МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ   
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное  
образовательное учреждение высшего образования  
«Самарский национальный исследовательский университет  
имени академика С.П. Королева»  
(Самарский университет)   
  
  
Институт информатики и кибернетики  
Кафедра программных систем  
  
Дисциплина  
**Логическое программирование  
  
  
  
ОТЧЕТ**по лабораторной работе №4  
Вариант №12

Студент: Мананников М.А.,   
Группа: 6303-020302D  
  
Преподаватель: Лобанков А.А.  
  
Оценка: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
  
Дата: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Самара 2024

**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 (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 страницы.

Задание 5

Оформить отчет. Ответить на контрольные вопросы.

**Предметная область**

Предметная область – библиотека. Каждый автор может быть описан структурой из трех компонент: имени, фамилии и годом рождения. Каждая книга может быть описана структурой: названием, автором, издательством, числом экземпляров. Каждое издательство может быть описана структурой: названием, номером, годом издания, количеством страниц. Реализовать следующие типы запросов:

* Найти автора, у которого книга имеет самый ранний год издания;
* Подсчитать число изданий заданной книги;
* Найти всех книги, изданные в заданном издательстве за последние десять лет;
* Найти все книги заданного автора;
* Найти все книги цена, которых превышает заданную сумму.

**Примеры работы программы**

На рисунках 1-14 приведены экранные снимки работы программы.

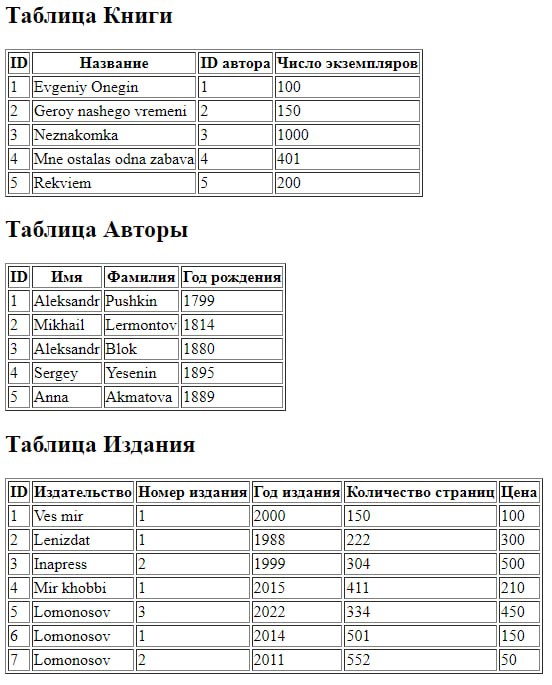
****

Рисунок 1 ­– Пример таблиц базы знаний ­­­

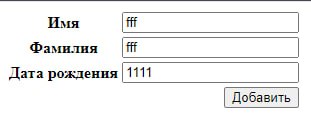


Рисунок 2 – Пример добавления родителя



Рисунок 3 – Пример таблицы “Авторы” после добавления нового автора

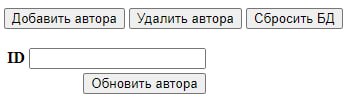


Рисунок 4 – Функционал для удаления, изменения и добавления элементов сущности “Авторы”

