МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ   
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное  
образовательное учреждение высшего образования  
«Самарский национальный исследовательский университет  
имени академика С.П. Королева»  
(Самарский университет)   
  
  
Факультет информатики  
Кафедра программных систем  
  
Дисциплина  
**Логическое программирование  
  
  
  
ОТЧЕТ**по лабораторной работе №4

Тема: Получение информации из реляционной базы данных.  
Вариант №10

Студент: Лапин К.С.   
Группа: 6303-020302D   
  
Преподаватель: Лобанков А.А.  
  
Оценка: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
  
Дата: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Самара 2024

**Теоретические сведения**

В SWI-Prolog, ODBC (Open Database Connectivity) – это стандартный интерфейс доступа к базам данных, который позволяет Prolog-программам взаимодействовать с различными СУБД (системами управления базами данных) через ODBC-драйверы.

Для установления соединения с базой данных с использованием ODBC в SWI-Prolog необходимо выполнить несколько шагов. Прежде всего, убедитесь, что установлен ODBC-драйвер для используемой базы данных.

Затем понадобится создать конфиг ODBC (ODBC Data Source) для базы данных. Это позволит установить соединение с базой данных через ODBC. Далее нужно загрузить библиотеку odbc и использовать предикат odbc\_connect, чтобы установить соединение с базой данных. Пример подключения представлен на рисунке 1.

Изображение выглядит как текст, электроника, снимок экрана, Шрифт

Автоматически созданное описание  
Рисунок 1 – Пример подключения к базе данных PostgreSQL

В приведенном примере swi представляет собой имя ODBC-источника данных, user и password – учетные данные для подключения к базе данных, alias – идентификатор соединения, модификатор open(once) означает, что второй вызов для открытия того же DSN просто возвращает существующее соединение.

odbc\_query – это встроенный предикат в SWI-Prolog, который используется для выполнения SQL-запросов к базе данных через ODBC-соединение. Предикат odbc\_query принимает три аргумента: соединение с базой данных, SQL-запрос в виде строки и переменную для хранения результатов запроса.

odbc\_disconnect – это встроенный предикат в SWI-Prolog, который используется для отключения от базы данных после завершения работы с ней через ODBC-соединение. Предикат odbc\_disconnect принимает один аргумент – соединение с базой данных, которое необходимо закрыть.

В качестве базы данных будет использоваться PostgreSQL.

**Задание**

Задание 1

На основе 3 лабораторной работы создать базу данных о заданной предметной области в виде таблиц. Таблицы должны создаваться, используя соответствующие предикаты.

Задание 2

Разработать набор sql запросов, при помощи которых можно реализовать все запросы, приведенные в варианте задания к лабораторной 3

Задание 3

Используя HTTP server libraries написать (Create/Read/Update/Delete) для всех таблиц базы данных. Предоставить пользователю возможность сбросить базу данных к исходному (тестовому) варианту заполнения.

Задание 4

Реализовать вывод результатов всех поисковых запросов к базе данных на html страницах.

Задание 5

Оформить отчет. Ответить на контрольные вопросы. Предметная область и запросы следующие:

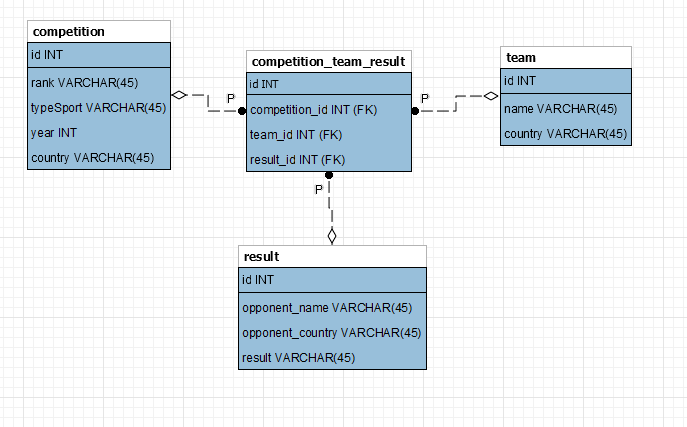
Предметная область – спортивные соревнования. Каждое соревнование может быть описано структурой: ранг соревнований, вид спорта, год проведения, страна проведения, список команд – участников. Команды - участники могут быть описаны следующей структурой: название команды, страна, результаты соревнований.

Результаты соревнований могут быть описаны списком структур: название команды – соперника, страна, тип результата (выигрыш, проигрыш, ничья). Реализовать следующие типы запросов:

1. Найти год, в который участвовало максимальное число команд, в заданном ранге соревнований;
2. Найти вид спорта, в котором выступает заданная команда;
3. Найти все команды, которые участвовали в Олимпийских играх по определенному виду спорта;
4. Найти все команды, участвовавшие в соревнованиях в заданном году;
5. Найти все команды определенной страны, у которых были выигрыши.

**Описание работы программы**

Предметная область реализована в виде таблиц competition, team, result и связующей таблицы competition\_team\_result. Таблицы и связи представлены на рисунке 2.

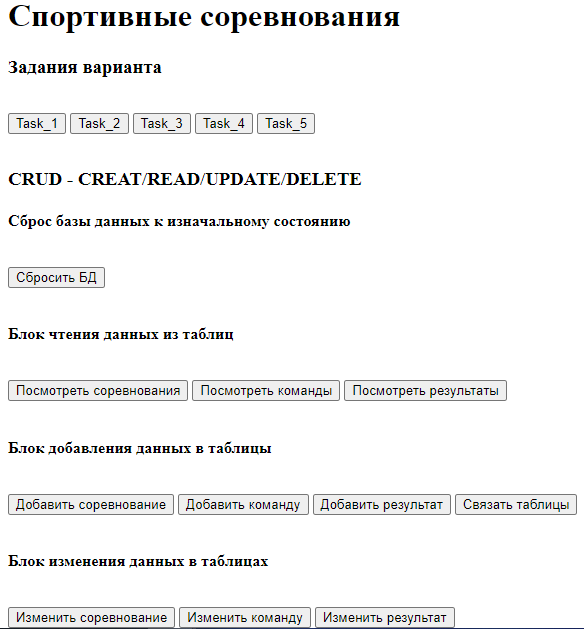
  
Рисунок 2 – Пример таблиц и связей между сущностями

Создание таблиц, добавление, удаление, обновление и поиск записей реализовано с помощью средств PostgreSQL.

* Создание таблиц: CREATE TABLE название\_таблицы (название\_столбца1 тип\_данных атрибуты\_столбца1, название\_столбца2 тип\_данных атрибуты\_столбца2, ... название\_столбцаN тип\_данных атрибуты\_столбцаN, атрибуты\_таблицы);
* добавление записей: INSERT INTO имя\_таблицы (столбец1, столбец2, ... столбецN) VALUES (значение1, значение2, ... значениеN);
* удаление записей: DELETE FROM имя\_таблицы [WHERE условие\_удаления];
* обновление записей: UPDATE имя\_таблицы SET столбец1 = значение1, столбец2 = значение2, ... столбецN = значениеN [WHERE условие\_обновления];
* получение данных: SELECT список\_столбцов FROM имя\_таблицы.

**Примеры работы программы**

На рисунках 3-33 приведены скриншоты работы лабораторной работы.

