

Санкт-Петербургский политехнический университет Петра Великого
Институт компьютерных наук и технологий
Кафедра «Компьютерные системы и программные технологии»

КУРСОВОЙ ПРОЕКТ

по курсу «Базы данных»

Выполнил
студент гр.3530901/70202

(подпись)

Павлов Д.В.
(инициалы, фамилия)

Руководитель

(подпись)

Мяснов А.В.
(инициалы, фамилия)

«__» _____ 2020 г.

Санкт-Петербург
2020

Санкт-Петербургский государственный политехнический университет

**ЗАДАНИЕ
НА ВЫПОЛНЕНИЕ КУРСОВОГО ПРОЕКТА**

студенту группы 3530901/70202
(номер группы)

Павлову Даниилу Вячеславовичу
(фамилия, имя, отчество)

- 1. Срок сдачи законченного проекта** 30.05.2020
- 2. Исходные данные к проекту:** Задание для курсового проекта.
- 3. Содержание пояснительной записки** (перечень подлежащих разработке вопросов): техническое задание, реализация, вывод, листинги.

Дата получения задания: «10» мая 2020 г.

Руководитель

(подпись)

Мяснов А.В.
(инициалы, фамилия)

Задание принял к исполнению

(подпись студента)

Павлов Д.В.
(инициалы, фамилия)

(дата)

Содержание

1. Техническое задание	4
1.1. Постановка задачи	4
1.2. Возможности приложения	4
1.3. План разработки	4
2. Ход выполнения работы	5
2.1. Структура базы данных	5
2.2. Структура консольного приложения.....	6
2.3. Обработка ошибок	19
3. Вывод	22
4. Листинги.....	23
4.1. Код базы данных	23
4.2. Код консольного приложения	26

1. Техническое задание

1.1. Постановка задачи

Необходимо разработать консольное приложение, позволяющее обычным пользователям просматривать и экспортировать данные в удобном виде, а администраторам – вставлять и обновлять их. Само приложение должно быть основано на базе данных футбольных лиг, созданной в течение семестра.

1.2. Возможности приложения

- Просмотр таблиц и их экспорт в виде .csv файла
- Просмотр заготовленных запросов с настраиваемыми параметрами
- Написание своих собственных запросов
- Вставка и обновление данных (для администраторов)

1.3. План разработки

- Обдумывание проекта в бумажном варианте
- Написание функций, описанных в п.1.2.
- Организация взаимодействия функций друг с другом
- Тестирование приложения
- Оценка результата

2. Ход выполнения работы

2.1. Структура базы данных

В качестве базы данных была взята разработанная мной в ходе лабораторных работ с использованием языка PostgreSQL база – “Football leagues”.

Данная БД до момента реализации консольного приложения состояла из 11 таблиц. Все они подробно описаны в отчете по лабораторным работам. Но, так как была поставлена цель разделить пользователей на 2 категории (обычные и администраторы), пришлось добавить еще 1 таблицу, содержащую логин, пароль и статус пользователя. Пароли были должным образом захешированы с помощью функции `crypt()` и засолены функцией `gen_salt()` (для всего этого требуется `CREATE EXTENSION pgcrypto`).

Соль – это случайная строка данных, хэшированная вместе с паролем, для сохранения уникальности результата хэширования. Соли следует воссоздавать каждый раз, когда новый пароль сохраняется, а соль сохраняется вместе с хэшированным результатом, чтобы ее можно было использовать снова для сравнения. Все это придумано потому, что злоумышленник может иметь предварительно вычисленную таблицу хэшей, а также может иметь шанс проверить каждый найденный хэш и посмотреть совпадает ли он с записью в таблице. Если это так, он может эффективно развернуть хэш и получить оригинальный текст. Вот поэтому нам и нужна соль.

Полный код для создания БД представлен в разделе “Листинги”-“Код базы данных”

Схема базы данных представлена на Рис.2.1.1.:

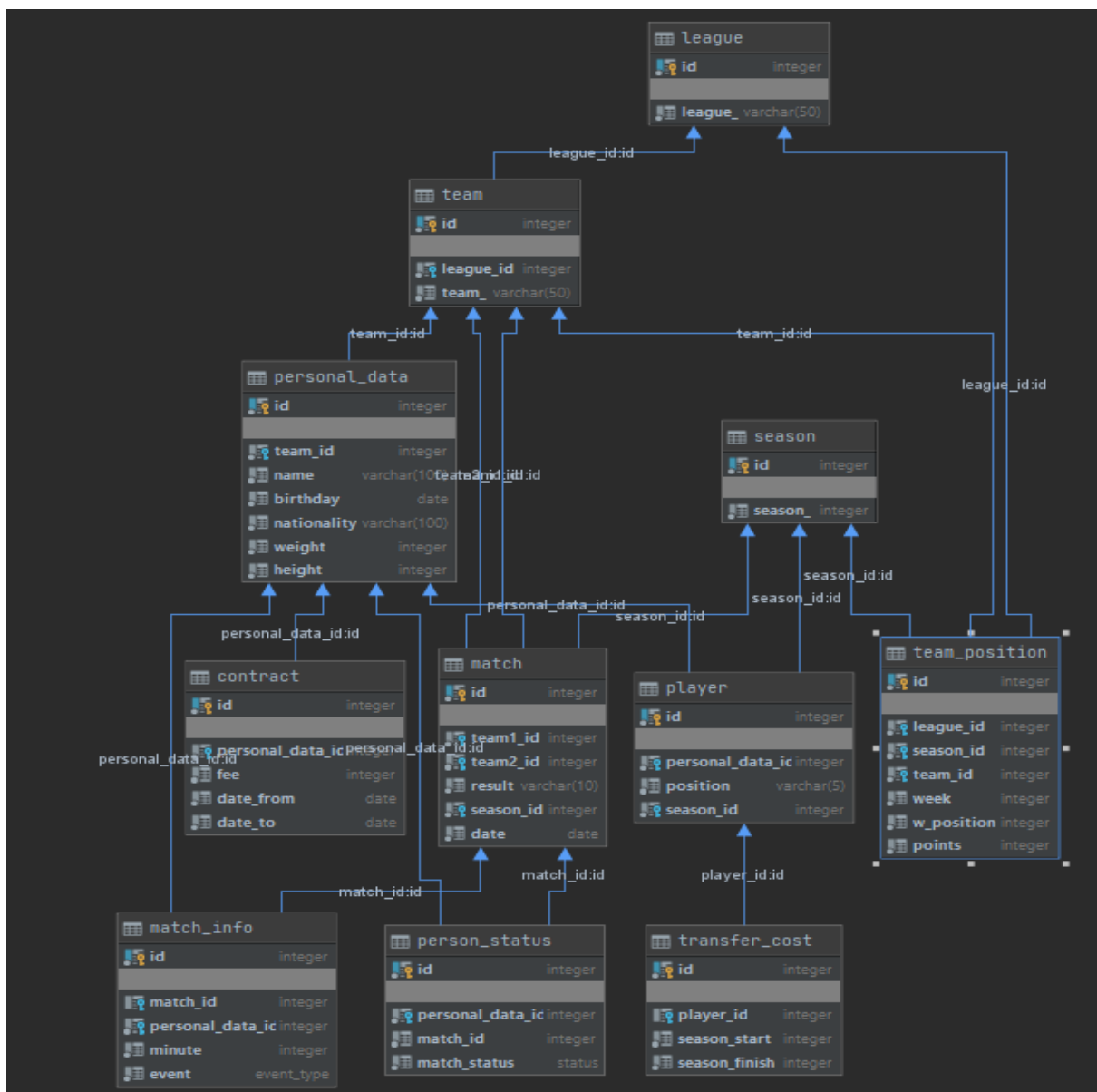


Рис.2.1.1. Схема базы данных

2.2. Структура консольного приложения

В качестве языка программирования для написания работы, как и во время выполнения лабораторных работ, был выбран Python, поэтому проектирование проходило в хорошем темпе.

В начале работы приложение выводит приветственное сообщение и спрашивает, желаем ли мы войти (Рис.2.2.1.). Если ответ пользователя отрицательный, программа завершает свою работу.

```
FOOTBALL LEAGUES

Welcome to the football leagues database!
Do you want to login?[1=yes][0=no]:
```

Рис.2.2.1. Приветственное сообщение

В случае положительного ответа, от пользователя требуется ввести логин и пароль. Если логина не окажется в базе, программа попросит ввести его снова. Если пароль введен неверно – пользователю потребуется ввести его снова, логин повторно вводить не потребуется.

