CS5200 Class Project Final Report

Zheyu Jin, jin.zh@husky.neu.edu
Yankang Jing, jing.ya@husky.neu.edu

CS5200 Class Project Final Report
- Background and specific aims
- Preliminary design
- Database Design Diagram
- Data Generation Queries
- SQL Queries for Applications and Reports
- Personal Report of Yankang Jing
- Personal Report of Zheyu Jin
- Contribution Breakdown

Appendix
- SQL Queries for DB creation and Data generation
  - Statistics
- Two basic helper functions.
  - --get_random_string(integer)
  - --get_random_number(integer, integer)
- Application for library users.
  - --Search books.ISBN
  - --Search books. Author or Title
  - --return a book
  - -- get user borrow limit
  - --Borrow a book
  - --Claim a book loss.
  - --Pay the penalty bill.
  - --Ask library to import a book.
- Application for library manager
  - --import a book
  - --get book id
  - --Delete a user's account from library.
  - --Delete a publication from library.
- Reports
  - --The top N most borrowed books list within a given periods.
  - --The top N most wanted books at the moment.
  - --A user's borrow history
  - --A user's penalty status.
- Roles and Privileges

Background and specific aims

This is a class project proposal for CS5200 (database management systems). We are planning to design and build a database for general university libraries, which focuses on daily management of books. We are still working on the improvement of our database and we will keep on making it more and more reasonable and logical.

Preliminary design

Now we have a preliminary idea about our database. Firstly, we design a set of applications and reports. Some examples of applications and reports we compose are simply showed here.

Application for library users:

1. Search books.
2. Borrow/return books.
3. Claim a book loss.
4. Pay the over-due penalty bill.
5. Ask the library to import a book (book wish list).
6.

Application for library manager:

1. Import books from external data sources (json, XML)
2. Remove a publication from library.

Reports:

1. The most borrowed books list within a given periods.
2. User's borrow history.
3. User's penalty status.

To achieve the application above, we could have a lot of tables. We preliminary generate 9 entity tables and 7 relation tables. Followings are example tables and their attributes :

Entities:

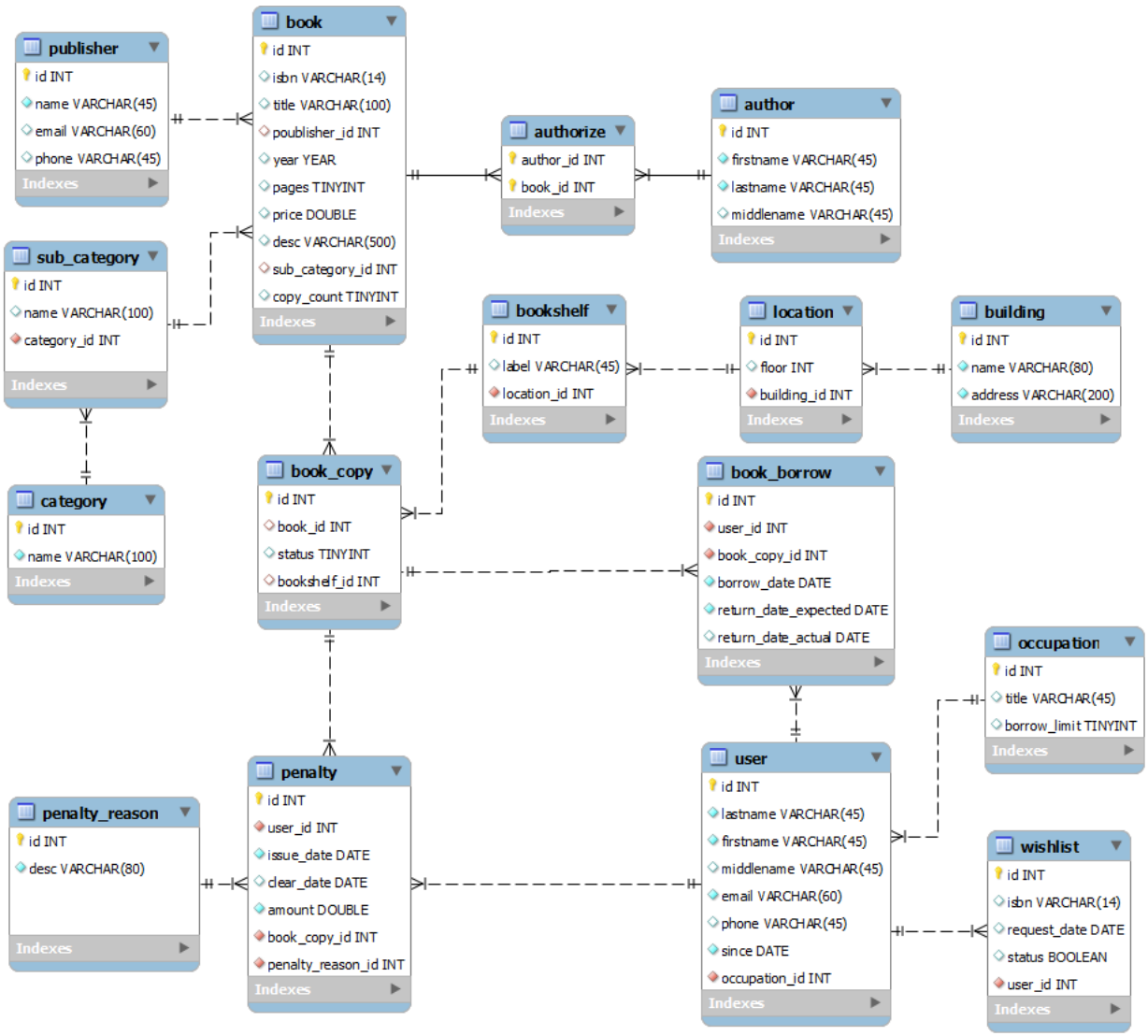
book (id, ISBN, name, author, publish_date, item_count….)
book_item(book_item_id, book_id, status)
user(id, name, birthdate, occupation, account_number,department ...)
bookshelf(id, location, status...)
location(id, bldgname, floor, room)
account(id, balance, order_id...)
locker(id, location,status...)
penalty_bill(id, userid, amount, date, status)
wishlist(id, userid, isbn, status)

Relations:

book_borrow(pid, bid, borrow_data, return_date, estimate_date...)

book_location(bid, bsid, book_status...)
locker_rent(lid, pid, start_time, end_time...)

Database Design Diagram



Data Generation Queries

Note:Please see the appendix for the code of below parts.

DB Schema Creation

Data population

get_random_number-

get_random_string-

populate table user from csv-

populate table wishlist-

populate table sub_category from csv-

populate table publisher-

populate table book-

populate table book_copy-

populate table occupation from csv-

populate table book_borrow -

creat csv file from dvdsales.customers-

populate table author-

populate table bookshelf-

populate table building -

populate table location -

populate table penalty_reason-

populate table penalty-

populate table authorize-

Statistics

SQL Queries for Applications and Reports

Note:Please see the appendix for the code of below parts.

Two basic helper functions.

--get_random_string(integer)

--get_random_number(integer, integer)

Application for library users.

--Search books.ISBN

--Search books. Author or Title

--return a book

--get user borrow limit

--Borrow a book

--Claim a book loss.

--Pay the penalty bill.

--Ask library to import a book.

Reports.