  
Рисунок 3 – Пример главной html-страницы 1

Изображение выглядит как текст, число, Параллельный, Шрифт

Автоматически созданное описание  
Рисунок 4 – Пример главной html-страницы 2

Изображение выглядит как снимок экрана, текст

Автоматически созданное описание  
Рисунок 5 – Пример страницы запроса для задания 1

Изображение выглядит как текст, снимок экрана, Шрифт, дизайн

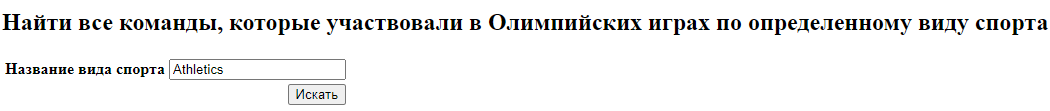
Автоматически созданное описание  
Рисунок 6 – Пример страницы результата запроса для задания 1

Изображение выглядит как текст, снимок экрана, Шрифт, линия

Автоматически созданное описание  
Рисунок 7 – Пример страницы запроса для задания 2

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание  
Рисунок 8 – Пример страницы результата запроса для задания 2

  
Рисунок 9 – Пример страницы запроса для задания 3

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание  
Рисунок 10 – Пример страницы результата запроса для задания 3

Изображение выглядит как снимок экрана, текст

Автоматически созданное описание  
Рисунок 11 – Пример страницы запроса для задания 4

Изображение выглядит как снимок экрана, текст, Шрифт

Автоматически созданное описание  
Рисунок 12 – Пример страницы результата запроса для задания 4

Изображение выглядит как снимок экрана, текст, Шрифт

Автоматически созданное описание  
Рисунок 13 – Пример страницы запроса для задания 5

Изображение выглядит как текст, снимок экрана, Шрифт, линия

Автоматически созданное описание  
Рисунок 14 – Пример страницы результата запроса для задания 5

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание  
Рисунок 15 – Пример страницы просмотра соревнований

Изображение выглядит как текст, чек, снимок экрана, число

Автоматически созданное описание  
Рисунок 16 – Пример страницы просмотра команд

Изображение выглядит как текст, снимок экрана, число, Шрифт

Автоматически созданное описание  
Рисунок 17 – Пример страницы просмотра результатов

Изображение выглядит как текст, снимок экрана, Шрифт, линия

Автоматически созданное описание  
Рисунок 18 – Пример страницы добавления соревнования

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание  
Рисунок 19 – Пример страницы с таблицей соревнований и новой записью

Изображение выглядит как текст, снимок экрана, Шрифт, линия

Автоматически созданное описание  
Рисунок 20 – Пример страницы добавления команды

Изображение выглядит как текст, снимок экрана, число, Шрифт

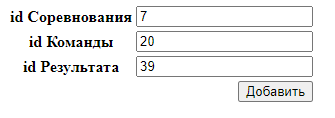
Автоматически созданное описание  
Рисунок 21 – Пример страницы с таблицей команд и новой записью

Изображение выглядит как текст, снимок экрана, Шрифт, линия

Автоматически созданное описание  
Рисунок 22 – Пример страницы добавления результата

Изображение выглядит как текст, снимок экрана, число, Параллельный

Автоматически созданное описание  
Рисунок 23 – Пример страницы с таблицей результатов и новой записью

  
Рисунок 24 – Пример страницы добавления id в связующую таблицу

Изображение выглядит как текст, снимок экрана, число, Шрифт

Автоматически созданное описание  
Рисунок 25 – Пример страницы с таблицей всех данных и новой записью

Изображение выглядит как текст, снимок экрана, дисплей, Шрифт

Автоматически созданное описание  
Рисунок 26 – Пример страницы с изменением данных соревнования по его id

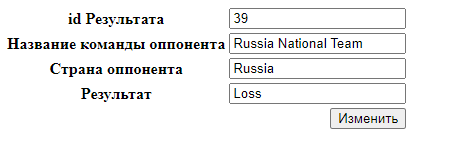
  
Рисунок 27 – Пример страницы с таблицей всех данных и измененной записью

Изображение выглядит как текст, снимок экрана, Шрифт, линия

Автоматически созданное описание  
Рисунок 28 – Пример страницы с изменением данных о команде по ее id

Изображение выглядит как текст, снимок экрана, линия, Параллельный

Автоматически созданное описание  
Рисунок 29 – Пример страницы с таблицей всех данных и измененной записью

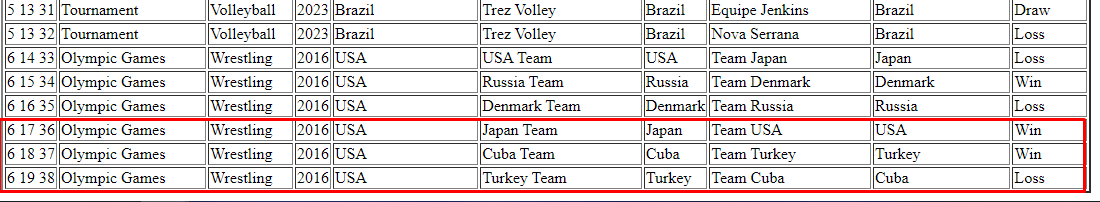
  
Рисунок 30 – Пример страницы с изменением данных о результате по его id

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание  
Рисунок 31 – Пример страницы с таблицей всех данных и измененной записью

Изображение выглядит как текст, снимок экрана, Шрифт, линия

Автоматически созданное описание  
Рисунок 32 – Пример страницы с удалением связи между таблицами по их id

  
Рисунок 33 – Пример страницы с таблицей всех данных, где удалена запись

**Листинг программы**

% Предметная область – спортивные соревнования.

% Каждое соревнование может быть описано структурой: ранг соревнований, вид спорта,

% год проведения, страна проведения, список команд - участников. Команды

% - участники могут быть описаны следующей структурой: название команды,

% страна, результаты соревнований. Результаты соревнований могут быть

% описаны списком структур: название команды – соперника, страна, тип

% результата (выигрыш, проигрыш, ничья). Реализовать следующие типы

% запросов:

% Библиотека для работы с БД

:- use\_module(library(odbc)).

% Библиотеки для работы с http

:-use\_module(library(http/http\_server)).

:-use\_module(library(http/http\_dispatch)).

:-use\_module(library(http/http\_parameters)).

:-use\_module(library(http/html\_write)).

% Покдючение к БД

connect\_db(Connection):-

odbc\_connect(

'swi',

Connection,

[

user(postgres),

password(root),

alias(dsn),

open(once)

]

).

% Задание 1

% На основе 3 лабораторной работы создать базу данных о заданной

% предметной области в виде таблиц. Таблицы должны создаваться используя

% соответствующие предикаты.

%Создание таблиц

create\_tables :-

connect\_db(\_),

odbc\_query(

dsn,

'CREATE TABLE public.competition

(

id SERIAL NOT NULL,

name TEXT NOT NULL,

type\_sport TEXT NOT NULL,

year INTEGER NOT NULL,

country TEXT NOT NULL,

PRIMARY KEY (id)

)'

),

odbc\_query(

dsn,

'CREATE TABLE public.team

(

id SERIAL NOT NULL,

name TEXT NOT NULL,

country TEXT NOT NULL,

PRIMARY KEY (id)

)'

),

odbc\_query(

dsn,

'CREATE TABLE public.result

(

id SERIAL NOT NULL,

opponent\_name TEXT NOT NULL,

country TEXT NOT NULL,

type\_result TEXT NOT NULL,

PRIMARY KEY (id)

)'

),

odbc\_query(

dsn,

'CREATE TABLE public.competition\_team

(

id SERIAL NOT NULL,

competition\_id INTEGER,

teams\_id INTEGER,

result\_id INTEGER,

PRIMARY KEY (id),

FOREIGN KEY (competition\_id) REFERENCES competition (id),

FOREIGN KEY (teams\_id) REFERENCES team (id),

FOREIGN KEY (result\_id) REFERENCES result (id)

)'

),

odbc\_disconnect(dsn).