В зависимости от того является пользователь обычным или администратором, перед ним предстанут разные возможности. На Рис.2.2.2 представлено меню для обычного пользователя, а на Рис.2.2.3 – для администратора.

```
FOOTBALL LEAGUES

Welcome to the football leagues database!
Do you want to login?[1=yes][0=no]: 1
Type your login: a
Type your password: 123

You logged in as User!

1 - Watch tables
2 - Use prepared queries
3 - Write your queries (for advanced users)
4 - Export tables
0 - Change user or exit

Make a choice:
```

Рис.2.2.2. Меню обычного пользователя

```
Type your login: b
Type your password: 456

You logged in as Admin!

1 - Watch tables
2 - Use prepared queries
3 - Write your queries (for advanced users)
4 - Export tables
5 - Import your data
6 - Update data
0 - Change user or exit

Make a choice:
```

Рис.2.2.3. Меню администратора

На рисунках выше показано, что пользователь может: просмотреть таблицы, использовать заготовленные запросы, написать свои запросы или экспортировать таблицы. Администратор, в свою очередь, может в добавок к этому импортировать или обновлять данные.

Далее требуется сделать выбор. Если мы вводим число вне диапазона или другие символы, приложение нам об этом скажет и предложит сделать выбор снова.

1) Визуализация 1 пункта основного меню представлена на Рис.2.2.4:

```
Make a choice: 1

Tables:
1 - league
2 - team
3 - season
4 - team_position
5 - player
6 - match
7 - personal_data
8 - match_info
9 - contract
10 - transfer_cost
11 - person_status
0 - Back

Choose table:
```

Рис.2.2.4. Watch tables

После выбора отобразится таблица (Рис.2.2.5) и у нас спросят, хотим ли мы экспортировать таблицу. Эта функция также выделена в основном меню отдельно, чтобы пользователь сразу мог получить ее в виде файла.

```
Choose table: 1

Leagues

|  id | league_      |
|-----+-----|
|  1 | Liechtenstein |
|  2 | Lithuania     |
|  3 | Austria       |
|  4 | NorthKorea    |
|  5 | Ethiopia      |
|  6 | Tanzania      |
|  7 | Netherlands   |

Do you want to export this query?[1=yes,0=no]0
```

Рис.2.2.5. Watch tables вывод

2) Визуализация 2 пункта основного меню представлена на Рис.2.2.6:

```
Make a choice: 2

Queries:
1 - The list of the players who at the start of the season had value you enter and cost more than at the end.
2 - End season results for the team with name you entered.
3 - Events of the man with name you entered.
4 - Squad of the team with name you entered.
5 - Top 20 players with weight & height characteristics you choose.
6 - Players with nationality you entered.
7 - Number of players with assists & goals & substitutions characteristics you choose.
0 - Back

Choose query:
```

Рис.2.2.6. Use prepared queries

Перед нами предстаёт список запросов, которые может использоваться пользователь (он легко может пополняться в будущем). В каждом варианте написано, какой параметр пользователь может устанавливать сам. После вывода таблицы мы можем наблюдать число строк, которое в ней содержится, и также вопрос хотим ли мы экспортировать запрос или нет. Название для файла мы можем придумать сами, но расширение у него будет .csv. На Рис.2.2.7-2.2.13 представлены выборы пунктов:

```

Enter cost: 140000000

Cost: 140000000

|   season | team       | name           |   season_finish |   season_start |
|-----+-----+-----+-----+-----|
|   2018 | LGDGaming | KASEY BLIDE    |   143683176 |   145455757 |
|   2018 | H2K       | RALEIGH GIRARDI |   142244669 |   142372273 |
|   2019 | Leipzig   | SANTOS SWANN   |   143210995 |   145131804 |
|   2019 | NorwichCity | TYSON SWIATKOWSKI |   144926353 |   145636436 |
|   2019 | M-gladbach | HORACE WADDLE   |   140916882 |   146883079 |
|   2019 | TeamKinguin | FRANCISCO BUCHMILLER |   142922102 |   149746819 |
|   2019 | Leipzig   | NOE KELLAWAY   |   142476671 |   145200667 |
|   2019 | ROXTigers | LONDON GOIST    |   149074289 |   149159093 |

The number of rows in the result set : 8

Do you want to export this query?[1=yes,0=no]

```

Рис.2.2.7. 1 запрос

```

Enter teams name: Augsburg

End season results for the team with name: Augsburg

|   season | league       | week | team   | w_position | points |
|-----+-----+-----+-----+-----+-----|
|   2019 | Liechtenstein |   38 | Augsburg |   10 |   56 |
|   2018 | Liechtenstein |   38 | Augsburg |    1 |   68 |
|   2017 | Liechtenstein |   38 | Augsburg |   20 |   40 |
|   2016 | Liechtenstein |   38 | Augsburg |    7 |   58 |
|   2015 | Liechtenstein |   38 | Augsburg |    3 |   63 |
|   2014 | Liechtenstein |   38 | Augsburg |   13 |   50 |
|   2013 | Liechtenstein |   38 | Augsburg |   11 |   54 |

The number of rows in the result set : 7

Do you want to export this query?[1=yes,0=no]

```

Рис.2.2.8. 2 запрос

Choose query: 3

Enter person name: FARRUKH SHARIPOV

Man events with name: FARRUKH SHARIPOV

team	name	match_id	minute	event
Augsburg	FARRUKH SHARIPOV	31	84	goal
Augsburg	FARRUKH SHARIPOV	31	90	goal
Augsburg	FARRUKH SHARIPOV	51	91	goal
Augsburg	FARRUKH SHARIPOV	101	34	assist
Augsburg	FARRUKH SHARIPOV	161	43	goal
Augsburg	FARRUKH SHARIPOV	181	61	assist
Augsburg	FARRUKH SHARIPOV	221	54	goal
Augsburg	FARRUKH SHARIPOV	231	43	goal
Augsburg	FARRUKH SHARIPOV	241	23	goal
Augsburg	FARRUKH SHARIPOV	261	66	assist
Augsburg	FARRUKH SHARIPOV	771	73	replaced

....

Augsburg	FARRUKH SHARIPOV	16291	54	replaced
Augsburg	FARRUKH SHARIPOV	16301	43	goal
Augsburg	FARRUKH SHARIPOV	16311	70	replaced

The number of rows in the result set : 74

Do you want to export this query?[1=yes,0=no]

Рис.2.2.9. 3 запрос

Enter teams name: *Fiorentina*

Squad of the team with name: Fiorentina

league	team	name	birthday	nationality
Liechtenstein	Fiorentina	GIL MINION	1984-09-14	Niger
Liechtenstein	Fiorentina	CHRISTOPHER WRIGLEY	1985-01-08	Somalia
Liechtenstein	Fiorentina	TAYLOR CLUSTER	1986-10-08	Sudan
Liechtenstein	Fiorentina	NELSON AHLBERG	1985-02-19	Afghanistan
Liechtenstein	Fiorentina	VINCENZO KIRALY	1994-01-22	Algeria
Liechtenstein	Fiorentina	MITCHEL RAYNES	1994-06-08	Indonesia
Liechtenstein	Fiorentina	DAVE PAULAUSKIS	1990-08-09	Guyana
Liechtenstein	Fiorentina	PETE OLIFF	1986-06-08	Ethiopia
Liechtenstein	Fiorentina	BERNARDO MUSILLI	1988-03-07	Kuwait
Liechtenstein	Fiorentina	DEXTER SHEN	1990-06-04	Swaziland
Liechtenstein	Fiorentina	JASPER RENDER	1993-07-08	Denmark
Liechtenstein	Fiorentina	DARRICK DIGREGORIO	1986-01-25	Grenada
Liechtenstein	Fiorentina	LES GRANSTROM	1987-04-19	Spain
Liechtenstein	Fiorentina	DELMAR GODINO	1984-02-14	Portugal
Liechtenstein	Fiorentina	ZACHARIAH BLUMENFELD	1985-02-13	Iraq
Liechtenstein	Fiorentina	JESSE MALOON	1995-11-13	SouthSudan
Liechtenstein	Fiorentina	BLAINE ZECHES	1989-07-09	Djibouti
Liechtenstein	Fiorentina	HOMER WOHLSCHEGEL	1994-07-26	Australia
Liechtenstein	Fiorentina	ANDERSON CORNELIA	1986-08-20	Guernsey
Liechtenstein	Fiorentina	DAVID MCKELLOP	1990-06-05	Tuvalu

The number of rows in the result set : 20

Do you want to export this query?[1=yes,0=no]