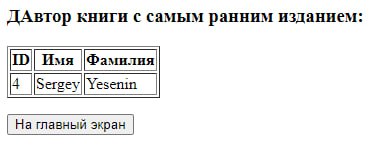


Рисунок 5 – Результат первого запроса



Рисунок 7 – Форма для выполнения второго запроса

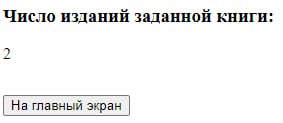


Рисунок 8 – Результат второго запроса

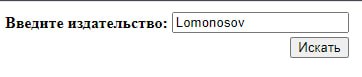


Рисунок 9 – Форма для выполнения третьего запроса

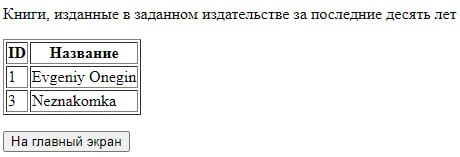


Рисунок 10 – Результат третьего запроса

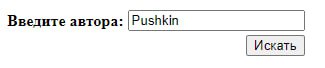


Рисунок 11 – Форма для выполнения четвертого запроса

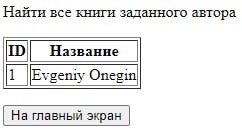


Рисунок 12 – Результат четвертого запроса

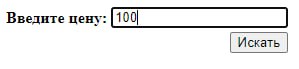


Рисунок 13 – Форма для выполнения пятого запроса

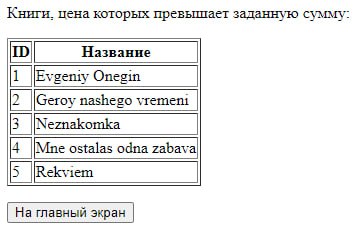


Рисунок 14 – Результат выполнения пятого запроса

**Листинг программы**

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

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

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

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

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

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

:- use\_module(library(http/thread\_httpd)).

%библиотека для полного стектрейса ошибок

% Обработчик для корневого URL

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

%хэндлер страницы для добавления нового требования

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

%хэндлер для добавления нового требования

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

%хэндлер страницы для удаления требования

:- http\_handler(root(delete\_requirements\_p), delete\_requirements\_p, []).

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

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

%хэндлер для удаления требования

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

%хэндлер для ресета БД

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

% Определение хэндлеров для различных запросов

:- http\_handler(root(task\_1), task\_1\_page, []).

:- http\_handler(root(task\_2), task\_2\_page, []).

:- http\_handler(root(task\_2\_V), task\_2\_page\_V, []).

:- http\_handler(root(task\_3), task\_3\_page, []).

:- http\_handler(root(task\_3\_V), task\_3\_page\_V, []).

:- http\_handler(root(task\_4), task\_4\_page, []).

:- http\_handler(root(task\_4\_V), task\_4\_page\_V, []).

:- http\_handler(root(task\_5), task\_5\_page, []).

:- http\_handler(root(task\_5\_S), task\_5\_page\_S, []).

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

% Запуск и остановка сервера

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

server:-server(8880).

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

stop:- stop(8880).

delete\_all\_authors :-

connect\_db(Connection),

odbc\_query(Connection, 'DELETE FROM Авторы', \_),

odbc\_disconnect(Connection).

delete\_all\_izd :-

connect\_db(Connection),

odbc\_query(Connection, 'DELETE FROM Издания', \_),

odbc\_disconnect(Connection).

delete\_all\_book :-

connect\_db(Connection),

odbc\_query(Connection, 'DELETE FROM Книги', \_),

odbc\_disconnect(Connection).

delete\_all\_book\_izd :-

connect\_db(Connection),

odbc\_query(Connection, 'DELETE FROM Книги\_Издания', \_),

odbc\_disconnect(Connection).

drop\_tables :-

connect\_db(Connection),

odbc\_query(Connection, 'DROP TABLE IF EXISTS Книги\_Издания', \_),

odbc\_query(Connection, 'DROP TABLE IF EXISTS Книги', \_),

odbc\_query(Connection, 'DROP TABLE IF EXISTS Издания', \_),

odbc\_query(Connection, 'DROP TABLE IF EXISTS Авторы', \_),

odbc\_disconnect(Connection).

reset\_DB(\_):-

drop\_tables,

create\_table\_izd,

create\_table\_author,

create\_table\_book,

create\_table\_book\_has\_izd,

%инициализация БД

%Автор может быть описан структурой: имя, фамилия, год рождения.

add\_author,

%Издание может быть описано структурой: издательство, номер издания, год издания, количество страниц, цена.

add\_izd,

%Каждая книга может быть описана структурой: название, автор, список изданий, число экземпляров.

add\_book,

add\_book\_izd,

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

book\_row([ID, Title, AuthorID, CopyCount]) :-

odbc\_query(dsn, 'SELECT \* FROM Книги', row(ID, Title, AuthorID, CopyCount)).