% Удаление таблиц

delete\_tables:-

connect\_db(\_),

odbc\_query(

dsn,

'DROP TABLE IF EXISTS public.competition, public.team, public.result, public.competition\_team'

),

odbc\_disconnect(dsn).

% Вставка данных в таблицы (значения)

insert\_data\_in\_tables:-

connect\_db(\_),

odbc\_query(

dsn,

'INSERT INTO competition (name, type\_sport, year, country)

VALUES

(\'Olympic Games\', \'Athletics\', 2024, \'France\'),

(\'Championship\', \'Football\', 2020, \'England\'),

(\'Championship\', \'Basketball\', 2021, \'USA\'),

(\'Friendly Match\', \'Football\', 2022, \'Russia\'),

(\'Tournament\', \'Volleyball\', 2023, \'Brazil\'),

(\'Olympic Games\', \'Wrestling\', 2016, \'USA\')'

),

odbc\_query(

dsn,

'INSERT INTO team (name, country)

VALUES

(\'USA National Team\', \'USA\'),

(\'Germany National Team\', \'Germany\'),

(\'Jamaica National Team\', \'Jamaica\'),

(\'England National Team\', \'England\'),

(\'France National Team\', \'France\'),

(\'Spain National Team\', \'Spain\'),

(\'Canada National Team\', \'Canada\'),

(\'Mexico National Team\', \'Mexico\'),

(\'Zenit\', \'Russia\'),

(\'Spartak\', \'Russia\'),

(\'Nova Serrana\', \'Brazil\'),

(\'Ekip Jenkins\', \'Brazil\'),

(\'Trez Volley\', \'Brazil\'),

(\'USA Team\', \'USA\'),

(\'Russia Team\', \'Russia\'),

(\'Denmark Team\', \'Denmark\'),

(\'Japan Team\', \'Japan\'),

(\'Cuba Team\', \'Cuba\'),

(\'Turkey Team\', \'Turkey\')'

),

odbc\_query(

dsn,

'INSERT INTO result (opponent\_name, country, type\_result)

VALUES

(\'Germany National Team\', \'Germany\', \'Win\'),

(\'Jamaica National Team\', \'Jamaica\', \'Win\'),

(\'USA National Team\', \'USA\', \'Loss\'),

(\'Jamaica National Team\', \'Jamaica\', \'Loss\'),

(\'USA National Team\', \'USA\', \'Loss\'),

(\'Germany National Team\', \'Germany\', \'Win\'),

(\'France National Team\', \'France\', \'Win\'),

(\'Spain National Team\', \'Spain\', \'Loss\'),

(\'England National Team\', \'England\', \'Loss\'),

(\'Spain National Team\', \'Spain\', \'Loss\'),

(\'England National Team\', \'England\', \'Win\'),

(\'France National Team\', \'France\', \'Win\'),

(\'Canada National Team\', \'Canada\', \'Win\'),

(\'Mexico National Team\', \'Mexico\', \'Win\'),

(\'France National Team\', \'France\', \'Loss\'),

(\'USA National Team\', \'USA\', \'Loss\'),

(\'Mexico National Team\', \'Mexico\', \'Draw\'),

(\'France National Team\', \'France\', \'Win\'),

(\'USA National Team\', \'USA\', \'Loss\'),

(\'Canada National Team\', \'Canada\', \'Draw\'),

(\'France National Team\', \'France\', \'Win\'),

(\'USA National Team\', \'USA\', \'Win\'),

(\'Canada National Team\', \'Canada\', \'Loss\'),

(\'Mexico National Team\', \'Mexico\', \'Loss\'),

(\'Spartak\', \'Russia\', \'Win\'),

(\'Zenit\', \'Russia\', \'Loss\'),

(\'Equipe Jenkins\', \'Brazil\', \'Loss\'),

(\'Three Volley\', \'Brazil\', \'Win\'),

(\'Nova Serrana\', \'Brazil\', \'Win\'),

(\'Three Volley\', \'Brazil\', \'Draw\'),

(\'Equipe Jenkins\', \'Brazil\', \'Draw\'),

(\'Nova Serrana\', \'Brazil\', \'Loss\'),

(\'Team Japan\', \'Japan\', \'Loss\'),

(\'Team Denmark\', \'Denmark\', \'Win\'),

(\'Team Russia\', \'Russia\', \'Loss\'),

(\'Team USA\', \'USA\', \'Win\'),

(\'Team Turkey\', \'Turkey\', \'Win\'),

(\'Team Cuba\', \'Cuba\', \'Loss\')'

),

odbc\_query(

dsn,

'INSERT INTO competition\_team (competition\_id, teams\_id, result\_id)

VALUES

(1, 1, 1),

(1, 1, 2),

(1, 2, 3),

(1, 2, 4),

(1, 3, 5),

(1, 3, 6),

(2, 4, 7),

(2, 4, 8),

(2, 5, 9),

(2, 5, 10),

(2, 6, 11),

(2, 6, 12),

(3, 1, 13),

(3, 1, 14),

(3, 1, 15),

(3, 7, 16),

(3, 7, 17),

(3, 7, 18),

(3, 8, 19),

(3, 8, 20),

(3, 8, 21),

(3, 5, 22),

(3, 5, 23),

(3, 5, 24),

(4, 9, 25),

(4, 10, 26),

(5, 11, 27),

(5, 11, 28),

(5, 12, 29),

(5, 12, 30),

(5, 13, 31),

(5, 13, 32),

(6, 14, 33),

(6, 15, 34),

(6, 16, 35),

(6, 17, 36),

(6, 18, 37),

(6, 19, 38)'

),

odbc\_disconnect(dsn).

%Вставка данных в таблицу competition

insert\_competition(Name, TypeSport, Year, Country):-

connect\_db(Connection),

odbc\_prepare(

Connection,

"INSERT INTO public.competition (name, type\_sport, year, country) VALUES (?, ?, ?, ?)",

[varchar, varchar, integer, varchar],

Statement

),

odbc\_execute(Statement, [Name, TypeSport, Year, Country]),

%Закрыть prepared statement и connection

close\_conn(Connection, Statement).

%Вставка данных в таблицу team

insert\_team(Name, Country):-

connect\_db(Connection),

odbc\_prepare(

Connection,

'INSERT INTO public.team (name, country) VALUES (?, ?)',

[varchar, varchar],

Statement

),

odbc\_execute(Statement, [Name, Country]),

%Закрыть prepared statement и connection

close\_conn(Connection, Statement).

%Вставка данных в таблицу result

insert\_result(OpponentName, Country, TypeResult):-

connect\_db(Connection),

odbc\_prepare(

Connection,

'INSERT INTO result (opponent\_name, country, type\_result) VALUES (?, ?, ?)',

[varchar, varchar, varchar],

Statement

),

odbc\_execute(Statement, [OpponentName, Country, TypeResult]),

%Закрыть prepared statement и connection

close\_conn(Connection, Statement).

%Вставка данных в таблицу competition\_team

insert\_competition\_team(CompetitionId, TeamId, ResultId):-

connect\_db(Connection),

odbc\_prepare(

Connection,

'INSERT INTO competition\_team (competition\_id, teams\_id, result\_id) VALUES (?, ?, ?)',

[integer, integer, integer],

Statement

),

odbc\_execute(Statement, [CompetitionId, TeamId, ResultId]),

%Закрыть prepared statement и connection

close\_conn(Connection, Statement).

