ՀԱՅԱՍՏԱՆԻ ՀԱՆՐԱՊԵՏՈՒԹՅԱՆ ԿՐԹՈՒԹՅԱՆ, ԳԻՏՈՒԹՅԱՆ,ՄՇԱԿՈՒՅԹԻ ԵՎ ՍՊՈՐՏԻ ՆԱԽԱՐԱՐՈՒԹՅՈՒՆ

ՀԱՅԱՍՏԱՆԻ ԱԶԳԱՅԻՆ ՊՈԼԻՏԵԽՆԻԿԱԿԱՆ ՀԱՄԱԼՍԱՐԱՆ (ՀԻՄՆԱԴՐԱՄ)

ԻՆՍՏԻՏՈՒՏ *ՏԵՂԵԿԱՏՎԱԿԱՆ ԵՎ ՀԵՌԱՀԱՂՈՐԴԱԿՑԱԿԱՆ ՏԵԽՆՈԼՈԳԻԱՆԵՐԻ ՈՒ ԷԼԵԿՏՐՈՆԻԿԱՅԻ*

ԱՄԲԻՈՆ ԱԼԳՈՐԻԹՄԱԿԱՆ ԼԵԶՈՒՆԵՐԻ ԵՎ ԾՐԱԳՐԱՎՈՐՄԱՆ

**ՀԵՏԱԶՈՏԱԿԱՆ ԱՇԽԱՏԱՆՔԻ ԱՌԱՋԱԴՐԱՆՔ**

***«Տվյալների մշակման բաշխված համակարգեր»*** առարկայից

Աշխատանքի թեման՝ «Գրադարան» տեղեկատվական համակարգ

Ակադեմիական խմբի ուսանող՝ ՄՏՏ319 Հակոբջանյան Էլեն

Հաշվեբացատրագրի բովանդակությունը

1. Տիտղոսաթերթ
2. ԿԱ առաջադրանքը
3. Ալգորիթմը, ալգորիթմի նկարագրությունը
4. Ծրագիրը, ծրագրի նկարագրությունը
5. Եզրակացություն

Ամբիոնի վարիչի պ/կ` ***Ս. Ս. Ավետիսյան*** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

ամսաթիվ ստորագրություն

Աշխատանքի ղեկավար \_\_\_Ա․Սիմոնյան\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

ազգանուն անուն ամսաթիվ ստորագրություն

Աշխատանքի առաջադրանքը ստացա ***10.02.2024թ.*** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

ամսաթիվ ստորագրություն

ՀԱՅԱՍՏԱՆԻ ՀԱՆՐԱՊԵՏՈՒԹՅԱՆ ԿՐԹՈՒԹՅԱՆ, ԳԻՏՈՒԹՅԱՆ,ՄՇԱԿՈՒՅԹԻ ԵՎ ՍՊՈՐՏԻ ՆԱԽԱՐԱՐՈՒԹՅՈՒՆ

ՀԱՅԱՍՏԱՆԻ ԱԶԳԱՅԻՆ ՊՈԼԻՏԵԽՆԻԿԱԿԱՆ ՀԱՄԱԼՍԱՐԱՆ (ՀԻՄՆԱԴՐԱՄ)

ՏՀՏԷ ինստիտուտ

ԱԼ և Ծ ամբիոն

**ՀԱՇՎԵԲԱՑԱՏՐԱԳԻՐ**

***«Տվյալների մշակման բաշխված համակարգեր» առարկայի հետազոտական աշխատանքի***

Թեմա՝ «Գրադարան» տեղեկատվական համակարգ

Ուսանող՝ \_\_\_\_\_\_ Հակոբջանյան Էլեն Աշոտի\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
 *ազգանուն, անուն հայրանուն* ստորագրություն

Ղեկավար՝ \_\_\_\_\_\_\_ Ա․Սիմոնյան\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
 *ազգանուն, անուն հայրանուն* ստորագրություն

Ամբիոնի վարիչի պ/կ ***Ս. Ս. Ավետիսյան*** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