--The top N most borrowed books list within a given periods.

--The top N most wanted books at the moment.

--A user's borrow history

--A user's penalty status.

Application for library manager

--import a book

--Delete a user's account from library.

--Delete a publication from library.

Roles and Privileges

--web_admin; /*admin with web interface*/

--local_admin;/*admin with local machine*/

--web_client; /*normal web users*/

Personal Report of Yankang Jing

As one of the two group member of our project team, I worked with our team leader Zheyu Jin on our database management system of library.

In this project, we review the lessons several times and we also read some online materials and learnt a lot from others examples of database management system on the internet. From this project, I went over the processes of how to design the database and optimize its structure, including building reasonable ER tables in the database to make the database more clear and comprehensive. I become familiar with estimating and generating data in the table and adding proper index for searching faster and more accurate. I also practiced the PGSQL script programming with using pgAdmin. Beyond the database itself, I learn how to using many applications in the Google drive to make a co-work with your partner.

For the contribution of individual person, I do not know how to divide it. Probably only the code could be divided in one project teamwork. At the beginning of the project, I and Zheyu talked several times for the topic of our project and finally we decided to generate the database management system for library. Then we work together on the structure of the database and after several times discussions, we successfully made our database structure comprehensive, clear and simplified, and then we generated the diagram.
Then for the next several steps, we began the database building process which needs a lot of coding. Honestly speaking, I was not as practiced and efficient as Zheyu on the coding and made less contribution than Zheyu. For the table generation step, I generated about half of the tables. And for the application and report part, I finally generated the reports and also help Zheyu to check the other code.

Due to the time difference and location distance from I (Boston) and Zheyu(Seattle), we should have been hard to work together and have a good communication. However, for these steps, I am glad that Zheyu is an expert of Google drive so that we could communicate much easier on the Google drive with revising the same document at the same time. We communicate with each other not only through email, but also through other methods such as Skype and instant message. That makes our teamwork much easier. Besides, I will here say many thanks to Zheyu for his help on telling me a lot about his experience about this field when I met questions on coding.

In conclusion, I think that Zheyu is a good guy who is not only good at computational skills but also very nice to work with. He is a nice person who will communicate with you in his patience. He likes to share his experience and help me on my questions. He is also a good leader and he could explain his opinion and suggestion in a soft way. Finally, thanks Zheyu and hope we could have more communication in the future!

Personal Report of Zheyu Jin

Lessons I learned can be categorized as technical and non-technical categories.

Technical skills I've learned while working on this project includes database design, role management, pgsql script programming, large data generation & performance tuning, DB backup and restore.

Non-technical skills learned is mainly about team working and collaboration. The most outstanding problem we had throughout the project is the delay and probably this is because none of us had an experience of working with a teammate with 3 hours time difference. However, what I observed during this project is that when estimating the time cost of work, we were both too optimistic. This is why I said in the presentation that starting when it seems necessary will definitely cause delay.

Another important non-technical lesson is that knowledge sharing (synchronize what you know) between members is critical for teamwork, especially when working with someone from another field. As a computer science graduate student, I should have actively shared the knowledge with teammate who is a biology student in order to improve the co-working experience. This experience will guide me in future teamwork with non-CS people.

Though Yankang is a non-CS student, he has a good understanding about programming and quick in learning new things related to Computer Science, which truly impressed me. While he is currently in a co-op job, he participated in the teamwork with respectful time investment and it was a joy to work with him.

Contribution Breakdown

Database Design:

Finished together with Yankang Jing.

Data Generation:

7 tables by Zheyu Jin

7 tables by Yankang Jing

Applications and Reports query coding:

9 by Zheyu Jin.

5 by Yankang Jing.

Presentation & Final Report:

together with Yankang Jing.