% Задание 2

% Разработать набор sql запросов, при помощи которых можно реализовать

% все запросы, приведенные в варианте задания к лабораторной 3.

% Чтение всех строк с одним полем из prepared statement

% Определение предиката processed\_fetch\_data с аккумулятором для записи

% данных в массив

processed\_fetch\_data(Statement, Data) :-

processed\_fetch\_data\_acc(Statement, [], Data).

% Определение вспомогательного предиката processedFetchDataAcc/3 с аккумулятором для накопления данных

processed\_fetch\_data\_acc(Statement, Acc, Result) :-

odbc\_fetch(Statement, Row, next),

( Row == end\_of\_file -> Result = Acc; % Если достигнут конец файла, возвращаем собранный массив

extract\_data(Row, ExtractedData),

append(Acc, [ExtractedData], NewAcc), % Добавляем данные в массив

processed\_fetch\_data\_acc(Statement, NewAcc, Result)

).

% pattern matchin для строки вида для запросов получения данных

extract\_data(row(Data), Data).

extract\_data(row(Data\_1, Data\_2, Data\_3), [Data\_1, Data\_2, Data\_3]).

extract\_data(row(Data\_1, Data\_2, Data\_3, Data\_4), [Data\_1, Data\_2, Data\_3, Data\_4]).

extract\_data(row(Data\_1, Data\_2, Data\_3, Data\_4, Data\_5), [Data\_1, Data\_2, Data\_3, Data\_4, Data\_5]).

extract\_data(row(Data\_1, Data\_2, Data\_3, Data\_4, Data\_5, Data\_6, Data\_7, Data\_8, Data\_9, Data\_10), [Data\_1, Data\_2, Data\_3, Data\_4, Data\_5, Data\_6, Data\_7, Data\_8, Data\_9, Data\_10]).

% 1 Найти год, в который участвовало максимальное число команд, в

% заданном ранге соревнований;

select\_task\_1(Rank, Data) :-

connect\_db(Connection),

odbc\_prepare(

Connection,

'SELECT max(subquery.year) as year

FROM (

SELECT competition.year as year, COUNT(DISTINCT team.name) AS num\_teams

FROM competition

INNER JOIN competition\_team ON competition.id = competition\_team.competition\_id

INNER JOIN team ON competition\_team.teams\_id = team.id

WHERE competition.name = (?)

GROUP BY year

ORDER BY num\_teams DESC

) AS subquery',

[varchar],

Statement,

[fetch(fetch)]

),

odbc\_execute(Statement, [Rank]),

processed\_fetch\_data(Statement, Data),

%Закрыть prepared statement и connection

close\_conn(Connection, Statement).

% 2 Найти вид спорта, в котором выступает заданная команда;

select\_task\_2(TeamName, Data) :-

connect\_db(Connection),

odbc\_prepare(

Connection,

'SELECT DISTINCT competition.type\_sport

FROM competition

INNER JOIN competition\_team ON competition.id = competition\_team.competition\_id

INNER JOIN team ON competition\_team.teams\_id = team.id

WHERE team.name = (?)',

[varchar],

Statement,

[fetch(fetch)]

),

odbc\_execute(Statement, [TeamName]),

processed\_fetch\_data(Statement, Data),

%Закрыть prepared statement и connection

close\_conn(Connection, Statement).

% 3 Найти все команды, которые участвовали в Олимпийских играх по

% определенному виду спорта;

select\_task\_3(SportName, Data) :-

connect\_db(Connection),

odbc\_prepare(

Connection,

'SELECT DISTINCT team.name

FROM team

INNER JOIN competition\_team ON team.id = competition\_team.teams\_id

INNER JOIN competition ON competition\_team.competition\_id = competition.id

WHERE competition.name = (?) and competition.type\_sport = (?)',

[varchar, varchar],

Statement,

[fetch(fetch)]

),

odbc\_execute(Statement, ['Olympic Games', SportName]),

processed\_fetch\_data(Statement, Data),

%Закрыть prepared statement и connection

close\_conn(Connection, Statement).

% 4 Найти все команды, участвовавшие в соревнованиях в заданном году;

select\_task\_4(Year, Data) :-

connect\_db(Connection),

odbc\_prepare(

Connection,

'SELECT DISTINCT team.name

FROM team

INNER JOIN competition\_team ON competition\_team.teams\_id = team.id

INNER JOIN competition ON competition\_team.competition\_id = competition.id

WHERE competition.year = (?)',

[integer],

Statement,

[fetch(fetch)]

),

odbc\_execute(Statement, [Year]),

processed\_fetch\_data(Statement, Data),

%Закрыть prepared statement и connection

close\_conn(Connection, Statement).

% 5 Найти все команды определенной страны, у которых были выигрыши.

select\_task\_5(Country, Data) :-

connect\_db(Connection),

odbc\_prepare(

Connection,

'SELECT DISTINCT team.name

FROM team

INNER JOIN competition\_team ON competition\_team.teams\_id = team.id

INNER JOIN competition ON competition\_team.competition\_id = competition.id

INNER JOIN result ON competition\_team.result\_id = result.id

WHERE result.type\_result = (?) AND team.country = (?)',

[varchar, varchar],

Statement,

[fetch(fetch)]

),

odbc\_execute(Statement, ['Win', Country]),

processed\_fetch\_data(Statement, Data),

%Закрыть prepared statement и connection

close\_conn(Connection, Statement).

% Закрыть prepared statement и connection

close\_conn(Connection, Statement):-

odbc\_free\_statement(Statement),

odbc\_disconnect(Connection).

% Задание 3

% Используя HTTP server libraries

% написать CRUD (Create/Read/Update/Delete) для всех таблиц базы данных.

% Предоставить пользователю возможность сбросить базу данных к исходному

% (тестовому) варианту заполнения.

% Запуск сервера

server(Port):- http\_server(http\_dispatch, [port(Port)]).

server:-server(8080).

% Остановка сервера

stop(Port):- http\_stop\_server(Port, http\_dispatch).

stop:- stop(8080).

% Хэндлер для ресета БД

:- http\_handler(root(reset\_db), reset\_db, [method(post)]).

% Сброс БД до исходного состояния

reset\_db(\_):-

delete\_tables,

create\_tables,

insert\_data\_in\_tables,

http\_redirect(moved, '/', \_Request).

create\_competition(Name, TypeSport, Year, Country):-

insert\_competition(Name, TypeSport, Year, Country).

create\_teams(Name, Country):-

insert\_team(Name, Country).

create\_results(OpponentName, Country, TypeResult):-

insert\_result(OpponentName, Country, TypeResult).

create\_competition\_team(CompetitionId, TeamId, ResultId):-

insert\_competition\_team(CompetitionId, TeamId, ResultId).

read\_all(Data):-

connect\_db(Connection),

odbc\_prepare(

Connection,

'SELECT competition\_team.competition\_id || \' \' || competition\_team.teams\_id

|| \' \' || competition\_team.result\_id as "id", competition.name as "Ранг",

competition.type\_sport as "Вид спорта", competition.year as "Год проведения",

competition.country as "Страна проведения", team.name as "Команда",

team.country as "Страна команды", result.opponent\_name as "Команда оппонет",

result.country as "Страна оппонента", result.type\_result as "Результат"

FROM competition

INNER JOIN competition\_team ON competition.id = competition\_team.competition\_id

INNER JOIN team ON competition\_team.teams\_id = team.id

INNER JOIN result ON competition\_team.result\_id = result.id',

[],

Statement,

[fetch(fetch)]

),

odbc\_execute(Statement, []),

processed\_fetch\_data(Statement, Data),

%Закрыть prepared statement и connection

close\_conn(Connection, Statement).