write\_book\_row([ID, Title, AuthorID, CopyCount]) :-

format('<tr><td>~w</td><td>~w</td><td>~w</td><td>~w</td></tr>', [ID, Title, AuthorID, CopyCount]).

generate\_table\_books :-

format('<table border="1">'),

format('<tr><th>ID</th><th>Название</th><th>ID автора</th><th>Число экземпляров</th></tr>'),

forall(book\_row(Row), write\_book\_row(Row)),

format('</table>').

generate\_table\_authors :-

format('<table border="1">'),

format('<tr><th>ID</th><th>Имя</th><th>Фамилия</th><th>Год рождения</th></tr>'),

forall(author\_row(Row), write\_author\_row(Row)),

format('</table>').

% Предикат для получения строк таблицы авторов

author\_row([ID, FirstName, LastName, BirthYear]) :-

odbc\_query(dsn, 'SELECT \* FROM Авторы', row(ID, FirstName, LastName, BirthYear)).

% Предикат для записи строки таблицы авторов в HTML

write\_author\_row([ID, FirstName, LastName, BirthYear]) :-

format('<tr><td>~w</td><td>~w</td><td>~w</td><td>~w</td></tr>', [ID, FirstName, LastName, BirthYear]).

% Предикат для генерации HTML-кода таблицы с изданиями

generate\_table\_editions :-

format('<table border="1">'),

format('<tr><th>ID</th><th>Издательство</th><th>Номер издания</th><th>Год издания</th><th>Количество страниц</th><th>Цена</th></tr>'),

forall(edition\_row(Row), write\_edition\_row(Row)),

format('</table>').

% Предикат для получения строк таблицы изданий

edition\_row([ID, Publisher, EditionNumber, Year, PageCount, Price]) :-

odbc\_query(dsn, 'SELECT \* FROM Издания', row(ID, Publisher, EditionNumber, Year, PageCount, Price)).

% Предикат для записи строки таблицы изданий в HTML

write\_edition\_row([ID, Publisher, EditionNumber, Year, PageCount, Price]) :-

format('<tr><td>~w</td><td>~w</td><td>~w</td><td>~w</td><td>~w</td><td>~w</td></tr>', [ID, Publisher, EditionNumber, Year, PageCount, Price]).

generate\_table\_books\_editions :-

format('<table border="1">'),

format('<tr><th>ID</th><th>ID книги</th><th>ID издания</th></tr>'),

forall(book\_edition\_row(Row), write\_book\_edition\_row(Row)),

format('</table>').

% Предикат для получения строк таблицы связей книг и изданий

book\_edition\_row([ID, BookID, EditionID]) :-

odbc\_query(dsn, 'SELECT \* FROM Книги\_Издания', row(ID, BookID, EditionID)).

% Предикат для записи строки таблицы связей книг и изданий в HTML

write\_book\_edition\_row([ID, BookID, EditionID]) :-

format('<tr><td>~w</td><td>~w</td><td>~w</td></tr>', [ID, BookID, EditionID]).

%Обработка главной страницы

home\_page(\_Request):-

reply\_html\_page(

title('Мананников М 6303'),

[

h1('Лабораторная работа 4'),

h2('Библиотека'),

h2('Поисковые запросы:'),

ul([

li(a([href('/task\_1')], 'Найти автора, у которого книга имеет самый ранний год издания')),

li(a([href('/task\_2')], 'Подчитать число изданий заданной книги')),

li(a([href('/task\_3')], 'Найти все книги, изданные в заданном издательстве за последние десять лет')),

li(a([href('/task\_4')], 'Найти все книги заданного автора')),

li(a([href('/task\_5')], 'Найти все книги, цена которых превышает заданную сумму'))

]),

form(

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

p(button([type(submit), formaction('/add\_requirements\_page')], 'Добавить автора'))

),

form(

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

p(button([type(submit), formaction('/delete\_requirements\_p')], 'Удалить автора'))

),

form(

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

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

),

form(

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

[table([

tr([th([align(left)],'ID'), td(input([name(id\_update)]))]),

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

])]

)

]

),

connect\_db(\_),

format('<html><head><title>Библиотека</title></head><body>'),

format('<h2>Таблица Книги</h2>'),

generate\_table\_books,

format('<h2>Таблица Авторы</h2>'),

generate\_table\_authors,

format('<h2>Таблица Издания</h2>'),

generate\_table\_editions,

format('<h2>Таблица Связь книг и изданий</h2>'),

generate\_table\_books\_editions,

format('</body></html>').