Appendix

SQL Queries for DB creation and Data generation

DB Schema Creation

```
SET statement_timeout = 0;
SET client_encoding = 'UTF8';
SET standard_conforming_strings = on;
SET check_function_bodies = false;
SET client_min_messages = warning;

 TOC entry 7 (class 2615 OID 19081)-

 Name: libdb; Type: SCHEMA; Schema: ; Owner: postgres


CREATE SCHEMA libdb;


ALTER SCHEMA libdb OWNER TO postgres;

SET search_path = libdb, pg_catalog;

SET default_tablespace = '';

SET default_with_oids = false;

 TOC entry 189 (class 1259 OID 19260)-
 Dependencies: 7-

 Name: author; Type: TABLE; Schema: libdb; Owner: postgres; Tablespace:


CREATE TABLE author (
    id integer NOT NULL,
    firstname character varying(45) NOT NULL,
    lastname character varying(45) NOT NULL,
    middlename character varying(45)
)
WITH (fillfactor=80);


ALTER TABLE libdb.author OWNER TO postgres;

 TOC entry 188 (class 1259 OID 19258)-
 Dependencies: 189 7-

 Name: author_id_seq; Type: SEQUENCE; Schema: libdb; Owner: postgres
```

```
CREATE SEQUENCE author_id_seq
    START WITH 1
    INCREMENT BY 1
    NO MINVALUE
    NO MAXVALUE
    CACHE 1;


ALTER TABLE libdb.author_id_seq OWNER TO postgres;

 TOC entry 2031 (class 0 OID 0)-
 Dependencies: 188-

 Name: author_id_seq; Type: SEQUENCE OWNED BY; Schema: libdb; Owner: postgres


ALTER SEQUENCE author_id_seq OWNED BY author.id;


 TOC entry 190 (class 1259 OID 19266)-
 Dependencies: 7-

 Name: authorize; Type: TABLE; Schema: libdb; Owner: postgres; Tablespace:


CREATE TABLE authorize (
    author_id integer NOT NULL,
    book_id integer NOT NULL
)
WITH (fillfactor=80);


ALTER TABLE libdb.authorize OWNER TO postgres;

 TOC entry 181 (class 1259 OID 19183)-
 Dependencies: 7-

 Name: book; Type: TABLE; Schema: libdb; Owner: postgres; Tablespace:


CREATE TABLE book (
    id integer NOT NULL,
    isbn character varying(14),
    title character varying(100),
    poublisher_id integer,
    year date,
    pages smallint,
    price real,
    "desc" character varying(500),
    sub_category_id integer,
    copy_count smallint
```

```
)
WITH (fillfactor=70);


ALTER TABLE libdb.book OWNER TO postgres;

 TOC entry 187 (class 1259 OID 19240)-
 Dependencies: 7-

 Name: book_borrow; Type: TABLE; Schema: libdb; Owner: postgres; Tablespace:


CREATE TABLE book_borrow (
    id integer NOT NULL,
    user_id integer NOT NULL,
    book_copy_id integer NOT NULL,
    borrow_date date NOT NULL,
    return_date_expected date NOT NULL,
    return_date_actual date
)
WITH (fillfactor=70);


ALTER TABLE libdb.book_borrow OWNER TO postgres;

 TOC entry 186 (class 1259 OID 19238)-
 Dependencies: 187 7-

 Name: book_borrow_id_seq; Type: SEQUENCE; Schema: libdb; Owner: postgres


CREATE SEQUENCE book_borrow_id_seq
    START WITH 1
    INCREMENT BY 1
    NO MINVALUE
    NO MAXVALUE
    CACHE 1;


ALTER TABLE libdb.book_borrow_id_seq OWNER TO postgres;

 TOC entry 2032 (class 0 OID 0)-
 Dependencies: 186-

 Name: book_borrow_id_seq; Type: SEQUENCE OWNED BY; Schema: libdb; Owner:
postgres


ALTER SEQUENCE book_borrow_id_seq OWNED BY book_borrow.id;
```

```
 TOC entry 183 (class 1259 OID 19206)-
 Dependencies: 7-

 Name: book_copy; Type: TABLE; Schema: libdb; Owner: postgres; Tablespace:


CREATE TABLE book_copy (
    id integer NOT NULL,
    book_id integer,
    status smallint,
    bookshelf_id integer
)
WITH (fillfactor=60);


ALTER TABLE libdb.book_copy OWNER TO postgres;

 TOC entry 182 (class 1259 OID 19204)-
 Dependencies: 183 7-

 Name: book_copy_id_seq; Type: SEQUENCE; Schema: libdb; Owner: postgres


CREATE SEQUENCE book_copy_id_seq
    START WITH 1
    INCREMENT BY 1
    NO MINVALUE
    NO MAXVALUE
    CACHE 1;


ALTER TABLE libdb.book_copy_id_seq OWNER TO postgres;

 TOC entry 2033 (class 0 OID 0)-
 Dependencies: 182-

 Name: book_copy_id_seq; Type: SEQUENCE OWNED BY; Schema: libdb; Owner:
postgres


ALTER SEQUENCE book_copy_id_seq OWNED BY book_copy.id;


 TOC entry 180 (class 1259 OID 19181)-
 Dependencies: 7 181-

 Name: book_id_seq; Type: SEQUENCE; Schema: libdb; Owner: postgres


CREATE SEQUENCE book_id_seq
    START WITH 1
    INCREMENT BY 1
    NO MINVALUE
```

```
    NO MAXVALUE
    CACHE 1;


ALTER TABLE libdb.book_id_seq OWNER TO postgres;

 TOC entry 2034 (class 0 OID 0)-
 Dependencies: 180-

 Name: book_id_seq; Type: SEQUENCE OWNED BY; Schema: libdb; Owner: postgres


ALTER SEQUENCE book_id_seq OWNED BY book.id;


 TOC entry 173 (class 1259 OID 19139)-
 Dependencies: 7-

 Name: bookshelf; Type: TABLE; Schema: libdb; Owner: postgres; Tablespace:


CREATE TABLE bookshelf (
    id integer NOT NULL,
    label character varying(45),
    location_id integer NOT NULL
)
WITH (fillfactor=80);


ALTER TABLE libdb.bookshelf OWNER TO postgres;

 TOC entry 172 (class 1259 OID 19137)-
 Dependencies: 173 7-

 Name: bookshelf_id_seq; Type: SEQUENCE; Schema: libdb; Owner: postgres


CREATE SEQUENCE bookshelf_id_seq
    START WITH 1
    INCREMENT BY 1
    NO MINVALUE
    NO MAXVALUE
    CACHE 1;


ALTER TABLE libdb.bookshelf_id_seq OWNER TO postgres;

 TOC entry 2035 (class 0 OID 0)-
 Dependencies: 172-

 Name: bookshelf_id_seq; Type: SEQUENCE OWNED BY; Schema: libdb; Owner:
postgres
```

```
ALTER SEQUENCE bookshelf_id_seq OWNED BY bookshelf.id;


 TOC entry 169 (class 1259 OID 19117)-
 Dependencies: 7-

 Name: building; Type: TABLE; Schema: libdb; Owner: postgres; Tablespace:


CREATE TABLE building (
    id integer NOT NULL,
    name character varying(80) NOT NULL,
    address character varying(200) NOT NULL
)
WITH (fillfactor=90);


ALTER TABLE libdb.building OWNER TO postgres;

 TOC entry 168 (class 1259 OID 19115)-
 Dependencies: 7 169-

 Name: building_id_seq; Type: SEQUENCE; Schema: libdb; Owner: postgres


CREATE SEQUENCE building_id_seq
    START WITH 1
    INCREMENT BY 1
    NO MINVALUE
    NO MAXVALUE
    CACHE 1;


ALTER TABLE libdb.building_id_seq OWNER TO postgres;

 TOC entry 2036 (class 0 OID 0)-
 Dependencies: 168-

 Name: building_id_seq; Type: SEQUENCE OWNED BY; Schema: libdb; Owner: postgres


ALTER SEQUENCE building_id_seq OWNED BY building.id;


 TOC entry 163 (class 1259 OID 19084)-
 Dependencies: 7-

 Name: category; Type: TABLE; Schema: libdb; Owner: postgres; Tablespace:
```

```
CREATE TABLE category (
    id integer NOT NULL,
    name character varying(100) NOT NULL
)
WITH (fillfactor=80);


ALTER TABLE libdb.category OWNER TO postgres;

 TOC entry 162 (class 1259 OID 19082)-
 Dependencies: 163 7-

 Name: category_id_seq; Type: SEQUENCE; Schema: libdb; Owner: postgres


CREATE SEQUENCE category_id_seq
    START WITH 1
    INCREMENT BY 1
    NO MINVALUE
    NO MAXVALUE
    CACHE 1;


ALTER TABLE libdb.category_id_seq OWNER TO postgres;

 TOC entry 2037 (class 0 OID 0)-
 Dependencies: 162-

 Name: category_id_seq; Type: SEQUENCE OWNED BY; Schema: libdb; Owner: postgres


ALTER SEQUENCE category_id_seq OWNED BY category.id;


 TOC entry 171 (class 1259 OID 19125)-
 Dependencies: 7-

 Name: location; Type: TABLE; Schema: libdb; Owner: postgres; Tablespace:


CREATE TABLE location (
    id integer NOT NULL,
    floor integer,
    building_id integer NOT NULL
)
WITH (fillfactor=90);


ALTER TABLE libdb.location OWNER TO postgres;

 TOC entry 170 (class 1259 OID 19123)-
```

Dependencies: 7 171-

Name: location_id_seq; Type: SEQUENCE; Schema: libdb; Owner: postgres

```
CREATE SEQUENCE location_id_seq
    START WITH 1
    INCREMENT BY 1
    NO MINVALUE
    NO MAXVALUE
    CACHE 1;
```

```
ALTER TABLE libdb.location_id_seq OWNER TO postgres;
```

TOC entry 2038 (class 0 OID 0)-
Dependencies: 170-

Name: location_id_seq; Type: SEQUENCE OWNED BY; Schema: libdb; Owner: postgres

```
ALTER SEQUENCE location_id_seq OWNED BY location.id;
```

TOC entry 175 (class 1259 OID 19153)-
Dependencies: 7-

Name: occupation; Type: TABLE; Schema: libdb; Owner: postgres; Tablespace:

```
CREATE TABLE occupation (
    id integer NOT NULL,
    title character varying(45),
    borrow_limit smallint
)
WITH (fillfactor=80);
```

```
ALTER TABLE libdb.occupation OWNER TO postgres;
```