read\_competition(Data):-

connect\_db(Connection),

odbc\_prepare(

Connection,

'SELECT competition.id as "id", competition.name as "Ранг",

competition.type\_sport as "Вид спорта", competition.year as "Год проведения",

competition.country as "Страна проведения"

FROM competition',

[],

Statement,

[fetch(fetch)]

),

odbc\_execute(Statement, []),

processed\_fetch\_data(Statement, Data),

%Закрыть prepared statement и connection

close\_conn(Connection, Statement).

read\_teams(Data):-

connect\_db(Connection),

odbc\_prepare(

Connection,

'SELECT team.id as "id", team.name as "Команда", team.country as "Страна команды"

FROM team',

[],

Statement,

[fetch(fetch)]

),

odbc\_execute(Statement, []),

processed\_fetch\_data(Statement, Data),

%Закрыть prepared statement и connection

close\_conn(Connection, Statement).

read\_results(Data):-

connect\_db(Connection),

odbc\_prepare(

Connection,

'SELECT result.id as "id", result.opponent\_name as "Команда оппонет",

result.country as "Страна оппонента", result.type\_result as "Результат"

FROM result',

[],

Statement,

[fetch(fetch)]

),

odbc\_execute(Statement, []),

processed\_fetch\_data(Statement, Data),

%Закрыть prepared statement и connection

close\_conn(Connection, Statement).

read\_competition\_team(Data):-

connect\_db(Connection),

odbc\_prepare(

Connection,

'SELECT competition\_team.competition\_id AS "id Соревнования",

competition\_team.teams\_id AS "id Команды",

competition\_team.result\_id AS "id Результата"

FROM competition\_team',

[],

Statement,

[fetch(fetch)]

),

odbc\_execute(Statement, []),

processed\_fetch\_data(Statement, Data),

%Закрыть prepared statement и connection

close\_conn(Connection, Statement).

update\_competition(Id, Name, TypeSport, Year, Country):-

connect\_db(Connection),

odbc\_prepare(

Connection,

'UPDATE competition

SET name = (?), type\_sport = (?), year = (?), country = (?)

WHERE competition.id = (?)',

[varchar, varchar, integer, varchar, integer],

Statement

),

odbc\_execute(Statement, [Name, TypeSport, Year, Country, Id]),

%Закрыть prepared statement и connection

close\_conn(Connection, Statement).

update\_teams(Id, Name, Country):-

connect\_db(Connection),

odbc\_prepare(

Connection,

'UPDATE team

SET name = (?), country = (?)

WHERE id = (?)',

[varchar, varchar, integer],

Statement

),

odbc\_execute(Statement, [Name, Country, Id]),

%Закрыть prepared statement и connection

close\_conn(Connection, Statement).

update\_results(Id, OpponentName, Country, TypeResult):-

connect\_db(Connection),

odbc\_prepare(

Connection,

'UPDATE result

SET opponent\_name = (?), country = (?), type\_result = (?)

WHERE id = (?)',

[varchar, varchar, varchar, integer],

Statement

),

odbc\_execute(Statement, [OpponentName, Country, TypeResult, Id]),

%Закрыть prepared statement и connection

close\_conn(Connection, Statement).

% Может произайти неизвестное

delete\_competition(Id):-

connect\_db(Connection),

odbc\_prepare(

Connection,

'UPDATE competition\_team

SET competition\_id = NULL

WHERE competition\_id = (?);

DELETE FROM competition

WHERE id = (?)',

[integer, integer],

Statement

),

odbc\_execute(Statement, [Id, Id]),

close\_conn(Connection, Statement).

delete\_teams(Id):-

connect\_db(Connection),

odbc\_prepare(

Connection,

'UPDATE competition\_team

SET teams\_id = NULL

WHERE teams\_id = (?);

DELETE FROM team

WHERE id = (?)

',

[integer, integer],

Statement

),

odbc\_execute(Statement, [Id, Id]),

close\_conn(Connection, Statement).

delete\_results(Id):-

connect\_db(Connection),

odbc\_prepare(

Connection,

'UPDATE competition\_team

SET result\_id = NULL

WHERE result\_id = (?);

DELETE FROM result

WHERE id = (?)

',

[integer, integer],

Statement

),

odbc\_execute(Statement, [Id, Id]),

close\_conn(Connection, Statement).

delete\_competition\_team(CompetitionId, TeamId, ResultId):-

connect\_db(Connection),

odbc\_prepare(

Connection,

'DELETE FROM competition\_team

WHERE id =

(

SELECT id FROM competition\_team

WHERE competition\_team.competition\_id = (?) AND

competition\_team.teams\_id = (?) AND competition\_team.result\_id = (?)

)',

[integer, integer, integer],

Statement

),

odbc\_execute(Statement, [CompetitionId, TeamId, ResultId]),

close\_conn(Connection, Statement).

% Задание 4 Реализовать вывод

% результатов всех поисковых запросов к базе данных на html страницах.

% Хэндлер для корневой страницы

:- http\_handler(root(.), home\_page, []).

% Базовый случай: преобразование пустого массива Data. Data - массив

% массивов

generate\_rows([], []).

% Рекурсивный случай: преобразование массива Data в массив Row

generate\_rows([Data | Tail], [Row | Result]) :-

generate\_row(Data, Row), % Переводим массив Data в строку Row

generate\_rows(Tail, Result). % Рекурсивный вызов для оставшихся элементов

% Предикат для преобразования массива Data в строку Row

generate\_row(Data, Row) :-

maplist(construct\_td, Data, TdStrings), % Преобразование каждого элемента в тег td

Row = tr(TdStrings).

% Предикат для создания тега td с данными

construct\_td(Data, td(Data)).

% Добавление элемента в начало списка

ins(L, El, [El|L]).

% Обработка главной страницы

home\_page(\_Request):-

read\_all(Data),

generate\_rows(Data, Row),

ins(Row, tr(

[

th('id'),

th('Ранг Соревнования'),

th('Вид спорта'),

th('Год'),

th('Страна проведения'),

th('Команда'),

th('Страна'),

th('Оппонент'),

th('Страна оппонента'),

th('Результат')

]

), Rows\_with\_Headers),

reply\_html\_page(

title('Таблица спортивных соревнований'),

[

h1('Спортивные соревнования'),

h3('Задания варианта'),

form(

[style('display: inline-block')],

p(button([type(submit), formaction('/max\_teams\_year\_find')], 'Task\_1'))

),

form(

[style('display: inline-block')],

p(button([type(submit), formaction('/sport\_by\_team\_find')], 'Task\_2'))

),

form(

[style('display: inline-block')],

p(button([type(submit), formaction('/olympc\_teams\_by\_sport\_find')], 'Task\_3'))

),

form(

[style('display: inline-block')],

p(button([type(submit), formaction('/teams\_in\_year\_find')], 'Task\_4'))

),

form(

[style('display: inline-block')],

p(button([type(submit), formaction('/winning\_teams\_by\_country\_find')], 'Task\_5'))

),

br(''),

h3('CRUD - CREAT/READ/UPDATE/DELETE'),

h4('Сброс базы данных к изначальному состоянию'),

form(

[style('display: inline-block'), method(post)],

p(button([type(submit), formaction(location\_by\_id('reset\_db'))], 'Сбросить БД'))

),

br(''),

h4('Блок чтения данных из таблиц'),

form(

[style('display: inline-block')],

p(button([type(submit), formaction('/read\_competitions')], 'Посмотреть соревнования'))

),

form(

[style('display: inline-block')],

p(button([type(submit), formaction('/read\_teams')], 'Посмотреть команды'))