%Страница с добавлением нового автора

add\_requirements\_page(\_Request):-

reply\_html\_page(

title('Добавление автора'),

[form(

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

[

table([

tr([th('Имя'), td(input([name(nameauthor)]))]),

tr([th('Фамилия'), td(input([name(surnameauthor)]))]),

tr([th('Дата рождения'), td(input([name(dateofBirthday)]))]),

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

])

]

)]

)

.

%Добавление автора

add\_requirements(Request):-

http\_parameters(

Request,

[

nameauthor(Nameauthor, []),

surnameauthor(Surnameauthor, []),

dateofBirthday(DateofBirthday, [])

]

),

atom\_number(DateofBirthday, DateofBirthdayNum),

connect\_db(Connection),

format(atom(Query), 'INSERT INTO Авторы (имя, фамилия, год\_рождения) VALUES (''~w'', ''~w'', ~d)', [Nameauthor, Surnameauthor, DateofBirthdayNum]),

odbc\_query(Connection, Query),

odbc\_disconnect(Connection),

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

update\_page(Request):-

http\_parameters(

Request, [

id\_update(Id, [])

]),

atom\_number(Id,Id\_num),

connect\_db(Connection),

odbc\_prepare(Connection,

'SELECT имя, фамилия, год\_рождения FROM public.Авторы WHERE id = ?',

[integer],

Statement),

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

Row = row(N, S, D),

%atom\_number(D, D\_num),

odbc\_free\_statement(Statement),

odbc\_disconnect(Connection),

reply\_html\_page(

title('Обновление автора'),

[form(

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

[

table([

tr([th('Id'), td(input([name(id), value(Id\_num)]))]),

tr([th('Имя'), td(input([name(nameauthor), value(N)]))]),

tr([th('Фамилия'), td(input([name(surnameauthor), value(S)]))]),

tr([th('Дата рождения'), td(input([name(dateofBirthday), value(D)]))]),

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

] )

]

)]

).

%обновление таблицы

update\_author(Request):-

http\_parameters(

Request,

[

id(Id, []),

nameauthor(Nameauthor, []),

surnameauthor(Surnameauthor, []),

dateofBirthday(DateofBirthday, [])

]

),

atom\_number(Id, Id\_num),

atom\_number(DateofBirthday, DateofBirthdayNum),

connect\_db(Connection),

odbc\_prepare(

Connection,

'UPDATE public.Авторы SET имя = ?, фамилия = ?, год\_рождения = ? WHERE id = ?',

[varchar, varchar, integer, integer],

Statement

),

odbc\_execute(Statement, [Nameauthor, Surnameauthor, DateofBirthdayNum, Id\_num]),

odbc\_free\_statement(Statement),

odbc\_disconnect(Connection),

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

close\_conn(Connection, Statement) :-

odbc\_free\_statement(Statement),

odbc\_disconnect(Connection).

%Страница с удалением автора

delete\_requirements\_p(\_Request):-

reply\_html\_page(

title('Удаление автора'),

[form(

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

[

table([

tr([th('ID'), td(input([name(id1), type=number]))]),

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

])

]

)]

).

%Удаление автора

delete\_requirements(Request):-

http\_parameters(

Request,

[

id1(Id1, [])

]

),

connect\_db(Connection),

atom\_number(Id1, Id\_num1),

odbc\_prepare(

Connection,

'UPDATE Книги

SET автор\_id = NULL

WHERE id = (?);

DELETE FROM Авторы

WHERE id = (?)