TOC entry 174 (class 1259 OID 19151)-
Dependencies: 175 7-

Name: occupation_id_seq; Type: SEQUENCE; Schema: libdb; Owner: postgres

```
CREATE SEQUENCE occupation_id_seq
    START WITH 1
    INCREMENT BY 1
    NO MINVALUE
    NO MAXVALUE
    CACHE 1;
```

```
ALTER TABLE libdb.occupation_id_seq OWNER TO postgres;

 TOC entry 2039 (class 0 OID 0)-
 Dependencies: 174-

 Name: occupation_id_seq; Type: SEQUENCE OWNED BY; Schema: libdb; Owner:
postgres


ALTER SEQUENCE occupation_id_seq OWNED BY occupation.id;


 TOC entry 191 (class 1259 OID 19282)-
 Dependencies: 7-

 Name: penalty; Type: TABLE; Schema: libdb; Owner: postgres; Tablespace:


CREATE TABLE penalty (
    user_id integer NOT NULL,
    issue_date date NOT NULL,
    clear_date date,
    amount real NOT NULL,
    book_borrow_id integer NOT NULL,
    penalty_reason_id integer NOT NULL
)
WITH (fillfactor=80);


ALTER TABLE libdb.penalty OWNER TO postgres;

 TOC entry 177 (class 1259 OID 19161)-
 Dependencies: 7-

 Name: penalty_reason; Type: TABLE; Schema: libdb; Owner: postgres; Tablespace:


CREATE TABLE penalty_reason (
    id integer NOT NULL,
    "desc" character varying(80) NOT NULL
)
WITH (fillfactor=90);


ALTER TABLE libdb.penalty_reason OWNER TO postgres;

 TOC entry 176 (class 1259 OID 19159)-
 Dependencies: 7 177-

 Name: penalty_reason_id_seq; Type: SEQUENCE; Schema: libdb; Owner: postgres
```

```
CREATE SEQUENCE penalty_reason_id_seq
    START WITH 1
    INCREMENT BY 1
    NO MINVALUE
    NO MAXVALUE
    CACHE 1;


ALTER TABLE libdb.penalty_reason_id_seq OWNER TO postgres;

 TOC entry 2040 (class 0 OID 0)-
 Dependencies: 176-

 Name: penalty_reason_id_seq; Type: SEQUENCE OWNED BY; Schema: libdb; Owner:
postgres


ALTER SEQUENCE penalty_reason_id_seq OWNED BY penalty_reason.id;


 TOC entry 167 (class 1259 OID 19107)-
 Dependencies: 7-

 Name: publisher; Type: TABLE; Schema: libdb; Owner: postgres; Tablespace:


CREATE TABLE publisher (
    id integer NOT NULL,
    name character varying(45) NOT NULL,
    email character varying(60),
    phone character varying(45)
)
WITH (fillfactor=80);


ALTER TABLE libdb.publisher OWNER TO postgres;

 TOC entry 166 (class 1259 OID 19105)-
 Dependencies: 167 7-

 Name: publisher_id_seq; Type: SEQUENCE; Schema: libdb; Owner: postgres


CREATE SEQUENCE publisher_id_seq
    START WITH 1
    INCREMENT BY 1
    NO MINVALUE
    NO MAXVALUE
    CACHE 1;
```

```
ALTER TABLE libdb.publisher_id_seq OWNER TO postgres;

 TOC entry 2041 (class 0 OID 0)-
 Dependencies: 166-

 Name: publisher_id_seq; Type: SEQUENCE OWNED BY; Schema: libdb; Owner:
postgres


ALTER SEQUENCE publisher_id_seq OWNED BY publisher.id;


 TOC entry 165 (class 1259 OID 19093)-
 Dependencies: 7-

 Name: sub_category; Type: TABLE; Schema: libdb; Owner: postgres; Tablespace:


CREATE TABLE sub_category (
    id integer NOT NULL,
    name character varying(100),
    category_id integer NOT NULL
)
WITH (fillfactor=80);


ALTER TABLE libdb.sub_category OWNER TO postgres;

 TOC entry 164 (class 1259 OID 19091)-
 Dependencies: 165 7-

 Name: sub_category_id_seq; Type: SEQUENCE; Schema: libdb; Owner: postgres


CREATE SEQUENCE sub_category_id_seq
    START WITH 1
    INCREMENT BY 1
    NO MINVALUE
    NO MAXVALUE
    CACHE 1;


ALTER TABLE libdb.sub_category_id_seq OWNER TO postgres;

 TOC entry 2042 (class 0 OID 0)-
 Dependencies: 164-

 Name: sub_category_id_seq; Type: SEQUENCE OWNED BY; Schema: libdb; Owner:
postgres


ALTER SEQUENCE sub_category_id_seq OWNED BY sub_category.id;
```

```
 TOC entry 179 (class 1259 OID 19169)-
 Dependencies: 7-

 Name: user; Type: TABLE; Schema: libdb; Owner: postgres; Tablespace:


CREATE TABLE "user" (
    id integer NOT NULL,
    lastname character varying(45) NOT NULL,
    firstname character varying(45) NOT NULL,
    middlename character varying(45),
    email character varying(60) NOT NULL,
    phone character varying(45),
    since date NOT NULL,
    occupation_id integer NOT NULL
)
WITH (fillfactor=70);


ALTER TABLE libdb."user" OWNER TO postgres;

 TOC entry 178 (class 1259 OID 19167)-
 Dependencies: 7 179-

 Name: user_id_seq; Type: SEQUENCE; Schema: libdb; Owner: postgres


CREATE SEQUENCE user_id_seq
    START WITH 1
    INCREMENT BY 1
    NO MINVALUE
    NO MAXVALUE
    CACHE 1;


ALTER TABLE libdb.user_id_seq OWNER TO postgres;

 TOC entry 2043 (class 0 OID 0)-
 Dependencies: 178-

 Name: user_id_seq; Type: SEQUENCE OWNED BY; Schema: libdb; Owner: postgres


ALTER SEQUENCE user_id_seq OWNED BY "user".id;


 TOC entry 185 (class 1259 OID 19226)-
 Dependencies: 7-

 Name: wishlist; Type: TABLE; Schema: libdb; Owner: postgres; Tablespace:
```

```
CREATE TABLE wishlist (
    id integer NOT NULL,
    isbn character varying(14),
    request_date date,
    status smallint,
    user_id integer NOT NULL
)
WITH (fillfactor=70);


ALTER TABLE libdb.wishlist OWNER TO postgres;

 TOC entry 184 (class 1259 OID 19224)-
 Dependencies: 185 7-

 Name: wishlist_id_seq; Type: SEQUENCE; Schema: libdb; Owner: postgres


CREATE SEQUENCE wishlist_id_seq
    START WITH 1
    INCREMENT BY 1
    NO MINVALUE
    NO MAXVALUE
    CACHE 1;


ALTER TABLE libdb.wishlist_id_seq OWNER TO postgres;

 TOC entry 2044 (class 0 OID 0)-
 Dependencies: 184-

 Name: wishlist_id_seq; Type: SEQUENCE OWNED BY; Schema: libdb; Owner: postgres


ALTER SEQUENCE wishlist_id_seq OWNED BY wishlist.id;


 TOC entry 1848 (class 2604 OID 19263)-
 Dependencies: 188 189 189-

 Name: id; Type: DEFAULT; Schema: libdb; Owner: postgres


ALTER TABLE ONLY author ALTER COLUMN id SET DEFAULT
nextval('author_id_seq'::regclass);


 TOC entry 1844 (class 2604 OID 19186)-
 Dependencies: 180 181 181-
```