),

form(

[style('display: inline-block')],

p(button([type(submit), formaction('/read\_results')], 'Посмотреть результаты'))

),

br(''),

h4('Блок добавления данных в таблицы'),

form(

[style('display: inline-block')],

p(button([type(submit), formaction('/add\_competition\_fill')], 'Добавить соревнование'))

),

form(

[style('display: inline-block')],

p(button([type(submit), formaction('/add\_team\_fill')], 'Добавить команду'))

),

form(

[style('display: inline-block')],

p(button([type(submit), formaction('/add\_result\_fill')], 'Добавить результат'))

),

form(

[style('display: inline-block')],

p(button([type(submit), formaction('/add\_competition\_teams\_fill')], 'Связать таблицы'))

),

br(''),

h4('Блок изменения данных в таблицах'),

form(

[style('display: inline-block')],

p(button([type(submit), formaction('/update\_competition\_find')], 'Изменить соревнование'))

),

form(

[style('display: inline-block')],

p(button([type(submit), formaction('/update\_team\_find')], 'Изменить команду'))

),

form(

[style('display: inline-block')],

p(button([type(submit), formaction('/update\_result\_find')], 'Изменить результат'))

),

br(''),

h4('Блок удаления данных в таблицах'),

form(

[style('display: inline-block')],

p(button([type(submit), formaction('/delete\_competition\_find')], 'Удалить соревнование'))

),

form(

[style('display: inline-block')],

p(button([type(submit), formaction('/delete\_team\_find')], 'Удалить команду'))

),

form(

[style('display: inline-block')],

p(button([type(submit), formaction('/delete\_result\_find')], 'Удалить результат'))

),

form(

[style('display: inline-block')],

p(button([type(submit), formaction('/delete\_full\_data\_find')], 'Удалить связь'))

),

h3('Таблица'),

table(

[border(2)],

Rows\_with\_Headers

)

]).

% Для Task\_1

% Хэндлер странички для ввода данных для поиска года, с максимальным

% числом команд

:- http\_handler(root(max\_teams\_year\_find), max\_teams\_year\_find\_page, []).

% Хэндлер поиска года, в который учавствовало максимальное число комад

:- http\_handler(root(max\_teams\_year), max\_teams\_year\_page, [method(post)]).

max\_teams\_year\_find\_page(\_Request):-

reply\_html\_page(

title('Ввод ранга соревнований'),

[form(

[action=location\_by\_id(max\_teams\_year\_page), method(post)],

[

h2('Найти год, в который участвовало максимальное число команд, в заданном ранге соревнований'),

table([

tr([th('Ранг'), td(input([name(rank)]))]),

tr(td([colspan(2), align(right)], input([type=submit, value='Искать'])))

] )

]

)]

).

% Data - массив простых данных, не содержит подмассивов, каждый элемент

% - новая строка

convert\_to\_row([], []).

convert\_to\_row([Data\_head | Data\_tail], [tr(td(Data\_head)) | Row\_tail]) :-

convert\_to\_row(Data\_tail, Row\_tail).

max\_teams\_year\_page(Request):-

http\_parameters(

Request,

[

rank(Rank, [])

]

),

select\_task\_1(Rank, Year),

convert\_to\_row(Year, YearRow),

ins(YearRow, tr(

[

th('Год')

]

), Rows\_with\_Headers),

reply\_html\_page(

title('Год, с максимальным числом команд'),

[

h1('Год, с максимальным числом команд'),

table(

[border(2)],

Rows\_with\_Headers

)

]

).

% Для Task 2

% Хэндлер странички для ввода данных для поиска вида спорта, в котором

% выступает заданная команда

:- http\_handler(root(sport\_by\_team\_find), sport\_by\_team\_find\_page, []).

% Хэндлер поиска вида спорта, в котором выступает заданная команда

:- http\_handler(root(sport\_by\_team), sport\_by\_team\_page, [method(post)]).

sport\_by\_team\_find\_page(\_Request):-

reply\_html\_page(

title('Ввод названия команды'),

[form(

[action=location\_by\_id(sport\_by\_team\_page), method(post)],

[

h2('Найти вид спорта, в котором выступает заданная команда'),

table([

tr([th('Название команды'), td(input([name(teamName)]))]),

tr(td([colspan(2), align(right)], input([type=submit, value='Искать'])))

] )

]

)]

).

sport\_by\_team\_page(Request):-

http\_parameters(

Request,

[

teamName(TeamName, [])

]

),

select\_task\_2(TeamName, TypeSport),

convert\_to\_row(TypeSport, SportRow),

ins(SportRow, tr(

[

th('Вид спорта')

]

), Rows\_with\_Headers),

reply\_html\_page(

title('Вид спорта, в котором выступает заданная команда'),

[

h1('Вид спорта, в котором выступает заданная команда'),

table(

[border(2)],

Rows\_with\_Headers

)

]

).

% Для Task 3

% Хэндлер странички для ввода данных для поиска всех команд, которые

% учавствовали в Олимпийских играх в определенном виде спорта

:- http\_handler(root(olympc\_teams\_by\_sport\_find), olympc\_teams\_by\_sport\_find\_page, []).

% Хэндлер поиска всех команд, которые учавствовали в Олимпийких играх в

% определенном виде спорта

:- http\_handler(root(olympc\_teams\_by\_sport), olympc\_teams\_by\_sport\_page, [method(post)]).

olympc\_teams\_by\_sport\_find\_page(\_Request):-

reply\_html\_page(

title('Ввод названия спорта'),

[form(

[action=location\_by\_id(olympc\_teams\_by\_sport\_page), method(post)],

[

h2('Найти все команды, которые участвовали в Олимпийских играх по определенному виду спорта'),

table([

tr([th('Название вида спорта'), td(input([name(sport)]))]),

tr(td([colspan(2), align(right)], input([type=submit, value='Искать'])))

] )

]

)]

).

olympc\_teams\_by\_sport\_page(Request):-

http\_parameters(

Request,

[

sport(Sport, [])

]

),

select\_task\_3(Sport, Teams),

convert\_to\_row(Teams, TeamsRow),

ins(TeamsRow, tr(

[

th('Название команды')

]

), Rows\_with\_Headers),

reply\_html\_page(

title('Команды, участвующие в Олимпийских играх в определенном виде спорта'),

[

h1('Команды, которые участвовали в Олимпийских играх в определенном виде спорта'),

table(

[border(2)],

Rows\_with\_Headers

)

]

).

% Для Task 4

% Хэндлер странички для ввода данных для поиска всех команд, которые

% учавствовали в заданном году

:- http\_handler(root(teams\_in\_year\_find), teams\_in\_year\_find\_page, []).

% Хэндлер поиска всех команд, учавствоваших в соревнованиях в заданном

% году

:- http\_handler(root(teams\_in\_year), teams\_in\_year\_page, [method(post)]).

teams\_in\_year\_find\_page(\_Request):-

reply\_html\_page(

title('Ввод года'),

[form(

[action=location\_by\_id(teams\_in\_year\_page), method(post)],

[

h2('Найти все команды, участвовавшие в соревнованиях в заданном году'),

table([

tr([th('Год'), td(input([name(year), type(number)]))]),

tr(td([colspan(2), align(right)], input([type=submit, value='Искать'])))

] )

]

)]

).

teams\_in\_year\_page(Request):-

http\_parameters(

Request,

[

year(Year, [])

]

),

atom\_number(Year, Number),

select\_task\_4(Number, Teams),

convert\_to\_row(Teams, TeamsRow),

ins(TeamsRow, tr(

[

th('Название команды')

]

), Rows\_with\_Headers),

reply\_html\_page(

title('Команды, участвовавшие в соревнованиях в заданном году'),

[

h1('Команды, участвовавшие в соревнованиях в заданном году'),

table(

[border(2)],

Rows\_with\_Headers

)

]

).