',

[integer, integer],

Statement

),

odbc\_execute(Statement, [Id\_num1, Id\_num1]),

close\_conn(Connection, Statement),

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

%коннект к БД

connect\_db(Connection):-

odbc\_connect(

'swi',

Connection,

[

user(postgres),

password(rinat),

alias(dsn),

open(once)

]

).

%%%%%%%%%%%%%%%%%%%Создание БД

%Создание таблицы c изданиями

create\_table\_izd :-

connect\_db(\_),

odbc\_query(

dsn,

'

CREATE TABLE Издания (

id SERIAL PRIMARY KEY,

издательство VARCHAR(100),

номер\_издания INT,

год\_издания INT,

количество\_страниц INT,

цена INT)

'), odbc\_disconnect(dsn).

%Создание таблицы c авторами

create\_table\_author :-

connect\_db(\_),

odbc\_query(

dsn,

'

CREATE TABLE Авторы (

id SERIAL PRIMARY KEY,

имя VARCHAR(50),

фамилия VARCHAR(50),

год\_рождения INT)

'), odbc\_disconnect(dsn).

%Создание таблицы c книгами

create\_table\_book :-

connect\_db(\_),

odbc\_query(

dsn,

'

CREATE TABLE Книги (

id SERIAL PRIMARY KEY,

название VARCHAR(255),

автор\_id INT,

число\_экземпляров INT,

FOREIGN KEY (автор\_id) REFERENCES Авторы(id)

)

'), odbc\_disconnect(dsn).

%Создание таблицы c книга-Издание(много к много)

create\_table\_book\_has\_izd :-

connect\_db(\_),

odbc\_query(

dsn,

'

CREATE TABLE Книги\_Издания (

id SERIAL PRIMARY KEY,

книга\_id INT,

издание\_id INT,

FOREIGN KEY (книга\_id) REFERENCES Книги(id),

FOREIGN KEY (издание\_id) REFERENCES Издания(id)

)

'), odbc\_disconnect(dsn).

% Добавление автора в таблицу

add\_author :-

connect\_db(\_),

odbc\_query(

dsn,

'INSERT INTO Авторы (имя, фамилия, год\_рождения) VALUES

(''Aleksandr'',''Pushkin'', 1799),

(''Mikhail'',''Lermontov'', 1814),

(''Aleksandr'',''Blok'', 1880),

(''Sergey'',''Yesenin'', 1895),

(''Anna'',''Akmatova'', 1889)'

),

odbc\_disconnect(dsn).

% Добавление издания в таблицу

add\_izd :-

connect\_db(\_),

odbc\_query(

dsn,

'INSERT INTO Издания (издательство, номер\_издания, год\_издания, количество\_страниц, цена) VALUES

(''Ves mir'', 1, 2000, 150, 100),

(''Lenizdat'', 1, 1988, 222, 300),

(''Inapress'', 2, 1999, 304, 500),

(''Mir khobbi'', 1, 2015, 411, 210),

(''Lomonosov'', 3, 2022, 334, 450),

(''Lomonosov'', 1, 2014, 501, 150),

(''Lomonosov'', 2, 2011, 552, 50)'

),

odbc\_disconnect(dsn).

% Добавление книги в таблицу

add\_book :-

connect\_db(\_),

odbc\_query(

dsn,

'INSERT INTO Книги (название, автор\_id, число\_экземпляров) VALUES

(''Evgeniy Onegin'', 1, 100),

(''Geroy nashego vremeni'', 2, 150),

(''Neznakomka'', 3, 1000),

(''Mne ostalas odna zabava'', 4, 401),

(''Rekviem'', 5, 200)'

),

odbc\_disconnect(dsn).

% Добавление книга-Издание в таблицу

add\_book\_izd :-

connect\_db(\_),

odbc\_query(

dsn,

'INSERT INTO Книги\_Издания (книга\_id, издание\_id) VALUES

(1, 6),

(2, 3),

(3, 1),

(3, 5),

(4, 2),

(4, 7),

(5, 4)'

),

odbc\_disconnect(dsn).