ստորագրություն

Հանձնաժողովի անդամներ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

ստորագրություն

Երևան 2024

Բովանդակություն

[ Խնդրի դրվածք 4](#_Toc167868894)

[ 1. Ալգորիթմի աշխատանք 5](#_Toc167868895)

[ 2. Ծրագրային իրականացում 6](#_Toc167868896)

[ 2.1. Տվյալների բազայի հարաբերությունների դիագրամ 6](#_Toc167868897)

[ 2.2. Աղյուսակներ 7](#_Toc167868898)

[ 2.3. Տրիգերներ, օգտագործված տրիգերներ 11](#_Toc167868899)

[ 2.4. Ֆունկցիաներ, օգտագործված ֆունկցիաներ 12](#_Toc167868900)

[ 2.5. Պրոցեդուրաներ, օգտագործված պրոցեդուրաներ 12](#_Toc167868901)

[ 2.6. Օգտագործված գրադարաններ 13](#_Toc167868902)

[ 2.7. Ծրագրի իմպլեմենտացիա 14](#_Toc167868903)

[ 3. Գրաֆիկական ինտերֆեյս 15](#_Toc167868904)

[ Եզրակացություն 16](#_Toc167868905)

[ Գրականության ցանկ 17](#_Toc167868906)

* Խնդրի դրվածք

Ստեղծել «Գրադարան» տեղեկատվական համակարգ, որը կտրամադրի ծրագրից օգտվելու երկու ռեժիմներ՝ որպես օգտատեր կամ որպես ադմին։ Օգտատիրոջ հնարավորությունները պետք է ներառեն գրքերի ցանկին հասանելիություն ունենալ, տրամադրված ցանկի մեջ սահմանափակումներ կիրառել՝ փնտրվող գրքի պարամետրեր նշելով, հնարավորություն ունենալ գիրք ամրագրելու և կարողանալ հետևել իր վերցրած գրքերի ժամկետներին ու պատմությանը։ Ադմինը պետք է հնարավորություն ունենա փոփոխություններ կատարել ինչպես գրադարանի գրքերի ցանկում, այնպես էլ օգտատերերի ցանկում, ի տարբերություն սովորական օգտատիրոջ ադմինը կարող է տեսնել բոլոր օգտատերերի վերցրած գրքերի ցուցակին, բայց, իր հերթին, ադմինի կարգավիճակով գրքեր ամրագրելու հնարավորություն պետք է չլինի։

Գրքերի ցուցակի և օգտատերերի ցուցակի դիտումը, դրանց մեջ փոփոխությունների կատարումը, ինչպես նաև օգտատիրոջ հաշվի հետ գործողություններն իրականացվելու են տվյալների բազայի համապատասխան աղյուսակներին դիմելով։

Ծրագրային իրականացումը տրվել է C++ ծրագրավորման լեզվի միջոցով, Qt գրաֆիկական միջավայրում։

|  |  |  |
| --- | --- | --- |
|  | Տեխնիկական պարամետրեր | Արժեք |
| 1. | Օգտագործվող ծրագրավորման լեզու | C++ |
| 2. | Գրաֆիկական ինտերֆեյս | Qt |
| 3. | Օգտագործվող գրադարանների ցանկ | STL  QSqlQuery  QSqlDatabase |
| 4. | Օգտագործվող տվյալների բազա | SQLite |
| 5. | Հարցումների իրականացման միջին ժամանակ, վրկ | ≈ 0.5 |

* 1. Ալգորիթմի աշխատանք

Ծրագիրը բաղկացած է երկու հիմնական էջերից։ Օգտատիրոջ էջն իր մեջ ներառում է ծրագրից օգտվելու ռեժիմի մասին ինֆորմացիա։ Ադմին ռեժիմի դեպքում հասանելի է աղյուսակներում փնտրումից բացի աղյուսակներում փոփոխություններ իրականացնելու հնարավորություն, իսկ օգտատիրոջ դեպքում նույն էջը պատասխանատու է գրքերի ամրագրման, վերադարձի և ունեցած գրքերի ցանկին հետևելու հնարավորությունների համար։