% Для Task 5

% Хэндлер странички для ввода данных для поиска всех команд определенной

% страны, у которых были выигрыши

:- http\_handler(root(winning\_teams\_by\_country\_find), winning\_teams\_by\_country\_find\_page, []).

% Хэндлер поиска всех команд определенной страны, у которых были

% выигрыши

:- http\_handler(root(winning\_teams\_by\_country), winning\_teams\_by\_country\_page, [method(post)]).

winning\_teams\_by\_country\_find\_page(\_Request):-

reply\_html\_page(

title('Ввод страны'),

[form(

[action=location\_by\_id(winning\_teams\_by\_country\_page), method(post)],

[

h2('Найти все команды определенной страны, у которых были выигрыши'),

table([

tr([th('Страна'), td(input([name(country)]))]),

tr(td([colspan(2), align(right)], input([type=submit, value='Искать'])))

] )

]

)]

).

winning\_teams\_by\_country\_page(Request):-

http\_parameters(

Request,

[

country(Country, [])

]

),

select\_task\_5(Country, Teams),

convert\_to\_row(Teams, TeamsRow),

ins(TeamsRow, tr(

[

th('Название команды')

]

), Rows\_with\_Headers),

reply\_html\_page(

title('Команды определенной страны, у которых были выигрыши'),

[

h1('Команды определенной страны, у которых были выигрыши'),

table(

[border(2)],

Rows\_with\_Headers

)

]

).

% Реализация страничек получения таблиц

% Хэдлер для странички с информацией о спортивных соревнованиях

:- http\_handler(root(read\_competitions), read\_competition\_page, []).

read\_competition\_page(\_Request):-

% Добавляем заголовки таблицы

read\_competition(Data),

generate\_rows(Data, Row),

ins(Row, tr(

[

th('id'),

th('Ранг Соревнования'),

th('Вид спорта'),

th('Год'),

th('Страна проведения') ]

), Rows\_with\_Headers),

reply\_html\_page(

title('Таблица спортивных соревнований'),

[

h1('Спортивные соревнования'),

table(

[border(2)],

Rows\_with\_Headers

)

]).

% Хэдлер для странички с информацией о спортивных командах

:- http\_handler(root(read\_teams), read\_teams\_page, []).

read\_teams\_page(\_Request):-

% Добавляем заголовки таблицы

read\_teams(Data),

generate\_rows(Data, Row),

ins(Row, tr(

[

th('id'),

th('Название команды'),

th('Страна команды') ]

), Rows\_with\_Headers),

reply\_html\_page(

title('Таблица спортивных команд'),

[

h1('Команды'),

table(

[border(2)],

Rows\_with\_Headers

)

]).

% Хэдлер для странички с информацией о результатах соревнований

:- http\_handler(root(read\_results), read\_results\_page, []).

read\_results\_page(\_Request):-

% Добавляем заголовки таблицы

read\_results(Data),

generate\_rows(Data, Row),

ins(Row, tr(

[

th('id'),

th('Название команды оппонента'),

th('Страна команды оппонента'),

th('Результат')

]

), Rows\_with\_Headers),

reply\_html\_page(

title('Таблица результатов соревнований'),

[

h1('Результаты'),

table(

[border(2)],

Rows\_with\_Headers

)

]).

% Реализация страничек для добавления данных в таблицы

% Хэндлер страницы для добавления нового соревнования

:- http\_handler(root(add\_competition\_fill), add\_competition\_fill\_page, []).

% Хэндлер для добавления нового соревнования

:- http\_handler(root(add\_competition), add\_competition, [method(post)]).

% Соревнования

add\_competition\_fill\_page(\_Request):-

reply\_html\_page(

title('Добавление соревнования'),

[form(

[action=location\_by\_id(add\_competition), method(post)],

[

table([

tr([th('Ранг'), td(input([name(rank)]))]),

tr([th('Вид спорта'), td(input([name(sport)]))]),

tr([th('Год'), td(input([name(year), type(number)]))]),

tr([th('Страна'), td(input([name(country)]))]),

tr(td([colspan(2), align(right)], input([type=submit, value='Добавить'])))

] )

]

)]

).

add\_competition(Request):-

http\_parameters(

Request,

[

rank(Rank, []),

sport(Sport, []),

year(Year, []),

country(Country, [])

]

),

atom\_number(Year, Number),

create\_competition(Rank, Sport, Number, Country),

http\_redirect(moved, '/', \_Request).

% Хэндлер страницы для добавления новой команды

:- http\_handler(root(add\_team\_fill), add\_team\_fill\_page, []).

% Хэндлер для добавления нового соревнования

:- http\_handler(root(add\_team), add\_team, [method(post)]).

% Команды

add\_team\_fill\_page(\_Request):-

reply\_html\_page(

title('Добавление команды'),

[form(

[action=location\_by\_id(add\_team), method(post)],

[

table([

tr([th('Название команды'), td(input([name(team)]))]),

tr([th('Страна команды'), td(input([name(country)]))]),

tr(td([colspan(2), align(right)], input([type=submit, value='Добавить'])))

] )

]

)]

).

add\_team(Request):-

http\_parameters(

Request,

[

team(TeamName, []),

country(Country, [])

]

),

create\_teams(TeamName, Country),

http\_redirect(moved, '/', \_Request).

% Хэндлер страницы для добавления результата соревнований

:- http\_handler(root(add\_result\_fill), add\_result\_fill\_page, []).

% Хэндлер для добавления нового соревнования

:- http\_handler(root(add\_result), add\_result, [method(post)]).

% Результаты

add\_result\_fill\_page(\_Request):-

reply\_html\_page(

title('Добавление результата'),

[form(

[action=location\_by\_id(add\_result), method(post)],

[

table([

tr([th('Название команды оппонента'), td(input([name(team)]))]),

tr([th('Страна команды оппонента'), td(input([name(country)]))]),

tr([th('Результат'), td(input([name(result)]))]),

tr(td([colspan(2), align(right)], input([type=submit, value='Добавить'])))

] )

]

)]

).

add\_result(Request):-

http\_parameters(

Request,

[

team(TeamName, []),

country(Country, []),

result(Result, [])

]

),

create\_results(TeamName, Country, Result),

http\_redirect(moved, '/', \_Request).

% Хэндлер страницы для добавления id-шников для связующей таблицы

:- http\_handler(root(add\_competition\_teams\_fill), add\_competition\_teams\_fill\_page, []).

% Хэндлер для добавления нового соревнования

:- http\_handler(root(add\_competition\_teams), add\_competition\_teams, [method(post)]).

% Связующая таблица

add\_competition\_teams\_fill\_page(\_Request):-

reply\_html\_page(

title('Добавление id-шников для связующей таблицы'),

[form(

[action=location\_by\_id(add\_competition\_teams), method(post)],

[

table([

tr([th('id Соревнования'), td(input([name(competition), type(number)]))]),

tr([th('id Команды'), td(input([name(team), type(number)]))]),

tr([th('id Результата'), td(input([name(result), type(number)]))]),

tr(td([colspan(2), align(right)], input([type=submit, value='Добавить'])))

] )

]

)]

).

add\_competition\_teams(Request):-

http\_parameters(

Request,

[

competition(CompetitionId, []),

team(TeamId, []),

result(ResultId, [])

]

),

atom\_number(CompetitionId, NCompetitionId),

atom\_number(TeamId, NTeamId),

atom\_number(ResultId, NResultId),

create\_competition\_team(NCompetitionId, NTeamId, NResultId),

http\_redirect(moved, '/', \_Request).