Name: id; Type: DEFAULT; Schema: libdb; Owner: postgres

```
ALTER TABLE ONLY book ALTER COLUMN id SET DEFAULT
nextval('book_id_seq'::regclass);
```

TOC entry 1847 (class 2604 OID 19243)-
Dependencies: 187 186 187-

Name: id; Type: DEFAULT; Schema: libdb; Owner: postgres

```
ALTER TABLE ONLY book_borrow ALTER COLUMN id SET DEFAULT
nextval('book_borrow_id_seq'::regclass);
```

TOC entry 1845 (class 2604 OID 19209)-
Dependencies: 183 182 183-

Name: id; Type: DEFAULT; Schema: libdb; Owner: postgres

```
ALTER TABLE ONLY book_copy ALTER COLUMN id SET DEFAULT
nextval('book_copy_id_seq'::regclass);
```

TOC entry 1840 (class 2604 OID 19142)-
Dependencies: 173 172 173-

Name: id; Type: DEFAULT; Schema: libdb; Owner: postgres

```
ALTER TABLE ONLY bookshelf ALTER COLUMN id SET DEFAULT
nextval('bookshelf_id_seq'::regclass);
```

TOC entry 1838 (class 2604 OID 19120)-
Dependencies: 169 168 169-

Name: id; Type: DEFAULT; Schema: libdb; Owner: postgres

```
ALTER TABLE ONLY building ALTER COLUMN id SET DEFAULT
nextval('building_id_seq'::regclass);
```

TOC entry 1835 (class 2604 OID 19087)-
Dependencies: 162 163 163-

Name: id; Type: DEFAULT; Schema: libdb; Owner: postgres

ALTER TABLE ONLY category ALTER COLUMN id SET DEFAULT
nextval('category_id_seq'::regclass);

TOC entry 1839 (class 2604 OID 19128)-
Dependencies: 171 170 171-

Name: id; Type: DEFAULT; Schema: libdb; Owner: postgres

ALTER TABLE ONLY location ALTER COLUMN id SET DEFAULT
nextval('location_id_seq'::regclass);

TOC entry 1841 (class 2604 OID 19156)-
Dependencies: 174 175 175-

Name: id; Type: DEFAULT; Schema: libdb; Owner: postgres

ALTER TABLE ONLY occupation ALTER COLUMN id SET DEFAULT
nextval('occupation_id_seq'::regclass);

TOC entry 1842 (class 2604 OID 19164)-
Dependencies: 176 177 177-

Name: id; Type: DEFAULT; Schema: libdb; Owner: postgres

ALTER TABLE ONLY penalty_reason ALTER COLUMN id SET DEFAULT
nextval('penalty_reason_id_seq'::regclass);

TOC entry 1837 (class 2604 OID 19110)-
Dependencies: 167 166 167-

Name: id; Type: DEFAULT; Schema: libdb; Owner: postgres

ALTER TABLE ONLY publisher ALTER COLUMN id SET DEFAULT
nextval('publisher_id_seq'::regclass);

TOC entry 1836 (class 2604 OID 19096)-
Dependencies: 164 165 165-

Name: id; Type: DEFAULT; Schema: libdb; Owner: postgres

ALTER TABLE ONLY sub_category ALTER COLUMN id SET DEFAULT

nextval('sub_category_id_seq'::regclass);

TOC entry 1843 (class 2604 OID 19172)-
Dependencies: 179 178 179-

Name: id; Type: DEFAULT; Schema: libdb; Owner: postgres

ALTER TABLE ONLY "user" ALTER COLUMN id SET DEFAULT
nextval('user_id_seq'::regclass);

TOC entry 1846 (class 2604 OID 19229)-
Dependencies: 184 185 185-

Name: id; Type: DEFAULT; Schema: libdb; Owner: postgres

ALTER TABLE ONLY wishlist ALTER COLUMN id SET DEFAULT
nextval('wishlist_id_seq'::regclass);

TOC entry 1898 (class 2606 OID 19265)-
Dependencies: 189 189 2028-
Name: author_pkey; Type: CONSTRAINT; Schema: libdb; Owner: postgres; Tablespace:-

ALTER TABLE ONLY author
    ADD CONSTRAINT author_pkey PRIMARY KEY (id);

TOC entry 1903 (class 2606 OID 19270)-
Dependencies: 190 190 190 2028-
Name: authorize_pkey; Type: CONSTRAINT; Schema: libdb; Owner: postgres;-

Tablespace:

ALTER TABLE ONLY authorize
    ADD CONSTRAINT authorize_pkey PRIMARY KEY (author_id, book_id);

TOC entry 1894 (class 2606 OID 19245)-
Dependencies: 187 187 2028-
Name: book_borrow_pkey; Type: CONSTRAINT; Schema: libdb; Owner: postgres;-

Tablespace:

ALTER TABLE ONLY book_borrow

```
    ADD CONSTRAINT book_borrow_pkey PRIMARY KEY (id);


 TOC entry 1885 (class 2606 OID 19211)-
 Dependencies: 183 183 2028-

 Name: book_copy_pkey; Type: CONSTRAINT; Schema: libdb; Owner: postgres;
Tablespace:


ALTER TABLE ONLY book_copy
    ADD CONSTRAINT book_copy_pkey PRIMARY KEY (id);


 TOC entry 1878 (class 2606 OID 19191)-
 Dependencies: 181 181 2028-
 Name: book_pkey; Type: CONSTRAINT; Schema: libdb; Owner: postgres; Tablespace:-


ALTER TABLE ONLY book
    ADD CONSTRAINT book_pkey PRIMARY KEY (id);


 TOC entry 1865 (class 2606 OID 19144)-
 Dependencies: 173 173 2028-
 Name: bookshelf_pkey; Type: CONSTRAINT; Schema: libdb; Owner: postgres;-

Tablespace:


ALTER TABLE ONLY bookshelf
    ADD CONSTRAINT bookshelf_pkey PRIMARY KEY (id);


 TOC entry 1860 (class 2606 OID 19122)-
 Dependencies: 169 169 2028-
 Name: building_pkey; Type: CONSTRAINT; Schema: libdb; Owner: postgres; Tablespace:-


ALTER TABLE ONLY building
    ADD CONSTRAINT building_pkey PRIMARY KEY (id);


 TOC entry 1850 (class 2606 OID 19089)-
 Dependencies: 163 163 2028-

 Name: category_pkey; Type: CONSTRAINT; Schema: libdb; Owner: postgres;
Tablespace:
```

```
ALTER TABLE ONLY category
    ADD CONSTRAINT category_pkey PRIMARY KEY (id);


 TOC entry 1863 (class 2606 OID 19130)-
 Dependencies: 171 171 2028-
 Name: location_pkey; Type: CONSTRAINT; Schema: libdb; Owner: postgres; Tablespace:-


ALTER TABLE ONLY location
    ADD CONSTRAINT location_pkey PRIMARY KEY (id);


 TOC entry 1868 (class 2606 OID 19158)-
 Dependencies: 175 175 2028-

 Name: occupation_pkey; Type: CONSTRAINT; Schema: libdb; Owner: postgres;
Tablespace:


ALTER TABLE ONLY occupation
    ADD CONSTRAINT occupation_pkey PRIMARY KEY (id);


 TOC entry 1909 (class 2606 OID 19286)-
 Dependencies: 191 191 2028-
 Name: penalty_pkey; Type: CONSTRAINT; Schema: libdb; Owner: postgres; Tablespace:-


ALTER TABLE ONLY penalty
    ADD CONSTRAINT penalty_pkey PRIMARY KEY (book_borrow_id);


 TOC entry 1870 (class 2606 OID 19166)-
 Dependencies: 177 177 2028-
 Name: penalty_reason_pkey; Type: CONSTRAINT; Schema: libdb; Owner: postgres;-

Tablespace:


ALTER TABLE ONLY penalty_reason
    ADD CONSTRAINT penalty_reason_pkey PRIMARY KEY (id);


 TOC entry 1858 (class 2606 OID 19112)-
 Dependencies: 167 167 2028-
 Name: publisher_pkey; Type: CONSTRAINT; Schema: libdb; Owner: postgres;-

Tablespace:
```

```
ALTER TABLE ONLY publisher
    ADD CONSTRAINT publisher_pkey PRIMARY KEY (id);


 TOC entry 1854 (class 2606 OID 19098)-
 Dependencies: 165 165 2028-

 Name: sub_category_pkey; Type: CONSTRAINT; Schema: libdb; Owner: postgres;
Tablespace:


ALTER TABLE ONLY sub_category
    ADD CONSTRAINT sub_category_pkey PRIMARY KEY (id);


 TOC entry 1875 (class 2606 OID 19174)-
 Dependencies: 179 179 2028-
 Name: user_pkey; Type: CONSTRAINT; Schema: libdb; Owner: postgres; Tablespace:-


ALTER TABLE ONLY "user"
    ADD CONSTRAINT user_pkey PRIMARY KEY (id);


 TOC entry 1892 (class 2606 OID 19231)-
 Dependencies: 185 185 2028-

 Name: wishlist_pkey; Type: CONSTRAINT; Schema: libdb; Owner: postgres; Tablespace:


ALTER TABLE ONLY wishlist
    ADD CONSTRAINT wishlist_pkey PRIMARY KEY (id);


 TOC entry 1900 (class 1259 OID 19307)-
 Dependencies: 190 2028-
 Name: authorize_author_id_idx; Type: INDEX; Schema: libdb; Owner: postgres;-

Tablespace:


CREATE INDEX authorize_author_id_idx ON authorize USING hash (author_id);


 TOC entry 1901 (class 1259 OID 19308)-
 Dependencies: 190 2028-
 Name: authorize_book_id_idx; Type: INDEX; Schema: libdb; Owner: postgres; Tablespace:-
```