Рис.2.2.10. 4 запрос

```

Choose query: 5

Top 20:
1 - The biggest weight.
2 - The biggest height.
3 - The smallest weight.
4 - The smallest height.

Choose : 1

Enter league name: Liechtenstein

Top 20 with the biggest weight in league: Liechtenstein

| league | team | name | birthday | nationality | weight | height |
|-----|-----|-----|-----|-----|-----|-----|
| Liechtenstein | Chelsea | CECIL FOSS | 1992-08-06 | Lithuania | 90 | 166 |
| Liechtenstein | 4Kings | LUCIANO NESBITT | 1984-04-07 | SanMarino | 90 | 186 |
| Liechtenstein | NorwichCity | JIMMY CHOCK | 1991-10-20 | Moldava | 90 | 179 |
| Liechtenstein | LuminosityGaming | LINCOLN FILIPPONE | 1993-06-27 | Thailand | 90 | 198 |
| Liechtenstein | G2Esports | AMBROSE WUESTE | 1986-05-14 | Kuwait | 90 | 167 |
| Liechtenstein | Chelsea | HUBERT SOROTZKIN | 1988-04-03 | Liberia | 90 | 186 |
| Liechtenstein | Hwaseung0Z | LAMONT FUERBRINGER | 1988-03-21 | UnitedKingdom | 90 | 176 |
| Liechtenstein | NoNameGaming | MOHAMMED CORCORAN | 1988-03-16 | Syria | 90 | 193 |
| Liechtenstein | Wolfsburg | TERRANCE GUTZWILLER | 1986-11-02 | Liberia | 90 | 196 |
| Liechtenstein | TeamLiquid | LYNWOOD ROWER | 1990-01-18 | Nigeria | 89 | 169 |
| Liechtenstein | Chelsea | EFRAIN BERENGER | 1993-11-01 | Finland | 89 | 187 |
| Liechtenstein | Sampdoria | PORFIRIO BREITEN | 1990-12-27 | Laos | 89 | 193 |
| Liechtenstein | 4Kings | MALIK BALMOS | 1995-05-01 | Chile | 89 | 175 |
| Liechtenstein | Paderborn | ARON ASKIEW | 1990-02-11 | Aruba | 89 | 179 |
| Liechtenstein | RoyalNeverGiveUp | THADDEUS TLLO | 1995-01-02 | SanMarino | 89 | 161 |
| Liechtenstein | Paderborn | REY CHERN | 1992-07-22 | NewZealand | 89 | 161 |
| Liechtenstein | TeamOG | DARRIN POINDEXTER | 1987-11-24 | Antarctica | 88 | 173 |
| Liechtenstein | LuminosityGaming | ANTIONE SHORB | 1990-01-03 | Uruguay | 88 | 189 |
| Liechtenstein | Valladolid | IKE BALSIGER | 1992-10-15 | Zimbabwe | 88 | 173 |
| Liechtenstein | Augsburg | ARDEN NOGUERA | 1990-04-09 | Armenia | 88 | 198 |

The number of rows in the result set : 20

Do you want to export this query?[1=yes,0=no]

```

Рис.2.2.11. 5 запрос

```

Choose query: 6

Enter nationality: Russia

List of players with nationality: Russia

| league | team | name | birthday | nationality |
|-----|-----|-----|-----|-----|
| Austria | TeamAllegiance | DAMIAN PEKAR | 1991-02-04 | Russia |
| Austria | NatusVincere | ISIDRO EICHORST | 1984-12-21 | Russia |
| NorthKorea | TeamDK | HERMAN GIACALONE | 1995-12-14 | Russia |
| Liechtenstein | Hwaseung0Z | LYMAN HEMKER | 1989-10-05 | Russia |
| Liechtenstein | TeamLiquid | ANTONY YLONEN | 1989-05-11 | Russia |
| Lithuania | Splyce | TAYLOR WOHLTZ | 1988-06-05 | Russia |
| Lithuania | Leipzig | ETHAN BETZOLD | 1985-11-08 | Russia |
| Tanzania | OMG | COURTNEY DANEKER | 1994-12-01 | Russia |
| Tanzania | OMG | GARLAND WENGERT | 1995-08-24 | Russia |
| Netherlands | LuminosityGaming | VIRGIL ARMELIN | 1995-01-01 | Russia |
| Netherlands | WoongjinStars | JOHN ANCHONDO | 1991-02-24 | Russia |
| Netherlands | WoongjinStars | GREGORIO KORNEGAY | 1984-11-07 | Russia |

The number of rows in the result set : 12

Do you want to export this query?[1=yes,0=no]

```

Рис.2.2.12. 6 запрос

```
Choose query: 7

Results for:
1 - Certain season.
2 - Certain league.
3 - All seasons and all leagues.
4 - Certain season and certain league.

Choose : 4

Events:
1 - Goals.
2 - Assists.
3 - Substitutions.

Choose : 1

Enter season: 2019

Enter league: Tanzania

Enter limit: 10

Your top:

| season | league | team | name | count |
|-----+-----+-----+-----+-----|
| 2019 | Tanzania | CJEntus | JULIAN POLIKOFF | 10 |
| 2019 | Tanzania | TongFu | ELIJAH HELMSTETTER | 9 |
| 2019 | Tanzania | Mallorca | ARCHIE BURLETT | 9 |
| 2019 | Tanzania | NRGEsports | VERNON GRAVITO | 9 |
| 2019 | Tanzania | RealMadrid | JUDE DERRAH | 8 |
| 2019 | Tanzania | TongFu | CHESTER GOLDRICH | 8 |
| 2019 | Tanzania | OMG | COURTNEY DANEKER | 8 |
| 2019 | Tanzania | Frankfurt | VAL MALKASIAN | 8 |
| 2019 | Tanzania | M-gladbach | VICTOR KRUG | 8 |
| 2019 | Tanzania | Frankfurt | PAUL FAHLSTEDT | 8 |

The number of rows in the result set : 10

Do you want to export this query?[1=yes,0=no]
```

Рис.2.2.13. 7 запрос

7 запрос является самым интересным для нас (хотя бы на данный момент), так как здесь самый большой выбор действий. Сначала мы должны выбрать, хотим ли мы составить топ для конкретного сезона или лиги, конкретного сезона и лиги или по всем сезонам и лигам. После мы выбираем топ чего мы будем делать – голов, ассистов или замен. Далее мы выбираем лимит. Если лимит будет больше, чем данных в таблице, то просто выведутся все данные.

На этом мы можем закончить со 2 пунктом.

3) Визуализация 3 пункта основного меню представлена на Рис.2.2.14:

```
Make a choice: 3

Pro queries:
1 - Write query.
0 - Back

Choose action: 1

Enter your query: SELECT * FROM personal_data where name LIKE 'BULAT KHISAMUTDINOV'
Your query: SELECT * FROM personal_data where name LIKE 'BULAT KHISAMUTDINOV'
- - -
2 1 BULAT KHISAMUTDINOV 1988-02-16 Guadeloupe 77 189
- - -
The number of rows in the result set : 1

Do you want to export this query?[1=yes,0=no]1

Type filename: BULAT
```

Рис.2.2.14. Write your queries

Все довольно очевидно. Задача человека в этом пункте написана в самом названии – создать запрос самому. Однако, если человек сделает в запросе ошибку, либо захочет сделать что-то плохое (удалить данные или таблицы), то у него этого не выйдет. Система основана на принципе “белого листа”, то есть если первым оператором будет не SELECT, то программа не пропустит запрос и выдаст ошибку (Рис.2.2.15.):

```
Enter your query: DELETE FROM users
Your query: DELETE FROM users
Error. You are only allowed to use SELECT here. Try again.
```

Рис.2.2.15. Попытка использования не оператора SELECT

4) Визуализация 4 пункта основного меню представлена на Рис.2.2.16:

```
Make a choice: 4

Tables:
1 - league
2 - team
3 - season
4 - team_position
5 - player
6 - match
7 - personal_data
8 - match_info
9 - contract
10 - transfer_cost
11 - person_status
0 - Back

Choose table:
```

Рис.2.2.16. Export tables

Как было описано ранее, этот режим частично добавлен в 1 пункт, но здесь человек сразу может экспортировать таблицу, без дополнительного вывода данных в консоль.

5) Визуализация 5 пункта основного меню представлена на Рис.2.2.17-2.2.18:

```
Make a choice: 5

Import:
1 - Insert data.
2 - Insert new user
0 - Back

Choose action: 1

Enter your query: INSERT into personal_data (team_id, name, birthday, nationality, weight, height) VALUES ('1', 'VASILIIY MAKSEM', '10-10-1999', 'Russia', 73,173)
Your query: INSERT into personal_data (team_id, name, birthday, nationality, weight, height) VALUES ('1', 'VASILIIY MAKSEM', '10-10-1999', 'Russia', 73,173)
```

Рис.2.2.17. Insert data

```
Choose action: 2

Enter user name: new

Enter user password: 897

Enter user status: user
```

Рис.2.2.18. Insert new user

Никаких ошибок получено не было. Теперь попробуем добавить в таблицу пользователя с логином, который уже существует в базе (Рис.2.2.20):

```
Choose action: 2

Enter user name: new

Enter user password: 123

Enter user status: admin

Error. Your query syntax is incorrect or this login already exists. Try again.
```

Рис.2.2.20. Login already exists

Выдается сообщение, что пользователь с таким именем существует и операция не выполняется, что является верным подходом.

