и таблиц [Электронный ресурс]. — Режим доступа: https://tlab.pro/ (дата обращения: 23.04.2023).
5. Иерархическая модель данных [Электронный ресурс]. — Режим доступа: https://xstud.org/250378/informatika/ierarhicheskaya_model_dannyh (дата обращения: 23.04.2023).
6. Что такое реляционная база данных [Электронный ресурс]. — Режим доступа: https://www.oracle.com/cis/database/what-is-a-relational-database/ (дата обращения: 23.04.2023).
7. Общие сведения о LINQ [Электронный ресурс]. — Режим доступа: https://learn.microsoft.com/ru-ru/dotnet/standard/linq/ (дата обращения: 20.05.2023).
8. Общие сведения о платформе .NET - .NET Framework [Электронный ресурс]. — Режим доступа: https://learn.microsoft.com/ru-ru/dotnet/framework/get-started/overview (дата обращения: 20.05.2023).
9. Что такое Windows Forms - Windows Forms .NET [Электронный ресурс]. — Режим доступа: https://learn.microsoft.com/ru-ru/dotnet/desktop/winforms/overview/?view=netdesktop-6.0 (дата обращения: 20.05.2023).
10. Intel® Core™ i5-1135G7 Processor [Электронный ресурс]. — Режим доступа: https://ark.intel.com/content/www/ru/ru/ark/products/208658/intel-core-i51135g7-processor-8m-cache-up-to-4-20-ghz.html (дата обращения: 20.05.2023).

ПРИЛОЖЕНИЕ А

Презентация состоит из 12 страниц.