CREATE INDEX authorize_book_id_idx ON authorize USING hash (book_id);

TOC entry 1883 (class 1259 OID 19312)-
Dependencies: 183 2028-

Name: book_copy_book_id_idx; Type: INDEX; Schema: libdb; Owner: postgres;
Tablespace:

CREATE INDEX book_copy_book_id_idx ON book_copy USING hash (book_id);

TOC entry 1886 (class 1259 OID 19313)-
Dependencies: 183 2028-

Name: book_copy_status_idx; Type: INDEX; Schema: libdb; Owner: postgres; Tablespace:

CREATE INDEX book_copy_status_idx ON book_copy USING btree (status);

TOC entry 1876 (class 1259 OID 19309)-
Dependencies: 181 2028-

Name: book_isbn_idx; Type: INDEX; Schema: libdb; Owner: postgres; Tablespace:

CREATE INDEX book_isbn_idx ON book USING btree (isbn);

TOC entry 1879 (class 1259 OID 19311)-
Dependencies: 181 2028-
Name: book_publisher_idx; Type: INDEX; Schema: libdb; Owner: postgres; Tablespace:-

CREATE INDEX book_publisher_idx ON book USING hash (poublisher_id);

TOC entry 1880 (class 1259 OID 19310)-
Dependencies: 181 2028-

Name: book_title_idx; Type: INDEX; Schema: libdb; Owner: postgres; Tablespace:

CREATE INDEX book_title_idx ON book USING btree (title);

TOC entry 1855 (class 1259 OID 19113)-
Dependencies: 167 2028-

Name: email_UNIQUE; Type: INDEX; Schema: libdb; Owner: postgres; Tablespace:-

CREATE UNIQUE INDEX "email_UNIQUE" ON publisher USING btree (email);

TOC entry 1904 (class 1259 OID 19281)-
Dependencies: 190 2028-
Name: fk_authorize_book1_idx; Type: INDEX; Schema: libdb; Owner: postgres;-

Tablespace:

CREATE INDEX fk_authorize_book1_idx ON authorize USING btree (book_id);

TOC entry 1895 (class 1259 OID 19257)-
Dependencies: 187 2028-
Name: fk_book_borrow_book_copy1_idx; Type: INDEX; Schema: libdb; Owner: postgres;-

Tablespace:

CREATE INDEX fk_book_borrow_book_copy1_idx ON book_borrow USING btree
(book_copy_id);

TOC entry 1896 (class 1259 OID 19256)-
Dependencies: 187 2028-
Name: fk_book_borrow_user1_idx; Type: INDEX; Schema: libdb; Owner: postgres;-

Tablespace:

CREATE INDEX fk_book_borrow_user1_idx ON book_borrow USING btree (user_id);

TOC entry 1887 (class 1259 OID 19222)-
Dependencies: 183 2028-
Name: fk_book_copy_book1_idx; Type: INDEX; Schema: libdb; Owner: postgres;-

Tablespace:

CREATE INDEX fk_book_copy_book1_idx ON book_copy USING btree (book_id);

TOC entry 1888 (class 1259 OID 19223)-
Dependencies: 183 2028-
Name: fk_book_copy_bookshelf1_idx; Type: INDEX; Schema: libdb; Owner: postgres;-

Tablespace:

CREATE INDEX fk_book_copy_bookshelf1_idx ON book_copy USING btree (bookshelf_id);

TOC entry 1881 (class 1259 OID 19202)-
Dependencies: 181 2028-
Name: fk_book_publisher1_idx; Type: INDEX; Schema: libdb; Owner: postgres;-

Tablespace:

CREATE INDEX fk_book_publisher1_idx ON book USING btree (poublisher_id);

TOC entry 1882 (class 1259 OID 19203)-
Dependencies: 181 2028-
Name: fk_book_sub_category1_idx; Type: INDEX; Schema: libdb; Owner: postgres;-

Tablespace:

CREATE INDEX fk_book_sub_category1_idx ON book USING btree (sub_category_id);

TOC entry 1866 (class 1259 OID 19150)-
Dependencies: 173 2028-
Name: fk_bookshelf_location1_idx; Type: INDEX; Schema: libdb; Owner: postgres;-

Tablespace:

CREATE INDEX fk_bookshelf_location1_idx ON bookshelf USING btree (location_id);

TOC entry 1861 (class 1259 OID 19136)-
Dependencies: 171 2028-
Name: fk_location_building1_idx; Type: INDEX; Schema: libdb; Owner: postgres;-

Tablespace:

CREATE INDEX fk_location_building1_idx ON location USING btree (building_id);

TOC entry 1905 (class 1259 OID 19304)-
Dependencies: 191 2028-

Name: fk_penalty_book_copy1_idx; Type: INDEX; Schema: libdb; Owner: postgres;
Tablespace:

CREATE INDEX fk_penalty_book_copy1_idx ON penalty USING btree (book_borrow_id);