Теперь создадим xml файл и попробуем его импортировать (Рис.2.2.21):

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <football>
3
4  <personal_data>
5    <team_id>1</team_id>
6    <name>someone</name>
7    <birthday>8-09-1984</birthday>
8    <nationality>Tanzania</nationality>
9    <weight>80</weight>
10   <height>180</height>
11  </personal_data>
12
13  <match>
14    <team1_id>1</team1_id>
15    <team2_id>100</team2_id>
16    <result>0-4</result>
17    <season_id>2</season_id>
18    <date>2018-07-19</date>
19  </match>
20
21  <contract>
22    <personal_data_id>5</personal_data_id>
23    <fee>50000</fee>
24    <date_from>2020-05-19</date_from>
25    <date_to>2025-05-19</date_to>
26  </contract>
27
28  <personal_data>
29    <team_id>2</team_id>
30    <name>somnoone</name>
31    <birthday>7-09-1984</birthday>
32    <nationality>Tanzania</nationality>
33    <weight>70</weight>
34    <height>190</height>
35  </personal_data>
48  <personal_status>
49    <personal_data_id>100</personal_data_id>
50    <match_id>1</match_id>
51    <match_status>sub</match_status>
52  </personal_status>
53
54  <player>
55    <personal_data_id>50</personal_data_id>
56    <position>ST</position>
57    <season_id>2</season_id>
58  </player>
59
60  <season>
61    <season_id>2012</season_id>
62  </season>
63
64  <team>
65    <league_id>5</league_id>
66    <team_name>Politech</team_name>
67  </team>
68
69  <team_position>
70    <league_id>5</league_id>
71    <season_id>1</season_id>
72    <team_id>sub</team_id>
73    <week>39</week>
74    <w_position>1</w_position>
75    <points>100</points>
76  </team_position>
77
78  <transfer_cost>
79    <player_id>5</player_id>
80    <season_start>5000000</season_start>
81    <season_finish>1000000</season_finish>
82  </transfer_cost>
83
84  </football>
```

Рис.2.2.21. Xml файл

Добавление произошло успешно. Также, можно заметить, что входные данные для таблицы “personal_data” находятся не друг за другом, но все проходит. Это связано с тем, что я использую функцию findall(“table name”) в цикле. Если мы импортируем тот же файл еще раз, то получим следующую информацию в выводе консоли (Рис.2.2.22):

```

Make a choice: 5

Import:
1 - Insert data.
2 - Insert new user
3 - Import xml file
0 - Back

Choose action: 3
Error. Data you are trying to import exists in personal_data table.
Error. Data you are trying to import exists in personal_data table.
Error. Data you are trying to import exists in match table.
Error. Data you are trying to import exists in contract table.
Error. Data you are trying to import exists in league table.
Error. Data you are trying to import exists in match_info table.
Error. Data you are trying to import exists in person_status table.
Error. Data you are trying to import exists in player table.
Error. Data you are trying to import exists in season table.
Error. Data you are trying to import exists in team table.
Error. Data you are trying to import exists in team_position table.
Error. Data you are trying to import exists in transfer_cost table.

```

Рис.2.2.22. Дубликаты строк

Так как мы не хотим дублировать строки, в базе данных на таблицах установлены операторы UNIQUE, которые предотвращают данный шаг.

б) Визуализация 6 пункта основного меню представлена на Рис.2.2.23.

Зафиксируем обновляемые данные, чтобы удостовериться в корректности работы оператора UPDATE (Рис.2.2.24):

id	team_id	name	birthday	nationality	weight	height
2803	1	VASILYIY MAKSEM	1999-10-10	Russia	73	173

Рис.2.2.23. Before update

```

Make a choice: 6

Update:
1 - Update data.
0 - Back

Choose action: 1

Enter your query: update personal_data set height = 179, weight = 70 where id = 2803
Your query: update personal_data set height = 179, weight = 70 where id = 2803

Update:
1 - Update data.
0 - Back

Choose action:

```

Рис.2.2.24. Update data

Посмотрим, что стало с нашими данными (Рис.2.2.25):

id	team_id	name	birthday	nationality	weight	height
2803	1	VASILIIY MAKSEM	1999-10-10	Russia	70	179

Рис.2.2.25. After update

Судя по результатам, все работает верно.

2.3. Обработка ошибок

Перейдем к не менее важной теме. Здесь будет описаны и показаны моменты, не вошедшие в предыдущий пункт.

Начнем со входа в приложение (Рис.2.3.1):

```
Type your login: wr
Login that you typed does not exist. Try again.

Type your login: a
Type your password: 567

Password is incorrect
Type your password: 123

You logged in as User!
```

Рис.2.3.1. Попытка входа

Сначала пользователь ввел неверный логин, его попросили повторить попытку. После того, как ему это удалось, он не смог ввести верный пароль. Далее ему предлагают ввести пароль снова и у него все получается.

Далее рассмотрим пример выбора вне диапазона (Рис.2.3.2):

```

You logged in as User!

1 - Watch tables
2 - Use prepared queries
3 - Write your queries (for advanced users)
4 - Export tables
0 - Change user or exit

Make a choice: 7
Error. Your choice was out of range. Try again.

You logged in as User!

1 - Watch tables
2 - Use prepared queries
3 - Write your queries (for advanced users)
4 - Export tables
0 - Change user or exit

```

Рис.2.3.2. Выбор вне диапазона

Теперь попробуем ввести в поле, где требуется число, строку (Рис.2.3.3):

```

Choose query: 1

Enter cost: выфффффф
Error. Input data is incorrect. Try again.

Queries:
1 - The list of the players who at the start of the season had value you enter and cost more than at the end.
2 - End season results for the team with name you entered.
3 - Events of the man with name you entered.
4 - Squad of the team with name you entered.
5 - Top 20 players with weight & height characteristics you choose.
6 - Players with nationality you entered.
7 - Number of players with assists & goals & substitutions characteristics you choose.
0 - Back

```

Рис.2.3.3. Строка вместо числа

Также, если в выборе написать строку, ошибка тоже будет (Рис.2.3.4):

```

Choose query: выфффф
Error. You did not type number. Try again.

```

Рис.2.3.4. Строка в выборе

Если мы напишем запрос с ошибкой (в 3 разделе), то получим следующее сообщение (Рис.2.3.5):

```
Pro queries:
1 - Write query.
0 - Back

Choose action: 1

Enter your query: SELECT * from personal_data where league_id = 20
Your query: SELECT * from personal_data where league_id = 20
Error. Your query syntax is incorrect. Try again.

Pro queries:
1 - Write query.
0 - Back

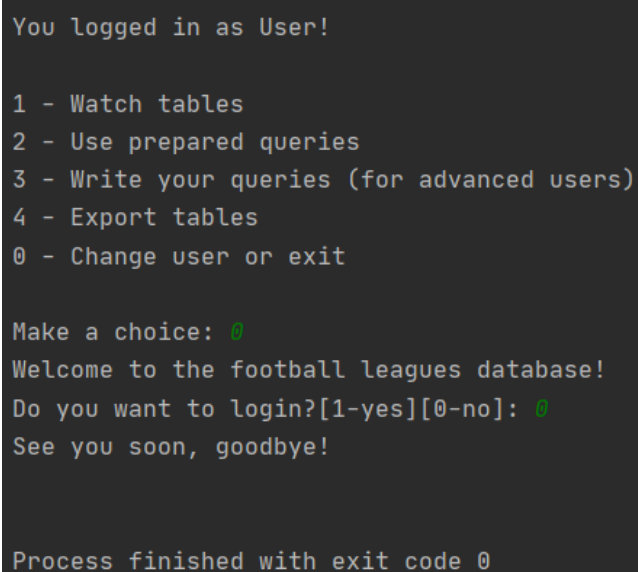
Choose action:
```

Рис.2.3.5. Запрос с ошибкой

3. Вывод

В ходе разработки данного консольного приложения были закреплены знания и практические навыки, полученные в течение всего курса. Также был приобретен опыт работы с одной из самых современных СУБД – PostgreSQL. Она является широко используемой СУБД в России и мире, поэтому полученный мною опыт будет очень полезным в разных компаниях.

Ну и напоследок, выход из приложения сопровождается нас приятной фразой – “See you soon, goodbye!” (Рис.3.1):



```
You logged in as User!

1 - Watch tables
2 - Use prepared queries
3 - Write your queries (for advanced users)
4 - Export tables
0 - Change user or exit

Make a choice: 0
Welcome to the football leagues database!
Do you want to login?[1=yes][0=no]: 0
See you soon, goodbye!