Հիմնական էջը նույնն է երկու ռեժիմների համար։ Այն հնարավորություն է տալիս գրադարանում գրանցված բոլոր հնարավոր գրքերի մեջ փնտրում իրականացնել՝ կիրառելով որոշակի սահմանափակումներ գրքի անվան, հեղինակի, տպագրման տարեթվի, գրքի գրադարանում առկա լինելու և այլ գիրքը նկարագրող փոփոխականների վրա։

Նշված սահմանափակումների կիրառման հետևանքով գեներացվում է հարցում տվյալների բազա, ամեն գեներացված հարցում արտածվում է ստորին պատուհանում, իսկ փնտրման արդյունքները երևում են վերին պատուհանում՝ աղյուսակային ֆորմատով։

* 2. Ծրագրային իրականացում

## 2.1. Տվյալների բազայի հարաբերությունների դիագրամ



Նկար 2.1.1. Աղյուսակների միջև հարաբերություններ

## 2.2. Աղյուսակներ

Ծրագրային իրականացման համար մշակվել և կիրառվել է տվյալների բազա, որն ունի հետևյալ կառուցվածքը՝

**CREATE TABLE reservation\_status (**

**id INT,**

**status\_value VARCHAR(50),**

**CONSTRAINT pk\_res\_status PRIMARY KEY (id)**

**);**

**CREATE TABLE category (**

**id INT,**

**category\_name VARCHAR(100),**

**CONSTRAINT pk\_category PRIMARY KEY (id)**

**);**

**CREATE TABLE book (**

**id INT,**

**title VARCHAR(500),**

**category\_id INT,**

**publication\_date DATE,**

**copies\_owned INT,**

**CONSTRAINT pk\_book PRIMARY KEY (id),**

**CONSTRAINT fk\_book\_category FOREIGN KEY (category\_id) REFERENCES category(id)**

**);**

**CREATE TABLE author (**

**id INT,**

**first\_name VARCHAR(300),**

**last\_name VARCHAR(300),**

**CONSTRAINT pk\_author PRIMARY KEY (id)**

**);**

**CREATE TABLE book\_author (**

**book\_id INT,**

**author\_id INT,**

**CONSTRAINT fk\_bookauthor\_book FOREIGN KEY (book\_id) REFERENCES book(id),**

**CONSTRAINT fk\_bookauthor\_author FOREIGN KEY (author\_id) REFERENCES author(id)**

**);**

**CREATE TABLE member\_status (**

**id INT,**

**status\_value VARCHAR(50),**

**CONSTRAINT pk\_memberstatus PRIMARY KEY (id),**

**);**

**CREATE TABLE member (**

**id INT,**

**first\_name VARCHAR(300),**

**last\_name VARCHAR(300),**

**joined\_date DATE,**

**active\_status\_id INT,**

**CONSTRAINT pk\_member PRIMARY KEY (id),**

**CONSTRAINT fk\_member\_status FOREIGN KEY (active\_status\_id) REFERENCES member\_status(id)**

**);**

**CREATE TABLE reservation (**

**id INT,**

**book\_id INT,**

**member\_id INT,**

**reservation\_date DATE,**

**reservation\_status\_id INT,**

**CONSTRAINT pk\_reservation PRIMARY KEY (id),**

**CONSTRAINT fk\_res\_book FOREIGN KEY (book\_id) REFERENCES book(id),**

**CONSTRAINT fk\_res\_member FOREIGN KEY (member\_id) REFERENCES member(id)**

**);**

**CREATE TABLE fine\_payment (**

**id INT,**

**member\_id INT,**

**payment\_date DATE,**

**payment\_amount INT,**

**CONSTRAINT pk\_fine\_payment PRIMARY KEY (id),**

**CONSTRAINT fk\_finepay\_member FOREIGN KEY (member\_id) REFERENCES member(id)**

**);**

**CREATE