TOC entry 1906 (class 1259 OID 19302)-
Dependencies: 191 2028-
Name: fk_penalty_penalty_reason1_idx; Type: INDEX; Schema: libdb; Owner: postgres;-

Tablespace:

CREATE INDEX fk_penalty_penalty_reason1_idx ON penalty USING btree
(penalty_reason_id);

TOC entry 1907 (class 1259 OID 19303)-
Dependencies: 191 2028-
Name: fk_penalty_user1_idx; Type: INDEX; Schema: libdb; Owner: postgres; Tablespace:-

CREATE INDEX fk_penalty_user1_idx ON penalty USING btree (user_id);

TOC entry 1852 (class 1259 OID 19104)-
Dependencies: 165 2028-

Name: fk_sub_category_category_idx; Type: INDEX; Schema: libdb; Owner: postgres;
Tablespace:

CREATE INDEX fk_sub_category_category_idx ON sub_category USING btree
(category_id);

TOC entry 1871 (class 1259 OID 19180)-
Dependencies: 179 2028-
Name: fk_user_occupation1_idx; Type: INDEX; Schema: libdb; Owner: postgres;-

Tablespace:

CREATE INDEX fk_user_occupation1_idx ON "user" USING btree (occupation_id);

TOC entry 1889 (class 1259 OID 19237)-
Dependencies: 185 2028-

Name: fk_wishlist_user1_idx; Type: INDEX; Schema: libdb; Owner: postgres; Tablespace:

CREATE INDEX fk_wishlist_user1_idx ON wishlist USING btree (user_id);

TOC entry 1899 (class 1259 OID 19306)-
Dependencies: 189 2028-

Name: id_idx; Type: INDEX; Schema: libdb; Owner: postgres; Tablespace:

```
CREATE INDEX id_idx ON author USING btree (id);
```

TOC entry 1851 (class 1259 OID 19090)-
Dependencies: 163 2028-
Name: name_UNIQUE; Type: INDEX; Schema: libdb; Owner: postgres; Tablespace:-

```
CREATE UNIQUE INDEX "name_UNIQUE" ON category USING btree (name);
```

TOC entry 1856 (class 1259 OID 19114)-
Dependencies: 167 2028-
Name: phone_UNIQUE; Type: INDEX; Schema: libdb; Owner: postgres; Tablespace:-

```
CREATE UNIQUE INDEX "phone_UNIQUE" ON publisher USING btree (phone);
```

TOC entry 1872 (class 1259 OID 19317)-
Dependencies: 179 2028-
Name: user_firstname_idx; Type: INDEX; Schema: libdb; Owner: postgres; Tablespace:-

```
CREATE INDEX user_firstname_idx ON "user" USING btree (firstname);
```

TOC entry 1873 (class 1259 OID 19318)-
Dependencies: 179 2028-

Name: user_lastname_idx; Type: INDEX; Schema: libdb; Owner: postgres; Tablespace:

```
CREATE INDEX user_lastname_idx ON "user" USING btree (lastname);
```

TOC entry 1890 (class 1259 OID 19319)-
Dependencies: 185 2028-

Name: wishlist_isbn_idx; Type: INDEX; Schema: libdb; Owner: postgres; Tablespace:

CREATE INDEX wishlist_isbn_idx ON wishlist USING hash (isbn);

TOC entry 1921 (class 2606 OID 19271)-
Dependencies: 189 190 1897 2028-
Name: fk_authorize_author1; Type: FK CONSTRAINT; Schema: libdb; Owner: postgres-

ALTER TABLE ONLY authorize
    ADD CONSTRAINT fk_authorize_author1 FOREIGN KEY (author_id) REFERENCES
author(id) ON UPDATE CASCADE ON DELETE CASCADE;

TOC entry 1922 (class 2606 OID 19276)-
Dependencies: 190 1877 181 2028-
Name: fk_authorize_book1; Type: FK CONSTRAINT; Schema: libdb; Owner: postgres-

ALTER TABLE ONLY authorize
    ADD CONSTRAINT fk_authorize_book1 FOREIGN KEY (book_id) REFERENCES
book(id) ON UPDATE CASCADE ON DELETE CASCADE;

TOC entry 1920 (class 2606 OID 19251)-
Dependencies: 1884 187 183 2028-
Name: fk_book_borrow_book_copy1; Type: FK CONSTRAINT; Schema: libdb; Owner:-

postgres

ALTER TABLE ONLY book_borrow
    ADD CONSTRAINT fk_book_borrow_book_copy1 FOREIGN KEY (book_copy_id)
REFERENCES book_copy(id) ON UPDATE CASCADE ON DELETE CASCADE;

TOC entry 1919 (class 2606 OID 19246)-
Dependencies: 179 187 1874 2028-
Name: fk_book_borrow_user1; Type: FK CONSTRAINT; Schema: libdb; Owner: postgres-

ALTER TABLE ONLY book_borrow
    ADD CONSTRAINT fk_book_borrow_user1 FOREIGN KEY (user_id) REFERENCES
"user"(id) ON UPDATE CASCADE ON DELETE CASCADE;

TOC entry 1916 (class 2606 OID 19212)-
Dependencies: 181 183 1877 2028-
Name: fk_book_copy_book1; Type: FK CONSTRAINT; Schema: libdb; Owner: postgres-

ALTER TABLE ONLY book_copy
    ADD CONSTRAINT fk_book_copy_book1 FOREIGN KEY (book_id) REFERENCES
book(id) ON UPDATE CASCADE ON DELETE CASCADE;

 TOC entry 1917 (class 2606 OID 19217)-
 Dependencies: 1864 183 173 2028-
 Name: fk_book_copy_bookshelf1; Type: FK CONSTRAINT; Schema: libdb; Owner:-

postgres

ALTER TABLE ONLY book_copy
    ADD CONSTRAINT fk_book_copy_bookshelf1 FOREIGN KEY (bookshelf_id)
REFERENCES bookshelf(id) ON UPDATE CASCADE ON DELETE CASCADE;

 TOC entry 1914 (class 2606 OID 19192)-
 Dependencies: 181 1857 167 2028-
 Name: fk_book_publisher1; Type: FK CONSTRAINT; Schema: libdb; Owner: postgres-

ALTER TABLE ONLY book
    ADD CONSTRAINT fk_book_publisher1 FOREIGN KEY (poublisher_id) REFERENCES
publisher(id) ON UPDATE CASCADE ON DELETE CASCADE;

 TOC entry 1915 (class 2606 OID 19197)-
 Dependencies: 1853 181 165 2028-
 Name: fk_book_sub_category1; Type: FK CONSTRAINT; Schema: libdb; Owner: postgres-

ALTER TABLE ONLY book
    ADD CONSTRAINT fk_book_sub_category1 FOREIGN KEY (sub_category_id)
REFERENCES sub_category(id) ON UPDATE CASCADE ON DELETE CASCADE;

 TOC entry 1912 (class 2606 OID 19145)-
 Dependencies: 171 1862 173 2028-
 Name: fk_bookshelf_location1; Type: FK CONSTRAINT; Schema: libdb; Owner: postgres-

ALTER TABLE ONLY bookshelf
    ADD CONSTRAINT fk_bookshelf_location1 FOREIGN KEY (location_id) REFERENCES
location(id) ON UPDATE CASCADE ON DELETE CASCADE;