Process finished with exit code 0
```

Рис.3.1. Goodbye!

4. Листинги

4.1. Код базы данных

```
DROP TABLE IF EXISTS
league,team,personal_data,transfer_cost,contract,player,match,match_info,person_status,team_positi
on,users;
DROP TYPE IF EXISTS status, event_type;
DROP EXTENSION pgcrypto;
CREATE EXTENSION pgcrypto;

CREATE TABLE IF NOT EXISTS public.users
(
    id      SERIAL    NOT NULL,
    login   TEXT      NOT NULL UNIQUE,
    password TEXT      NOT NULL,
    user_status VARCHAR(10) NOT NULL,
    PRIMARY KEY (id)
);

CREATE TABLE IF NOT EXISTS public.league
(
    id      SERIAL    NOT NULL,
    league_ VARCHAR(50) NOT NULL,
    PRIMARY KEY (id)
);

ALTER TABLE league ADD UNIQUE (league_);

CREATE TABLE IF NOT EXISTS public.team
(
    id      SERIAL    NOT NULL,
    league_id int      NOT NULL,
    team_   VARCHAR(50) NOT NULL,
    PRIMARY KEY (id),

    CONSTRAINT league_id_foreign
        FOREIGN KEY (league_id) REFERENCES public.league (id)
        ON DELETE NO ACTION
        ON UPDATE NO ACTION
);

ALTER TABLE team ADD UNIQUE (league_id, team_);

CREATE TABLE IF NOT EXISTS public.personal_data
(
    id      SERIAL    NOT NULL,
    team_id int      NOT NULL,
    name    VARCHAR(100) NOT NULL,
    birthday date     NOT NULL,
    nationality VARCHAR(100) NOT NULL,
    weight  int       NOT NULL,
    height  int       NOT NULL,
    PRIMARY KEY (id),

    CONSTRAINT team_id_foreign
        FOREIGN KEY (team_id) REFERENCES public.team (id)
        ON DELETE NO ACTION
        ON UPDATE NO ACTION
);

ALTER TABLE personal_data ADD UNIQUE (team_id, name, nationality,birthday, weight, height);
```

```

CREATE TABLE IF NOT EXISTS public.contract
(
    id          SERIAL NOT NULL,
    personal_data_id int  NOT NULL,
    fee         int  NOT NULL,
    date_from   date  NOT NULL,
    date_to     date  NOT NULL,
    PRIMARY KEY (id),

    CONSTRAINT personal_id_foreign
        FOREIGN KEY (personal_data_id) REFERENCES public.personal_data (id)
        ON DELETE NO ACTION
        ON UPDATE NO ACTION
);

ALTER TABLE contract ADD UNIQUE (personal_data_id, fee, date_from, date_to);

CREATE TABLE IF NOT EXISTS public.season
(
    id  SERIAL NOT NULL,
    season_int  NOT NULL,
    PRIMARY KEY (id)
);

ALTER TABLE season ADD UNIQUE (season_);

CREATE TABLE IF NOT EXISTS public.event_types
(
    id SERIAL  NOT NULL,
    type varchar(20) NOT NULL,
    PRIMARY KEY (id)
);

CREATE TABLE IF NOT EXISTS public.player
(
    id          SERIAL  NOT NULL,
    personal_data_id int  NOT NULL,
    position    VARCHAR(5) NOT NULL,
    season_id   int  NOT NULL,
    PRIMARY KEY (id),

    CONSTRAINT personal_id_foreign
        FOREIGN KEY (personal_data_id) REFERENCES public.personal_data (id)
        ON DELETE NO ACTION
        ON UPDATE NO ACTION,
    CONSTRAINT season_id_foreign
        FOREIGN KEY (season_id) REFERENCES public.season (id)
        ON DELETE NO ACTION
        ON UPDATE NO ACTION
);

ALTER TABLE player ADD UNIQUE (personal_data_id, season_id);

CREATE TABLE IF NOT EXISTS public.transfer_cost
(
    id          SERIAL NOT NULL,
    player_id   int  NULL,
    season_start int  NOT NULL,
    season_finish int  NOT NULL,
    PRIMARY KEY (id),

```



```

CONSTRAINT player_id_foreign
  FOREIGN KEY (player_id) REFERENCES public.player (id)
  ON DELETE NO ACTION
  ON UPDATE NO ACTION
);

ALTER TABLE transfer_cost ADD UNIQUE (player_id, season_start, season_finish);

CREATE TABLE IF NOT EXISTS public.match
(
  id      SERIAL      NOT NULL,
  team1_id int        NOT NULL,
  team2_id int        NOT NULL,
  result  VARCHAR(10) NOT NULL,
  season_id int       NOT NULL,
  date    date        NOT NULL,
  PRIMARY KEY (id),

  CONSTRAINT team1_id_foreign
    FOREIGN KEY (team1_id) REFERENCES public.team (id)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT team2_id_foreign
    FOREIGN KEY (team2_id) REFERENCES public.team (id)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT season_id_foreign
    FOREIGN KEY (season_id) REFERENCES public.season (id)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION
);

ALTER TABLE match ADD UNIQUE (team1_id, team2_id, result, season_id, date);

CREATE TYPE status AS ENUM ('main', 'sub');

CREATE TABLE IF NOT EXISTS public.person_status
(
  id          SERIAL NOT NULL,
  personal_data_id int NOT NULL,
  match_id    int    NOT NULL,
  match_status status NOT NULL,
  PRIMARY KEY (id),

  CONSTRAINT personal_id_foreign
    FOREIGN KEY (personal_data_id) REFERENCES public.personal_data (id)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION
);

ALTER TABLE person_status ADD UNIQUE (personal_data_id, match_id, match_status);

CREATE TYPE event_type AS ENUM ('goal', 'assist', 'replaced');

CREATE TABLE IF NOT EXISTS public.match_info
(
  id          SERIAL NOT NULL,
  match_id    int    NULL,
  personal_data_id int NOT NULL,
  minute      int    NOT NULL,
  event       event_type NOT NULL,
  PRIMARY KEY (id),

```

```

CONSTRAINT match_id_foreign
    FOREIGN KEY (match_id) REFERENCES public.match (id)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,

CONSTRAINT person_status_foreign
    FOREIGN KEY (personal_data_id) REFERENCES public.personal_data (id)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION
);

ALTER TABLE match_info ADD UNIQUE (match_id, personal_data_id, minute, event);

CREATE TABLE IF NOT EXISTS public.team_position
(
    id SERIAL NOT NULL,
    league_id int NOT NULL,
    season_id int NOT NULL,
    team_id int NOT NULL,
    week int NOT NULL,
    w_position int NOT NULL,
    points int NOT NULL,
    PRIMARY KEY (id),

    CONSTRAINT league_id_foreign
        FOREIGN KEY (league_id) REFERENCES public.league (id)
        ON DELETE NO ACTION
        ON UPDATE NO ACTION,

    CONSTRAINT season_id_foreign
        FOREIGN KEY (season_id) REFERENCES public.season (id)
        ON DELETE NO ACTION
        ON UPDATE NO ACTION,

    CONSTRAINT team_id_foreign
        FOREIGN KEY (team_id) REFERENCES public.team (id)
        ON DELETE NO ACTION
        ON UPDATE NO ACTION
);

ALTER TABLE team_position ADD UNIQUE (league_id, season_id, team_id, week, points);

```

4.2 Код консольного приложения

```

5 import psycopg2
  from pip._vendor.distlib.compat import raw_input
  from tabulate import tabulate
  from lxml import etree

  connection = psycopg2.connect(dbname='football_league', user='postgres', password='ligaliga',
                                host='127.0.0.1')
  cursor = connection.cursor()

  def log_or_exit():
      while True:
          choice = raw_input('Do you want to login?[1-yes][0-no]: ')
          try:
              if 0 <= int(choice) <= 1:
                  return int(choice)

```

```

        else:
            print('Error. Your choice was out of range. Try again.\n')
    except ValueError:
        print('Error. You did not type number. Try again.')

def log_in():
    print("Welcome to the football leagues database!")
    loggingin = log_or_exit()
    if loggingin == 0:
        cursor.close()
        connection.close()
        print("See you soon, goodbye!\n")
        exit()
    else:
        while True:
            login = str(raw_input("Type your login: "))
            query = "SELECT count(id) FROM users WHERE login = " + str("'" + login + "'")
            cursor.execute(query)
            result = cursor.fetchall()
            if 1 in result[0]:
                while True:
                    write = str(raw_input("Type your password: "))
                    query = "SELECT count(id) FROM users WHERE login = " + str(
                        "'" + login + "'" + " and password = crypt(" + str("'" + write + "'") + ", password)"
                    )
                    cursor.execute(query)
                    result = cursor.fetchall()
                    if 1 in result[0]:
                        query = "SELECT user_status FROM users WHERE login = " + str(
                            "'" + login + "'" + " and password = crypt(" + str("'" + write + "'") + ", password)"
                        )
                        cursor.execute(query)
                        user = cursor.fetchall()
                        return user[0][0]
                    else:
                        print('\nPassword is incorrect')
            print("Login that you typed does not exist. Try again.\n")

def main():
    print("\nFOOTBALL LEAGUES\n")
    user_type = log_in()
    while True:
        if user_type == "user":
            print("\nYou logged in as User!\n")
            user_main_menu()
        else:
            print("\nYou logged in as Admin!\n")
            admin_main_menu()
    choice = make_choice_in_menu()
    if choice == 0:
        user_type = log_in()
    elif choice == 1:
        show_table(cursor)
    elif choice == 2:
        show_query(cursor)
    elif choice == 3:
        execute_pro_query(cursor)
    elif choice == 4:
        export_table(cursor)
    elif choice == 5 and user_type == "admin":
        import_data()
    elif choice == 6 and user_type == "admin":

```