%1. Найти автора, у которого книга имеет самый ранний год издания

task\_1(Authors) :-

connect\_db(Connection),

atomic\_list\_concat(['SELECT "Авторы"."id", "Авторы"."имя", "Авторы"."фамилия"

FROM "Авторы"

JOIN "Книги" ON "Авторы"."id" = "Книги"."автор\_id"

JOIN "Книги\_Издания" ON "Книги"."id" = "Книги\_Издания"."книга\_id"

JOIN "Издания" ON "Книги\_Издания"."издание\_id" = "Издания"."id"

WHERE "Издания"."год\_издания" = (SELECT MIN("год\_издания") FROM "Издания")

LIMIT 1'], Query),

findall([ID, FirstName, LastName], odbc\_query(Connection, Query, row(ID, FirstName, LastName)), Authors),

odbc\_disconnect(Connection).

%print\_authors([]).

%print\_authors([[ID, FirstName, LastName]|Rest]) :-

% format('<li>~w: ~w ~w</li>', [ID, FirstName, LastName]),

% print\_authors(Rest).

%task\_1\_page(\_Request):-

% task\_1(Authors),

% format('Content-type: text/html; charset=UTF-8~n~n', []),

% format('<html><head><title>Автор книги с самым ранним изданием</title></head><body>', []),

% print\_authors(Authors),

% format('<form style="display: inline-block;"><p><button type="submit" formaction="/">На главный экран</button></p></form></body></html>', []).

print\_authors([]).

print\_authors([[ID, FirstName, LastName]|Rest]) :-

format('<tr><td>~w</td><td>~w</td><td>~w</td></tr>', [ID, FirstName, LastName]),

print\_authors(Rest).

task\_1\_page(\_Request):-

task\_1(Authors),

format('Content-type: text/html; charset=UTF-8~n~n', []),

format('<html><head><h3>ДАвтор книги с самым ранним изданием:</h3></head><body>', []),

format('<table border="1">'),

format('<tr><th>ID</th><th>Имя</th><th>Фамилия</th></tr>'),

print\_authors(Authors),

format('</table>'),

format('<form style="display: inline-block;"><p><button type="submit" formaction="/">На главный экран</button></p></form></body></html>', []).

%2. Подчитать число изданий заданной книги

count\_editions(BookTitle, EditionsCount) :-

connect\_db(Connection),

format(atom(Query), 'SELECT COUNT(\*) AS count

FROM "Книги"

JOIN "Книги\_Издания" ON "Книги"."id" = "Книги\_Издания"."книга\_id"

WHERE "Книги"."название" = ''~w''', [BookTitle]),

odbc\_query(Connection, Query, row(EditionsCount)),

odbc\_disconnect(Connection).

task\_2\_page(\_Request):-

reply\_html\_page(

title('Введите книгу'),

[form(

[action=location\_by\_id(task\_2\_page\_V), method(post)],

[

table([

tr([th('Введите книгу:'), td(input([name(book)]))]),

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

] )

]

)]

).

task\_2\_page\_V(Request):-

http\_parameters(

Request,

[book(B, [])]

),

count\_editions(B, Count),

reply\_html\_page(

title('Число изданий заданной книги'),

[h3('Число изданий заданной книги:'), p(Count),

form(

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

p(button([type(submit), formaction('/')], 'На главный экран'))

)]

).

%3. Найти все книги, изданные в заданном издательстве за последние десять лет

task\_3(Publisher, Books) :-

connect\_db(Connection),

get\_time(CurrentTime),

stamp\_date\_time(CurrentTime, DateTime, 'UTC'),

date\_time\_value(year, DateTime, CurrentYear),

TenYearsAgo is CurrentYear - 10,

format(atom(Query), 'SELECT Книги.id, Книги.название

FROM Книги

JOIN Книги\_Издания ON Книги.id = Книги\_Издания.книга\_id

JOIN Издания ON Книги\_Издания.издание\_id = Издания.id

WHERE Издания.издательство = ''~w''