% Реализация страничек для изменения данных в таблицах

% Хэндлер страницы для обновления данных в таблице соревнований

:- http\_handler(root(update\_competition\_find), update\_competition\_find\_page, []).

% Хэндлер для обновления соревнования

:- http\_handler(root(update\_competition), update\_competition, [method(post)]).

update\_competition\_find\_page(\_Request):-

reply\_html\_page(

title('Изменение данных в таблице соревнований'),

[form(

[action=location\_by\_id(update\_competition), method(post)],

[

table([

tr([th('id Соревнования'), td(input([name(id), type(number)]))]),

tr([th('Ранг соревнования'), td(input([name(rank)]))]),

tr([th('Вид спорта'), td(input([name(sport)]))]),

tr([th('Год проведения'), td(input([name(year), type(number)]))]),

tr([th('Страна проведения'), td(input([name(country)]))]),

tr(td([colspan(2), align(right)], input([type=submit, value='Изменить'])))

] )

]

)]

).

update\_competition(Request):-

http\_parameters(

Request,

[

id(CompetitionId, []),

rank(Rank, []),

sport(TypeSport, []),

year(Year, []),

country(Country, [])

]

),

atom\_number(CompetitionId, NCompetitionId),

atom\_number(Year, NYear),

update\_competition(NCompetitionId, Rank, TypeSport, NYear, Country),

http\_redirect(moved, '/', \_Request).

% Хэндлер страницы для обновления команды

:- http\_handler(root(update\_team\_find), update\_team\_find\_page, []).

% Хэндлер для обновления команды

:- http\_handler(root(update\_team), update\_team, [method(post)]).

update\_team\_find\_page(\_Request):-

reply\_html\_page(

title('Изменение данных в таблице команд'),

[form(

[action=location\_by\_id(update\_team), method(post)],

[

table([

tr([th('id Команды'), td(input([name(id), type(number)]))]),

tr([th('Название команды'), td(input([name(name)]))]),

tr([th('Страна команды'), td(input([name(country)]))]),

tr(td([colspan(2), align(right)], input([type=submit, value='Изменить'])))

] )

]

)]

).

update\_team(Request):-

http\_parameters(

Request,

[

id(TeamId, []),

name(TeamName, []),

country(Country, [])

]

),

atom\_number(TeamId, NTeamId),

update\_teams(NTeamId, TeamName, Country),

http\_redirect(moved, '/', \_Request).

% Хэндлер страницы для обновления результата

:- http\_handler(root(update\_result\_find), update\_result\_find\_page, []).

% Хэндлер для обновления результата

:- http\_handler(root(update\_result), update\_result, [method(post)]).

update\_result\_find\_page(\_Request):-

reply\_html\_page(

title('Изменение данных в таблице результатов'),

[form(

[action=location\_by\_id(update\_result), method(post)],

[

table([

tr([th('id Результата'), td(input([name(id), type(number)]))]),

tr([th('Название команды оппонента'), td(input([name(name)]))]),

tr([th('Страна оппонента'), td(input([name(country)]))]),

tr([th('Результат'), td(input([name(result)]))]),

tr(td([colspan(2), align(right)], input([type=submit, value='Изменить'])))

] )

]

)]

).

update\_result(Request):-

http\_parameters(

Request,

[

id(ResultId, []),

name(OpponentName, []),

country(OpponentCountry, []),

result(Result, [])

]

),

atom\_number(ResultId, NResultId),

update\_results(NResultId, OpponentName, OpponentCountry, Result),

http\_redirect(moved, '/', \_Request).

% Реализация страничек для удаления данных в таблицах

% Хэндлер страницы для полного удаления связанных данных

:- http\_handler(root(delete\_full\_data\_find), delete\_full\_date\_find\_page, []).

% Хэндлер для полного удаления связанных данных

:- http\_handler(root(delete\_full\_data), delete\_full\_data, [method(post)]).

delete\_full\_date\_find\_page(\_Request):-

reply\_html\_page(

title('Получение id-шников для связующей таблицы'),

[form(

[action=location\_by\_id(delete\_full\_data), method(post)],

[

table([

tr([th('id Соревнования'), td(input([name(competition), type(number)]))]),

tr([th('id Команды'), td(input([name(team), type(number)]))]),

tr([th('id Результата'), td(input([name(result), type(number)]))]),

tr(td([colspan(2), align(right)], input([type=submit, value='Удалить'])))

] )

]

)]

).

delete\_full\_data(Request):-

http\_parameters(

Request,

[

competition(CompetitionId, []),

team(TeamId, []),

result(ResultId, [])

]

),

atom\_number(CompetitionId, NCompetitionId),

atom\_number(TeamId, NTeamId),

atom\_number(ResultId, NResultId),

delete\_competition\_team(NCompetitionId, NTeamId, NResultId),

http\_redirect(moved, '/', \_Request).

% Хэндлер страницы для удаления данных из соревнований

:- http\_handler(root(delete\_competition\_find), delete\_competition\_find\_page, []).

% Хэндлер для удаления данных из соревнований

:- http\_handler(root(delete\_competition), delete\_competition\_method, [method(post)]).

delete\_competition\_find\_page(\_Request):-

reply\_html\_page(

title('Удаление данных в таблице соревнований'),

[form(

[action=location\_by\_id(delete\_competition\_method), method(post)],

[

table([

tr([th('id Соревнования'), td(input([name(id), type(number)]))]),

tr(td([colspan(2), align(right)], input([type=submit, value='Удалить'])))

] )

]

)]

).

delete\_competition\_method(Request):-

http\_parameters(

Request,

[

id(CompetitionId, [])

]

),

atom\_number(CompetitionId, NCompetitionId),

delete\_competition(NCompetitionId),

http\_redirect(moved, '/', \_Request).

% Хэндлер страницы для удаления данных из команд

:- http\_handler(root(delete\_team\_find), delete\_team\_find\_page, []).

% Хэндлер для удаления команд

:- http\_handler(root(delete\_team), delete\_team\_method, [method(post)]).

delete\_team\_find\_page(\_Request):-

reply\_html\_page(

title('Удаление данных из таблицы команд'),

[form(

[action=location\_by\_id(delete\_team\_method), method(post)],

[

table([

tr([th('id Команды'), td(input([name(id), type(number)]))]),

tr(td([colspan(2), align(right)], input([type=submit, value='Удалить'])))

] )

]

)]

).

delete\_team\_method(Request):-

http\_parameters(

Request,

[

id(TeamId, [])

]

),

atom\_number(TeamId, NTeamId),

delete\_teams(NTeamId),

http\_redirect(moved, '/', \_Request).

% Хэндлер страницы для удаления данных из таблицы результатов

:- http\_handler(root(delete\_result\_find), delete\_result\_find\_page, []).

% Хэндлер для удаления результатов

:- http\_handler(root(delete\_result), delete\_result\_method, [method(post)]).

delete\_result\_find\_page(\_Request):-

reply\_html\_page(

title('Удаление данных из таблицы результатов'),

[form(

[action=location\_by\_id(delete\_result\_method), method(post)],

[

table([

tr([th('id Результата'), td(input([name(id), type(number)]))]),

tr(td([colspan(2), align(right)], input([type=submit, value='Удалить'])))

] )

]

)]

).

delete\_result\_method(Request):-

http\_parameters(

Request,

[

id(ResultId, []) ]

),

atom\_number(ResultId, NResultId),

delete\_results(NResultId),

http\_redirect(moved, '/', \_Request).