```

update_data()

def user_main_menu():
    print("1 - Watch tables\n"
          "2 - Use prepared queries\n"
          "3 - Write your queries (for advanced users)\n"
          "4 - Export tables\n"
          "0 - Change user or exit\n")

def admin_main_menu():
    print("1 - Watch tables\n"
          "2 - Use prepared queries\n"
          "3 - Write your queries (for advanced users)\n"
          "4 - Export tables\n"
          "5 - Import your data\n"
          "6 - Update data\n"
          "0 - Change user or exit\n")

def make_choice_in_menu():
    choice = raw_input('Make a choice: ')
    try:
        if 0 <= int(choice) <= 6:
            return int(choice)
        else:
            print('Error. Your choice was out of range. Try again.\n')
    except ValueError:
        print('Error. You did not type number. Try again.')

def show_tables_menu():
    print("\nTables:\n"
          "1 - league\n"
          "2 - team\n"
          "3 - season\n"
          "4 - team_position\n"
          "5 - player\n"
          "6 - match\n"
          "7 - personal_data\n"
          "8 - match_info\n"
          "9 - contract\n"
          "10 - transfer_cost\n"
          "11 - person_status\n"
          "0 - Back")

def choose_table():
    choice = raw_input('\nChoose table: ')
    try:
        if 0 <= int(choice) <= 11:
            return int(choice)
        else:
            print('Error. Your choice was out of range. Try again.\n')
    except ValueError:
        print('Error. You did not type number. Try again.')

def show_table(cursor):
    show_tables_menu()
    move_back = False

```

```

while not move_back:
    choice = choose_table()
    if choice == 0:
        move_back = True
        break
    elif choice == 1:
        query_view(cursor, select='select * from public.league;', table_name='Leagues',
                    table_heads=['id', 'league_'])
    elif choice == 2:
        query_view(cursor, select='select * from public.team;', table_name='Teams',
                    table_heads=['id', 'league_id', 'team_'])
    elif choice == 3:
        query_view(cursor, select='select * from public.season;', table_name='Seasons',
                    table_heads=['id', 'season_'])
    elif choice == 4:
        query_view(cursor, select='select * from public.team_position', table_name='Teams Positions',
                    table_heads=['id', 'league_id', 'season_id', 'date', 'team_id', 'week', 'w_position', 'points'])
    elif choice == 5:
        query_view(cursor, select='select * from public.player;', table_name='Players',
                    table_heads=['id', 'personal_data_id', 'position', 'season_id'])
    elif choice == 6:
        query_view(cursor, select='select * from public.match;', table_name='Matches',
                    table_heads=['id', 'team1_id', 'team2_id', 'result', 'season_id', 'date'])
    elif choice == 7:
        query_view(cursor, select='select * from public.personal_data;', table_name='Personal data',
                    table_heads=['id', 'team_id', 'name', 'area', 'birthday', 'nationality', 'weight', 'height'])
    elif choice == 8:
        query_view(cursor, select='select * from public.match_info;', table_name='Match infos',
                    table_heads=['id', 'match_id', 'personal_data_id', 'minute', 'event'])
    elif choice == 9:
        query_view(cursor, select='select * from public.contract;', table_name='Contracts',
                    table_heads=['id', 'personal_data_id', 'fee', 'date_from', 'date_to'])
    elif choice == 10:
        query_view(cursor, select='select * from public.transfer_cost;', table_name='Transfer costs',
                    table_heads=['id', 'player_id', 'season_start', 'season_finish'])
    elif choice == 11:
        query_view(cursor, select='select * from public.person_status;', table_name='Persons statuses',
                    table_heads=['id', 'personal_data_id', 'match_id', 'match_status'])
    show_tables_menu()

```

```

def table_view(cursor, select, table_name, table_heads):
    cursor.execute(select)
    result = cursor.fetchall()
    print('\n' + table_name + '\n')
    print(tabulate(result, headers=table_heads, tablefmt='orgtbl'))

```

```

def show_queries_menu():
    print("\nQueries: \n")
    "1 - The list of the players who at the start of the season had value you enter "
    "and cost more than at the end.\n"
    "2 - End season results for the team with name you entered.\n"
    "3 - Events of the man with name you entered.\n"
    "4 - Squad of the team with name you entered.\n"
    "5 - Top 20 players with weight & height characteristics you choose.\n"
    "6 - Players with nationality you entered.\n"
    "7 - Number of players with assists & goals & substitutions characteristics you choose.\n"
    "0 - Back")

```

```

def characteristics():

```

```

while True:
    print("\nTop 20: \n"
          "1 - The biggest weight.\n"
          "2 - The biggest height.\n"
          "3 - The smallest weight.\n"
          "4 - The smallest height.")
    tmp = raw_input('\nChoose : ')
    try:
        if 1 <= int(tmp) <= 4:
            return int(tmp)
        else:
            print('Error. Your choice was out of range. Try again.\n')
    except ValueError:
        print('Error. You did not type number. Try again.')

def event_type():
    while True:
        print("\nEvents: \n"
              "1 - Goals.\n"
              "2 - Assists.\n"
              "3 - Substitutions.")
        tmp = raw_input('\nChoose : ')
        try:
            if int(tmp) == 1:
                return "goal"
            if int(tmp) == 2:
                return "assist"
            if int(tmp) == 3:
                return "replaced"
            else:
                print('Error. Your choice was out of range. Try again.\n')
        except ValueError:
            print('Error. You did not type number. Try again.')

def seasons_or_leagues():
    while True:
        print("\nResults for: \n"
              "1 - Certain season.\n"
              "2 - Certain league.\n"
              "3 - All seasons and all leagues.\n"
              "4 - Certain season and certain league.")
        tmp = raw_input('\nChoose : ')
        try:
            if 1 <= int(tmp) <= 4:
                return int(tmp)
            else:
                print('Error. Your choice was out of range. Try again.\n')
        except ValueError:
            print('Error. You did not type number. Try again.')

def choose_query():
    choice = raw_input('\nChoose query: ')
    try:
        if 0 <= int(choice) <= 7:
            return int(choice)
        else:
            print('Error. Your choice was out of range. Try again.\n')
    except ValueError:
        print('Error. You did not type number. Try again.')

```

```

def query_export(cursor, query):
    choice = raw_input('\nDo you want to export this query?[1=yes,0=no]')
    try:
        if int(choice) == 1:
            filename = raw_input('\nType filename: ')
            cursor.execute("COPY(" + query + ") TO
'D:\\Files\\Football_league\\console_app\\export_queries\\" + str(
            filename) + ".csv' CSV HEADER")
        elif int(choice) != 0:
            print('Error. Your choice was out of range. Try again.\n')
    except ValueError:
        print('Error. You did not type number. Try again.')

def show_query(cursor):
    show_queries_menu()
    move_back = False
    while not move_back:
        choice = choose_query()
        if choice == 0:
            move_back = True
            break
        elif choice == 1:
            tmp = raw_input('\nEnter cost: ')
            query = "SELECT season.season_, team.team_, personal_data.name, season_finish, season_start " \
            "FROM transfer_cost JOIN personal_data on personal_data.id = player_id and " \
            "season_finish > " + str("'" + tmp + "'") + " and season_start > " \
            "season_finish JOIN player on player.id = player_id " \
            "JOIN season on player.season_id = season.id JOIN " \
            "team on team.id = personal_data.team_id"
            query_view(cursor, select=query, table_name='Cost: ' + str(tmp),
            table_heads=['season', 'team', 'name', 'season_finish', 'season_start'])
        elif choice == 2:
            tmp = raw_input('\nEnter teams name: ')
            query = "SELECT season.season_, league.league_ as league, week, team.team_ as team, w_position,
points " \
            "FROM team_position JOIN team ON team_position.team_id = team.id and team.team_ = " + \
            str("'" + tmp + "'") + " and week = 38 JOIN season ON season_id = season.id " \
            "JOIN league on league.id = team.league_id"
            query_view(cursor, select=query,
            table_name='End season results for the team with name: ' + str(tmp),
            table_heads=['season', 'league', 'week', 'team', 'w_position', 'points'])
        elif choice == 3:
            tmp = raw_input('\nEnter person name: ')
            query = "SELECT team.team_ as team, personal_data.name, match_id, minute, event FROM
match_info JOIN " \
            "personal_data on personal_data.name = " + str("'" + tmp + "'") \
            + " and match_info.personal_data_id = personal_data.id JOIN team on team.id =
personal_data.team_id"
            query_view(cursor, select=query, table_name='Man events with name: ' + str(tmp),
            table_heads=['team', 'name', 'match_id', 'minute', 'event'])
        elif choice == 4:
            tmp = raw_input('\nEnter teams name: ')
            query = "SELECT league.league_ as league, team.team_ as team, name, birthday, nationality FROM " \
            "personal_data JOIN team ON team.id = personal_data.team_id and team_ = " + str("'" + tmp +
            "'") \
            + " JOIN league on league.id = team.league_id"
            query_view(cursor, select=query, table_name='Squad of the team with name: ' + str(tmp),
            table_heads=['league', 'team', 'name', 'birthday', 'nationality'])

```