AND Издания.год\_издания >= ~w', [Publisher, TenYearsAgo]),

findall([ID, Title], odbc\_query(Connection, Query, row(ID, Title)), Books),

odbc\_disconnect(Connection).

print\_books([]).

print\_books([[ID, Title]|Rest]) :-

format('<tr><td>~w</td><td>~w</td></tr>', [ID, Title]),

print\_books(Rest).

task\_3\_page(\_Request):-

reply\_html\_page(

title('Введите издательство'),

[form(

[action=location\_by\_id(task\_3\_page\_V), method(post)],

[

table([

tr([th('Введите издательство:'), td(input([name(izdadelstvo)]))]),

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

] )

]

)]

).

task\_3\_page\_V(Request):-

http\_parameters(

Request,

[izdadelstvo(Izd, [])]

),

task\_3(Izd, Books),

format('Content-type: text/html; charset=UTF-8~n~n', []),

format('<p>Книги, изданные в заданном издательстве за последние десять лет</p>'),

format('<table border="1">'),

format('<tr><th>ID</th><th> Название</th></tr>'),

print\_books(Books),

format('</table>'),

format('<form style="display: inline-block;"><p><button type="submit" formaction="/">На главный экран</button></p></form></body></html>', []).

%4. Найти все книги заданного автора

task\_4(LastName, Books) :-

connect\_db(Connection),

format(atom(Query), 'SELECT Книги.id, Книги.название FROM Книги JOIN Авторы ON Книги.автор\_id = Авторы.id WHERE Авторы.фамилия = ''~w''', [LastName]),

findall([ID, Title], odbc\_query(Connection, Query, row(ID, Title)), Books),

odbc\_disconnect(Connection).

task\_4\_page(\_Request):-

reply\_html\_page(

title('Введите фамилию автора'),

[form(

[action=location\_by\_id(task\_4\_page\_V), method(post)],

[

table([

tr([th('Введите автора:'), td(input([name(avtor)]))]),

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

] )

]

)]

).

task\_4\_page\_V(Request):-

http\_parameters(

Request,

[avtor(Avt, [])]

),

task\_4(Avt, Books),

format('Content-type: text/html; charset=UTF-8~n~n', []),

format('<p>Найти все книги заданного автора</p>'),

format('<table border="1">'),

format('<tr><th>ID</th><th> Название</th></tr>'),

print\_books(Books),

format('</table>'),

format('<form style="display: inline-block;"><p><button type="submit" formaction="/">На главный экран</button></p></form></body></html>', []).

%5. Найти все книги, цена которых превышает заданную сумму

task\_5(Price, Books) :-

connect\_db(Connection),

format(atom(Query), 'SELECT "Книги"."id", "Книги"."название"

FROM "Книги"

JOIN "Книги\_Издания" ON "Книги"."id" = "Книги\_Издания"."книга\_id"

JOIN "Издания" ON "Книги\_Издания"."издание\_id" = "Издания"."id"

WHERE "Издания"."цена" > ''~w''', [Price]),

findall([ID, Title], odbc\_query(Connection, Query, row(ID, Title)), Books),

odbc\_disconnect(Connection).

task\_5\_page(\_Request):-

reply\_html\_page(

title('Все книги, цена которых превышает заданную сумму'),

[form(

[action=location\_by\_id(task\_5\_page\_S), method(post)],

[

table([

tr([th('Введите цену:'), td(input([name(cena)]))]),

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

] )

]

)]

).

task\_5\_page\_S(Request):-

http\_parameters(

Request,

[cena(S, [])]

),

% Преобразование S в число

atom\_number(S, SNum),

task\_5(SNum, Books),

format('Content-type: text/html; charset=UTF-8~n~n', []),

format('<p>Книги, цена которых превышает заданную сумму:</p>'),

format('<table border="1">'),

format('<tr><th>ID</th><th> Название</th></tr>'),

print\_books(Books),

format('</table>'),

format('<form style="display: inline-block;"><p><button type="submit" formaction="/">На главный экран</button></p></form></body></html>', []).