TABLE loan (**

**id INT,**

**book\_id INT,**

**member\_id INT,**

**loan\_date DATE,**

**returned\_date DATE,**

**CONSTRAINT pk\_loan PRIMARY KEY (id),**

**CONSTRAINT fk\_loan\_book FOREIGN KEY (book\_id) REFERENCES book(id),**

**CONSTRAINT fk\_loan\_member FOREIGN KEY (member\_id) REFERENCES member(id)**

**);**

**CREATE TABLE fine (**

**id INT,**

**book\_id INT,**

**loan\_id INT,**

**fine\_date DATE,**

**fine\_amount INT,**

**CONSTRAINT pk\_fine PRIMARY KEY (id),**

**CONSTRAINT fk\_fine\_book FOREIGN KEY (book\_id) REFERENCES book(id),**

**CONSTRAINT fk\_fine\_loan FOREIGN KEY (loan\_id) REFERENCES loan(id)**

**);**

## 2.3. Տրիգերներ, օգտագործված տրիգերներ

Տրիգերը պահվող ընթացակարգ է տվյալների բազայում, որն ինքնաբերաբար ներգրավվում է, երբ տեղի է ունենում հատուկ իրադարձություն տվյալների բազայում: Այն SQL հայտարարությունների հավաքածու է` հատուկ անուններով, որոնք պահվում են համակարգի հիշողության մեջ: Այն պատկանում է պահված ընթացակարգերի հատուկ դասին, որոնք ավտոմատ կերպով կանչվում են ի պատասխան տվյալների բազայի սերվերի իրադարձություններին: Յուրաքանչյուր տրիգեր ունի դրան կցված սեղան:

Տրիգերն ինքնաբերաբար կանչվում է, երբ տեղի է ունենում տվյալների փոփոխության միջոցառում, որը տեղի է ունենում կցված սեղանին, ինչը հիմնական տարբերությունն է տրիգերի և պրոցեդուրայի միջև:

* Տրիգերները չեն կարող ձեռքով կանչվել և իրականացվել։
* Հնարավորություն չկա, որ տրիգերին պարամետրեր փոխանցվեն:
* Գործարք չի կարող իրականացվել տրիգերի կանչում:

Ծրագրային իրականացման ընթացքում մշակվել են հետևյալ տրիգերները՝

* Օգտատիրոջ կարգավիճակից գրքերի աղյուսակում փոփոխություններ թույլ չտալու և օգտատերին այդ մասին զգուշացնող անվտանգության տրիգեր։

**create trigger safety**  
**on book**  
for  
**create\_table,alter\_table,drop\_table**  
as  
**print 'invalid access for modification of a table in database'**

* Գրքերի ամրագրման և վերադարձի տրիգեր, որը հասանելի գրքի օրինակների քանակը շատացնում կամ քչացնում է մեկով։

**create trigger amount**  
**before delete on reservation**  
for **each row**  
set  
**copies\_owned = copies\_owned + 1;**

## 2.4. Ֆունկցիաներ, օգտագործված ֆունկցիաներ

SQL ֆունկցիաները ներկառուցված ծրագրեր են, որոնք օգտագործվում են տվյալների բազայում տարբեր գործողություններ կատարելու համար:

SQL- ում գործում են երկու տեսակի ֆունկցիաներ՝

* Համախառն (Aggregate) ֆունկցիաներ
* Սկալյար (Scalar) ֆունկցիաներ

**SQL համախառն ֆունկցիաները** գործում են տվյալների խմբի վրա և վերադարձնում են եզակի ելք: Դրանք հիմնականում օգտագործվում են խմբի հետ ըստ տվյալների, տվյալների խմբավորելու համար:

**SQL սկալյար գործառույթները** ներկառուցված գործառույթներ են, որոնք գործում են մեկ արժեքով և վերադարձնում են մեկ արժեք:

SQL-ում սկալյար գործառույթները օգնում են տվյալների արդյունավետ մանիպուլյացիայում և SQL հարցումներում բարդ հաշվարկների պարզեցում:

Ծրագրային իրականացման ընթացքում մշակվել են հետևյալ ֆունկցիաները՝

## 2.5. Պրոցեդուրաներ, օգտագործված պրոցեդուրաներ

SQL- ում պահված պրոցեդուրան SQL հարցումների խումբ է, որը կարող է պահվել և նորից օգտագործվել բազմիցս: Այն շատ օգտակար է, քանի որ այն նվազեցնում է SQL հարցումներ կրկնելու անհրաժեշտությունը: Այն ուժեղացնում է արդյունավետությունը, վերաօգտագործումը և անվտանգությունը տվյալների բազայի կառավարման ոլորտում:

Օգտագործողները կարող են նաև պարամետրեր փոխանցել պահվող պրոցեդուրաներին, որպեսզի պահված պրոցեդուրան օգտվի պարամետրերի արժեքներից:

Պահպանված պրոցեդուրաները ստեղծվում են տվյալների բազայում մեկ կամ մի քանի DML գործառնություններ կատարելու համար: Այն իրենից ներկայացնում է SQL-ի մի խումբ հայտարարություններ, որոնք ընդունում են որոշ ներդրումներ պարամետրերի տեսքով, կատարում են որոշակի առաջադրանք, և կարող են չվերադառնալ արժեք:

Ծրագրային իրականացման ընթացքում մշակվել են հետևյալ պրոցեդուրաները՝

* Պրոցեդուրա, որն արտածում է տրված ամսաթվից նոր գրքերը, որը օգտակար է լինում հետազոտական աշխատանքների համար թարմ ռեսուրսների փնտրման ժամանակ։

CREATE PROCEDURE getNewestSources  
 @date timestamp  
AS  
BEGIN  
 SELECT title, publication\_date  
 FROM book  
 WHERE publication\_date >= @date;  
END;

* Պրոցեդուրա գեներացված օգտատիրոջը տրամադրված ֆիլտրի հնարավորությամբ, որը վերոնշյալ պրոցեդուրայի ընդլայնված տարբերակն է։