```

elif choice == 5:
    chr = characteristics()
    tmp = raw_input('\nEnter league name: ')
    if chr == 1:
        name = "Top 20 with the biggest weight in league: "
        query = "SELECT league.league_ as league, team.team_ as team, name, birthday, nationality,
weight, " \
            "height FROM personal_data JOIN team ON team.id = personal_data.team_id JOIN " \
            "league on league.league_ = " + str("'" + tmp + "'") \
            + " and league.id = team.league_id order by weight DESC LIMIT 20"
    if chr == 2:
        name = "Top 20 with the biggest height in league: "
        query = "SELECT league.league_ as league, team.team_ as team, name, birthday, nationality,
weight, " \
            "height FROM personal_data JOIN team ON team.id = personal_data.team_id JOIN " \
            "league on league.league_ = " + str("'" + tmp + "'") \
            + " and league.id = team.league_id order by height DESC LIMIT 20"
    if chr == 3:
        name = "Top 20 with the smallest weight in league: "
        query = "SELECT league.league_ as league, team.team_ as team, name, birthday, nationality,
weight, " \
            "height FROM personal_data JOIN team ON team.id = personal_data.team_id JOIN " \
            "league on league.league_ = " + str("'" + tmp + "'") \
            + " and league.id = team.league_id order by weight LIMIT 20"
    if chr == 4:
        name = "Top 20 with the smallest height in league: "
        query = "SELECT league.league_ as league, team.team_ as team, name, birthday, nationality,
weight, " \
            "height FROM personal_data JOIN team ON team.id = personal_data.team_id JOIN " \
            "league on league.league_ = " + str("'" + tmp + "'") \
            + " and league.id = team.league_id order by height LIMIT 20"
    query_view(cursor, select=query, table_name=name + str(tmp),
        table_heads=['league', 'team', 'name', 'birthday', 'nationality', 'weight', 'height'])
elif choice == 6:
    tmp = raw_input('\nEnter nationality: ')
    query = "SELECT league.league_ as league, team.team_ as team, name, birthday, nationality FROM " \
        "personal_data JOIN team ON team.id = personal_data.team_id and nationality = " + \
        str("'" + tmp + "'") + " JOIN league on league.id = team.league_id"
    query_view(cursor, select=query, table_name='List of players with nationality: ' + str(tmp),
        table_heads=['league', 'team', 'name', 'birthday', 'nationality'])
elif choice == 7:
    s_o_l = seasons_or_leagues()
    ev_type = event_type()
    if s_o_l == 1:
        tmp = raw_input('\nEnter season: ')
        limit = raw_input('\nEnter limit: ')
        name = "Top in certain season: "
        query = "select season.season_ as season, league.league_ as league, team.team_ as team, " \
            "name, count(event) as goals from personal_data JOIN team on team.id = " \
            "personal_data.team_id JOIN league on league.id = " \
            "team.league_id JOIN match_info ON (personal_data.id = personal_data_id and event = " + \
            str("'" + ev_type + "'") + ") JOIN match on match.id = " \
            "match_info.match_id JOIN season on season.season_ = " + \
            str("'" + tmp + "'") + " and season.id = match.season_id group by season.season_, " \
            "league.league_, team.team_, name order by count(event) DESC LIMIT " + \
            str(limit)
        query_view(cursor, select=query, table_name=name,
            table_heads=['season', 'league', 'team', 'name', 'count'])
    if s_o_l == 2:
        tmp = raw_input('\nEnter league: ')
        limit = raw_input('\nEnter limit: ')

```



```

name = "Top in certain league: "
query = "select league.league_ as league, team.team_ as team, " \
        "name, count(event) as goals from personal_data JOIN team on team.id = " \
        "personal_data.team_id JOIN league on league.league_ = " + \
        str("" + tmp + "") + " and league.id = team.league_id JOIN match_info ON (" \
        "personal_data.id = personal_data_id and " \
        "event = " + str("" + ev_type + "") + ") JOIN match on match.id = " \
        "match_info.match_id group by league.league_, team.team_,
name order by count(event) DESC LIMIT " + str(
    limit)
query_view(cursor, select=query, table_name=name,
            table_heads=['league', 'team', 'name', 'count'])
if s_o_l == 3:
    limit = raw_input("\nEnter limit: ")
    name = "All time top: "
    query = "select league.league_ as league, team.team_ as team, " \
            "name, count(event) as goals from personal_data JOIN team on team.id = " \
            "personal_data.team_id JOIN league on league.id = " \
            "team.league_id JOIN match_info ON (personal_data.id = personal_data_id and event = " + \
            str("" + ev_type + "") + ") JOIN match on match.id = " \
            "match_info.match_id group by league.league_, " \
            "team.team_, name order by count(event) DESC LIMIT " + str(limit)
    query_view(cursor, select=query, table_name=name,
                table_heads=['league', 'team', 'name', 'count'])
if s_o_l == 4:
    tmps = raw_input("\nEnter season: ")
    tmp1 = raw_input("\nEnter league: ")
    limit = raw_input("\nEnter limit: ")
    name = "Your top: "
    query = "select season.season_ as season, league.league_ as league, team.team_ as team, " \
            "name, count(event) as cnt from personal_data JOIN team on team.id = " \
            "personal_data.team_id JOIN league on league.league_ = " + \
            str("" + tmp1 + "") + " and league.id = team.league_id JOIN match_info ON (" \
            "personal_data.id = personal_data_id and event = " + \
            str("" + ev_type + "") + ") JOIN match on match.id = match_info.match_id JOIN season on " \
            "season.season_ = " + str(
            "" + tmps + "") + " and season.id = match.season_id group by season.season_, league.league_, "
    \
            "team.team_, name order by count(event) DESC LIMIT " + str(limit)
    query_view(cursor, select=query, table_name=name,
                table_heads=['season', 'league', 'team', 'name', 'count'])
show_queries_menu()

"1 - Certain season.\n"
"2 - Certain league.\n"
"3 - Certain season and certain league.\n"
"4 - All seasons and all leagues."

def query_view(cursor, select, table_name, table_heads):
    try:
        cursor.execute(select)
        result = cursor.fetchall()
        print("\n" + table_name + "\n")
        print(tabulate(result, headers=table_heads, tablefmt='orgtbl'))
        print("\nThe number of rows in the result set : " + str(len(result)))
        query_export(cursor, select)
    except:
        connection.commit()
        print('Error. Input data is incorrect. Try again.')

```

```

def show_pro_queries_menu():
    print("\nPro queries: \n"
          "1 - Write query.\n"
          "0 - Back")

def choose_pro_query():
    choice = raw_input('\nChoose action: ')
    try:
        if 0 <= int(choice) <= 3:
            return int(choice)
        else:
            print('Error. Your choice was out of range. Try again.\n')
    except ValueError:
        print('Error. You did not type number. Try again.')

def execute_pro_query(cursor):
    show_pro_queries_menu()
    move_back = False
    while not move_back:
        choice = choose_pro_query()
        if choice == 0:
            move_back = True
            break
        elif choice == 1:
            try:
                tmp = raw_input('\nEnter your query: ')
                print("Your query: " + tmp)
                check_select = tmp.split(" ", 1)[0].lower()
                if check_select == "select":
                    query = str(tmp)
                    cursor.execute(query)
                    result = cursor.fetchall()
                    print(tabulate(result))
                    print("The number of rows in the result set : " + str(len(result)))
                    query_export(cursor, query)
                else:
                    print("Error. You are only allowed to use SELECT here. Try again.")
            except:
                connection.commit()
                print('Error. Your query syntax is incorrect. Try again.')
        show_pro_queries_menu()

def export_table_data(cursor, query):
    choice = raw_input('\nDo you want to export this table?[1=yes,0=no]')
    try:
        if int(choice) == 1:
            filename = raw_input('\nType filename: ')
            cursor.execute("COPY(" + query + ") TO
'D:\\Files\\Football_league\\console_app\\export_tables\\" + str(
            filename) + ".csv' CSV HEADER")
        elif int(choice) != 0:
            print('Error. Your choice was out of range. Try again.\n')
    except ValueError:
        print('Error. You did not type number. Try again.')

def show_import_menu():
    print("\nImport: \n"
          "1 - Insert data.\n")

```

```

"2 - Insert new user\n"
"3 - Import xml file\n"
"0 - Back")

def import_data():
    show_import_menu()
    move_back = False
    while not move_back:
        choice = choose_pro_query()
        if choice == 0:
            move_back = True
            break
        elif choice == 1:
            try:
                tmp = raw_input('\nEnter your query: ')
                print("Your query: " + tmp)
                check_insert = tmp.split(" ", 1)[0].lower()
                if check_insert == "insert":
                    cursor.execute(tmp)
                    connection.commit()
                else:
                    print("Error. You are only allowed to use INSERT here. Try again.")
            except:
                connection.commit()
                print('Error. Your query syntax is incorrect. Try again.')
        elif choice == 2:
            try:
                name = raw_input('\nEnter user name: ')
                passw = raw_input('\nEnter user password: ')
                stat = raw_input('\nEnter user status: ')
                query = "INSERT INTO users (login,password,user_status) VALUES (" + str("'" + name + "',"
crypt("") + \
                        str("'" + passw + "',gen_salt('bf'))," + str("'" + stat + "'"))
                cursor.execute(query)
                connection.commit()
            except:
                connection.commit()
                print('Error. Your query syntax is incorrect or this login already exists. Try again.')
        elif choice == 3:
            root = etree.parse("D:\\Files\\Football_league\\console_app\\import.xml")
            for i in root.findall("personal_data"):
                p = [i.find(n).text for n in ("team_id", "name", "nationality", "birthday", "weight", "height")]
                query = ('INSERT INTO personal_data (team_id, name, nationality,birthday, weight,'
                        'height) VALUES (%s, %s, %s,%s,%s,%s)')
                vars = p[0], p[1], p[2], p[3], p[4], p[5]
                try:
                    cursor.execute(query, vars)
                    connection.commit()
                except:
                    connection.commit()
                    print('Error. Data you are trying to import exists in personal_data table.')
            for i in root.findall("match"):
                p = [i.find(n).text for n in ("team1_id", "team2_id", "result", "season_id", "date")]
                query = ('INSERT INTO match (team1_id, team2_id, result,season_id,date)'
                        ' VALUES (%s, %s, %s,%s,%s)')
                vars = p[0], p[1], p[2], p[3], p[4]
                try:
                    cursor.execute(query, vars)
                    connection.commit()
                except:
                    connection.commit()

```

```

        print('Error. Data you are trying to import exists in match table.')
    for i in root.findall("contract"):
        p = [i.find(n).text for n in ("personal_data_id", "fee", "date_from", "date_to")]
        query = ('INSERT INTO contract (personal_data_id, fee, date_from, date_to)'
                 ' VALUES (%s, %s, %s, %s)')
        vars = p[0], p[1], p[2], p[3]
        try:
            cursor.execute(query, vars)
            connection.commit()
        except:
            connection.commit()
            print('Error. Data you are trying to import exists in contract table.')
    for i in root.findall("league"):
        p = i.find("league_").text
        query = ('INSERT INTO league (league_)'
                 ' VALUES (%s)')
        vars = p[0]
        try:
            cursor.execute(query, vars)
            connection.commit()
        except:
            connection.commit()
            print('Error. Data you are trying to import exists in league table.')
    for i in root.findall("match_info"):
        p = [i.find(n).text for n in ("match_id", "personal_data_id", "minute", "event")]
        query = ('INSERT INTO match_info (match_id, personal_data_id, minute, event)'
                 ' VALUES (%s, %s, %s, %s)')
        vars = p[0], p[1], p[2], p[3]
        try:
            cursor.execute(query, vars)
            connection.commit()
        except:
            connection.commit()
            print('Error. Data you are trying to import exists in match_info table.')
    for i in root.findall("person_status"):
        p = [i.find(n).text for n in ("personal_data_id", "match_id", "match_status")]
        query = ('INSERT INTO person_status (personal_data_id, match_id, match_status)'
                 ' VALUES (%s, %s, %s)')
        vars = p[0], p[1], p[2]
        try:
            cursor.execute(query, vars)
            connection.commit()
        except:
            connection.commit()
            print('Error. Data you are trying to import exists in person_status table.')
    for i in root.findall("player"):
        p = [i.find(n).text for n in ("personal_data_id", "position", "season_id")]
        query = ('INSERT INTO player (personal_data_id, position, season_id)'
                 ' VALUES (%s, %s, %s)')
        vars = p[0], p[1], p[2]
        try:
            cursor.execute(query, vars)
            connection.commit()
        except:
            connection.commit()
            print('Error. Data you are trying to import exists in player table.')
    for i in root.findall("season"):
        p = i.find("season_").text
        query = ('INSERT INTO season (season_)'
                 ' VALUES (%s)')
        vars = p[0]
        try:

```

```

        cursor.execute(query, vars)
        connection.commit()
    except:
        connection.commit()
        print('Error. Data you are trying to import exists in season table.')
for i in root.findall("team"):
    p = [i.find(n).text for n in ("league_id", "team_")]
    query = ('INSERT INTO team (league_id,team_)'
            ' VALUES (%s,%s)')
    vars = p[0], p[1]
    try:
        cursor.execute(query, vars)
        connection.commit()
    except:
        connection.commit()
        print('Error. Data you are trying to import exists in team table.')
for i in root.findall("team_position"):
    p = [i.find(n).text for n in ("league_id", "season_id", "team_id", "week", "w_position", "points")]
    query = ('INSERT INTO team_position (league_id, season_id, team_id, week,w_position, points)'
            ' VALUES (%s,%s,%s,%s,%s,%s)')
    vars = p[0], p[1], p[2], p[3], p[4]
    try:
        cursor.execute(query, vars)
        connection.commit()
    except:
        connection.commit()
        print('Error. Data you are trying to import exists in team_position table.')
for i in root.findall("transfer_cost"):
    p = [i.find(n).text for n in ("player_id", "season_start", "season_finish")]
    query = ('INSERT INTO transfer_cost (player_id, season_start, season_finish)'
            ' VALUES (%s,%s,%s)')
    vars = p[0], p[1], p[2]
    try:
        cursor.execute(query, vars)
        connection.commit()
    except:
        connection.commit()
        print('Error. Data you are trying to import exists in transfer_cost table.')
show_import_menu()

def show_update_menu():
    print("\nUpdate: \n"
          "1 - Update data.\n"
          "0 - Back")

def update_data():
    show_update_menu()
    move_back = False
    while not move_back:
        choice = choose_pro_query()
        if choice == 0:
            move_back = True
            break
        elif choice == 1:
            try:
                tmp = raw_input('\nEnter your query: ')
                print("Your query: " + tmp)
                check_update = tmp.split(" ", 1)[0].lower()
                if check_update == "update":
                    cursor.execute(tmp)

```

```

        connection.commit()
    else:
        print("Error. You are only allowed to use UPDATE here. Try again.")
    except:
        connection.commit()
        print("Error. Your query syntax is incorrect. Try again.")
show_update_menu()

def export_table(cursor):
    show_tables_menu()
    move_back = False
    while not move_back:
        choice = choose_table()
        if choice == 0:
            move_back = True
            break
        elif choice == 1:
            filename = raw_input("\nType filename: ")
            cursor.execute(
                "COPY(SELECT * FROM league) TO 'D:\\Files\\Football_league\\console_app\\export_tables\\"
+ str(
            filename) + ".csv' CSV HEADER")
        elif choice == 2:
            filename = raw_input("\nType filename: ")
            cursor.execute(
                "COPY(SELECT * FROM team) TO 'D:\\Files\\Football_league\\console_app\\export_tables\\"
+ str(
            filename) + ".csv' CSV HEADER")
        elif choice == 3:
            filename = raw_input("\nType filename: ")
            cursor.execute(
                "COPY(SELECT * FROM season) TO
'D:\\Files\\Football_league\\console_app\\export_tables\\" + str(
            filename) + ".csv' CSV HEADER")
        elif choice == 4:
            filename = raw_input("\nType filename: ")
            cursor.execute(
                "COPY(SELECT * FROM team_position) TO
'D:\\Files\\Football_league\\console_app\\export_tables\\" + str(
            filename) + ".csv' CSV HEADER")
        elif choice == 5:
            filename = raw_input("\nType filename: ")
            cursor.execute(
                "COPY(SELECT * FROM player) TO 'D:\\Files\\Football_league\\console_app\\export_tables\\"
+ str(
            filename) + ".csv' CSV HEADER")
        elif choice == 6:
            filename = raw_input("\nType filename: ")
            cursor.execute(
                "COPY(SELECT * FROM match) TO 'D:\\Files\\Football_league\\console_app\\export_tables\\"
+ str(
            filename) + ".csv' CSV HEADER")
        elif choice == 7:
            filename = raw_input("\nType filename: ")
            cursor.execute(
                "COPY(SELECT * FROM personal_data) TO
'D:\\Files\\Football_league\\console_app\\export_tables\\" + str(
            filename) + ".csv' CSV HEADER")
        elif choice == 8:
            filename = raw_input("\nType filename: ")
            cursor.execute(

```

```

        "COPY(SELECT * FROM match_info) TO
'D:\\Files\\Football_league\\console_app\\export_tables\\" + str(
    filename) + ".csv' CSV HEADER")
    elif choice == 9:
        filename = raw_input("\nType filename: ")
        cursor.execute(
            "COPY(SELECT * FROM contract) TO
'D:\\Files\\Football_league\\console_app\\export_tables\\" + str(
    filename) + ".csv' CSV HEADER")
    elif choice == 10:
        filename = raw_input("\nType filename: ")
        cursor.execute(
            "COPY(SELECT * FROM transfer_cost) TO
'D:\\Files\\Football_league\\console_app\\export_tables\\" + str(
    filename) + ".csv' CSV HEADER")
    elif choice == 11:
        filename = raw_input("\nType filename: ")
        cursor.execute(
            "COPY(SELECT * FROM person_status) TO
'D:\\Files\\Football_league\\console_app\\export_tables\\" + str(
    filename) + ".csv' CSV HEADER")
    show_tables_menu()

if __name__ == '__main__':
    main()

```