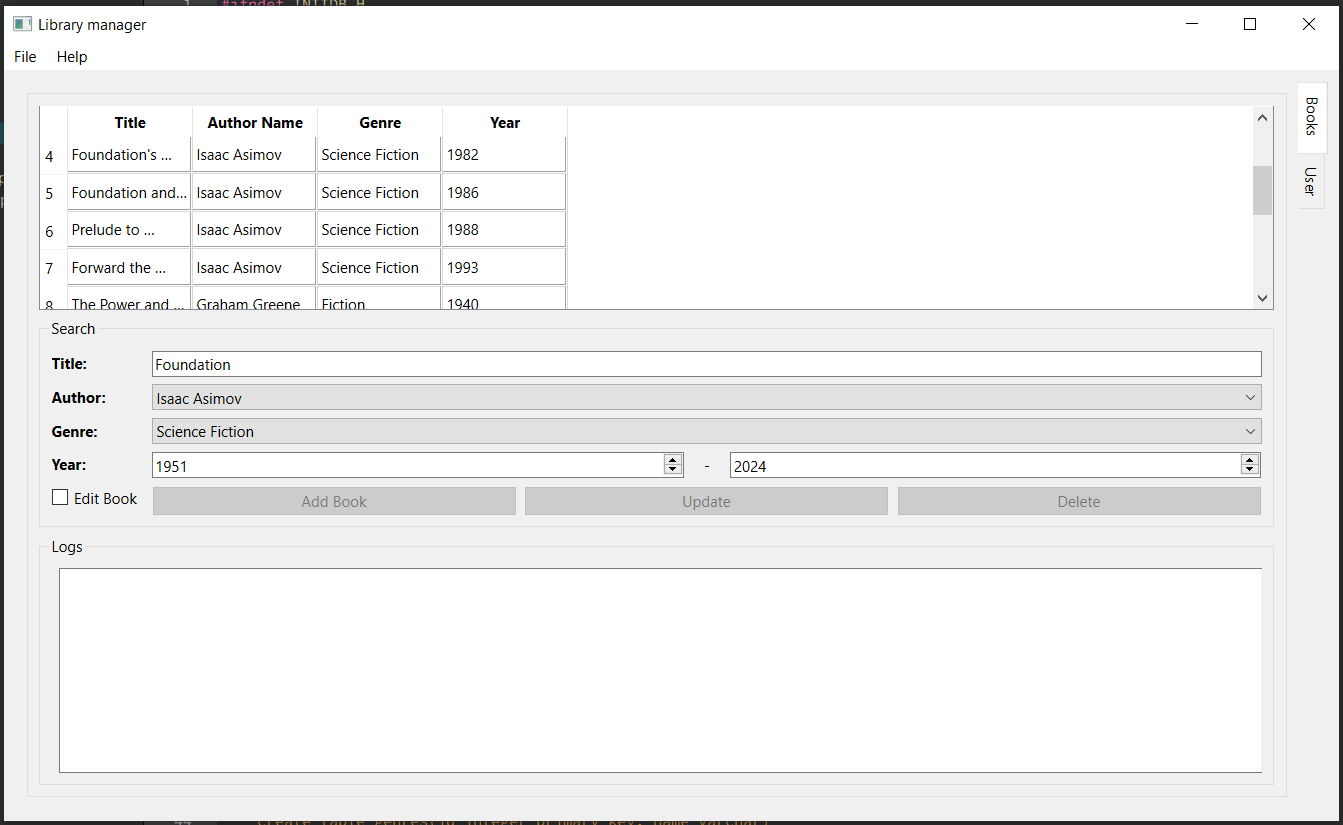
## 2.6. Օգտագործված գրադարաններ

* **QSqlDatabase** դասը տրամադրում է ինտերֆեյս՝ կապի միջոցով տվյալների բազա մուտք գործելու համար: QSqlDatabase-ի օբյեկտն իրենից հենց ներկայացնում է կապը: Կապը մատուցում է տվյալների բազա հասանելիություն:
* **QSqlQuery**-ն ընդգրկում է SQL հարցումներից տվյալներ ստեղծելու, նավարկելու և առբերելու գործառույթները, որոնք կատարվում են QSqlDatabase-ում: Այն կարող է օգտագործվել DML (տվյալների մանիպուլյացիայի լեզու) հայտարարությունները կատարելու համար, ինչպիսիք են ընտրելը, տեղադրումը, թարմացումը և ջնջելը, ինչպես նաև DDL (տվյալների սահմանման լեզու) հայտարարությունները, ինչպիսիք են, ստեղծել աղյուսակը: Այն կարող է օգտագործվել նաև տվյալների բազայի հատուկ հրամաններ կատարելու համար, որոնք ստանդարտ SQL չեն։
* **QSqlTableModel**-ը բարձր մակարդակի ինտերֆեյս է` մեկ աղյուսակից տվյալների բազայի գրառումներ կարդալու և գրելու համար: Այն կառուցված է ստորին մակարդակի QSqlQuery-ի վրայից և կարող է օգտագործվել տվյալներ դիտելու համար, ինչպիսիք են QTableView-ը:

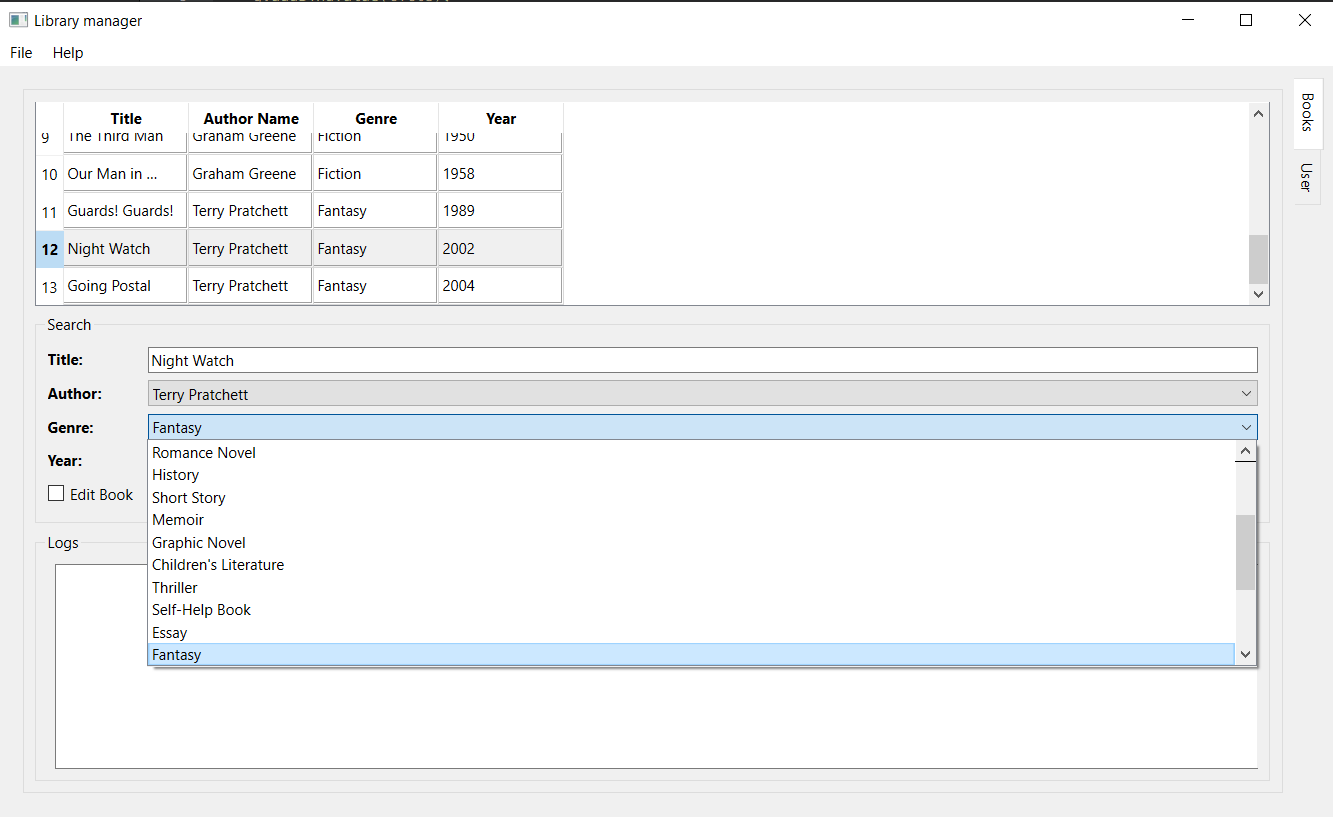
## 2.7. Ծրագրի իմպլեմենտացիա

https://github.com/ElenHakobjanyan/Library-Management-System/

* 3. Գրաֆիկական ինտերֆեյս



Նկար 3.1. Փնտրման էջ



Նկար 3.2. Փնտրման համար տրվող սահմանափակումների ընտրություն

# Եզրակացություն

Գրադարանային համակարգի ավտոմատացումը հեշտացնում է ինֆորմացիայի և ռեսուրսների փնտրման գործընթացը։ Ընդգրկելով նման համակարգերը ուսումնասիրություններ իրականացնելու պրոցեսի մեջ հնարավոր կդառնա կրճատել փնտրտուքի վրա ծախսվող ժամանակը, որը կարելի կլինի ավելի արդյունավետ ծախսել։ Որակյալ գրադարանային համակարգի մշակումը քայլ է դեպի արդյունավետ հետազոտական պրոցես։

# Գրականության ցանկ

1. Adrienne Watt, Nelson Eng, Database Design - 2nd Edition, BCcampus, BC Open Textbook Project, 2014 – 128 p
2. B. Schwartz, P. Zaitsev, V. Tkachenko, J. D. Zawodny, A. Lentz, High Performance MySQL: Optimization, Backups, Replication, and More 2nd Edition, O'Reilly Media, 2008, 684p
3. Տվյալների բազայի արդյունավետ ձևավորում: <https://www.databasestar.com/sql-roadmap/>
4. Ուղեցույց։ <https://bbrumm.gumroad.com/l/oqggau>