TOC entry 1911 (class 2606 OID 19131)-
Dependencies: 1859 171 169 2028-
Name: fk_location_building1; Type: FK CONSTRAINT; Schema: libdb; Owner: postgres-

ALTER TABLE ONLY location
    ADD CONSTRAINT fk_location_building1 FOREIGN KEY (building_id) REFERENCES building(id) ON UPDATE CASCADE ON DELETE CASCADE;

TOC entry 1925 (class 2606 OID 19297)-
Dependencies: 187 191 1893 2028-
Name: fk_penalty_book_borrow_id; Type: FK CONSTRAINT; Schema: libdb; Owner:-

postgres

ALTER TABLE ONLY penalty
    ADD CONSTRAINT fk_penalty_book_borrow_id FOREIGN KEY (book_borrow_id) REFERENCES book_borrow(id) ON UPDATE CASCADE ON DELETE CASCADE;

TOC entry 1923 (class 2606 OID 19287)-
Dependencies: 1869 177 191 2028-
Name: fk_penalty_penalty_reason1; Type: FK CONSTRAINT; Schema: libdb; Owner:-

postgres

ALTER TABLE ONLY penalty
    ADD CONSTRAINT fk_penalty_penalty_reason1 FOREIGN KEY (penalty_reason_id) REFERENCES penalty_reason(id) ON UPDATE CASCADE ON DELETE CASCADE;

TOC entry 1924 (class 2606 OID 19292)-
Dependencies: 191 1874 179 2028-
Name: fk_penalty_user1; Type: FK CONSTRAINT; Schema: libdb; Owner: postgres-

ALTER TABLE ONLY penalty
    ADD CONSTRAINT fk_penalty_user1 FOREIGN KEY (user_id) REFERENCES "user"(id) ON UPDATE CASCADE ON DELETE CASCADE;

TOC entry 1910 (class 2606 OID 19099)-
Dependencies: 1849 163 165 2028-
Name: fk_sub_category_category; Type: FK CONSTRAINT; Schema: libdb; Owner:-

postgres

```
ALTER TABLE ONLY sub_category
    ADD CONSTRAINT fk_sub_category_category FOREIGN KEY (category_id)
REFERENCES category(id) ON UPDATE CASCADE ON DELETE CASCADE;


 TOC entry 1913 (class 2606 OID 19175)-
 Dependencies: 1867 175 179 2028-
 Name: fk_user_occupation1; Type: FK CONSTRAINT; Schema: libdb; Owner: postgres-


ALTER TABLE ONLY "user"
    ADD CONSTRAINT fk_user_occupation1 FOREIGN KEY (occupation_id) REFERENCES
occupation(id) ON UPDATE CASCADE ON DELETE CASCADE;


 TOC entry 1918 (class 2606 OID 19232)-
 Dependencies: 1874 185 179 2028-

 Name: fk_wishlist_user1; Type: FK CONSTRAINT; Schema: libdb; Owner: postgres


ALTER TABLE ONLY wishlist
    ADD CONSTRAINT fk_wishlist_user1 FOREIGN KEY (user_id) REFERENCES "user"(id)
ON UPDATE CASCADE ON DELETE CASCADE;

 Completed on 20140309 22:43:15-


 PostgreSQL database dump complete
```

Data population

To populate test data, we used both of copy and random data generation.

```
--get_random_number--set search_path to libdb;
SELECT get_random_number(1, 100);

CREATE OR REPLACE FUNCTION get_random_number(INTEGER, INTEGER) RETURNS
INTEGER AS $$
DECLARE
    start_int ALIAS FOR $1;
    end_int ALIAS FOR $2;
BEGIN
    RETURN trunc(random() * (end_intstart_int) + start_int);
END;
$$ LANGUAGE 'plpgsql' STRICT;


--get_random_string--set search_path to libdb;
```

```
CREATE OR REPLACE FUNCTION get_random_string(length INTEGER)
RETURNS TEXT
LANGUAGE PLPGSQL
AS $$
DECLARE
   possible_chars TEXT := '0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ';
   output TEXT := '';
   i INT4;
BEGIN

   FOR i IN 1..length LOOP
      output := output || substr(possible_chars, get_random_number(1,
length(possible_chars)), 1);
   END LOOP;

   RETURN output;
END;
$$;


--populate table user  from csv--copy customers (firstname, lastname, email, phone, birthdate)
to 'D:\study\repo\neu_cs5200_db_classproject\dvd.customers.csv'
with csv header ;

set search_path to libdb;
copy "user" (firstname, lastname, email, phone, since,  occupation_id )
from 'D:\study\repo\neu_cs5200_db_classproject\dvd.customers.csv'
with csv header;

select * from "user"

--populate table wishlist--set search_path to libdb;

insert into wishlist (isbn,request_date, status, user_id)
select  get_random_string(14) as isbn,
now()::date  generate_series(1,30) as request_date, get_random_number(0,2) as status,
get_random_number( (select min(id) from "user") ,(select max(id) from "user") ) as user_id;

select * from wishlist;

--populate table category from csv--
set search_path to libdb;
copy category (name)
from 'D:\study\repo\neu_cs5200_db_classproject\class_proj  category.csv'
with csv header;

select * from category;


--populate table sub_category from csv--set search_path to libdb;
copy sub_category (name, category_id)
from 'D:\study\repo\neu_cs5200_db_classproject\class_proj  sub_category.csv'
```

```
with csv header;

select * from sub_category;

--populate table publisher--set search_path to libdb;
insert into publisher (name,email)
select  get_random_string(7) as name, get_random_string(7) || '@gmail.com' as email
from generate_series(1,10);

select * from publisher;

--populate table book--set search_path to libdb;
insert into book (isbn,title, publisher_id, year, pages, price , "desc", sub_category_id,
copy_count)
select  get_random_string(14) as isbn, get_random_string(2) || ' ' || get_random_string(7) as
title,
get_random_number( (select min(id) from "publisher") ,(select max(id) from "publisher") ) as
publisher_id,
now()::date  get_random_number(0, 5000) as  "year",
get_random_number(100, 200) as pages,
get_random_number(10, 100) as price,
get_random_string(get_random_number(100, 400)) as desc,
get_random_number( (select min(id) from "sub_category") ,(select max(id) from
"sub_category") ) as sub_category_id,
5 as copy_count
from generate_series (1,30);

select * from book;

--populate table book_copy--set search_path to libdb;

copy book_copy (book_id, status, bookshelf_id)
from 'D:\study\repo\neu_cs5200_db_classproject\book_copy.csv'
with csv header;

select * from book_copy;



--populate table occupation from csv--set search_path to libdb, public;
 data population for occupation table.-

copy occupation (title, borrow_limit) from
'D:\study\repo\neu_cs5200_db_classproject\class_proj  occupation.csv'
with csv header;

select * from occupation;


--populate table book_borrow --set search_path to libdb;

insert into book_borrow (user_id, book_copy_id, borrow_date, return_date_expected,
```

```
return_date_actual)
select user_id as a , book_copy_id as b, borrow_date as c , borrow_date + 30 as d,
borrow_date + 30 + get_random_number(10, 10) as e
from
(select get_random_number( (select min(id) from "user") ,(select max(id) from "user") ) as
user_id ,
get_random_number( (select min(id) from "book_copy") ,(select max(id) from "book_copy") )
as book_copy_id,
now()::date  get_random_number(0,365)  as borrow_date
from generate_series (1,30)) as base;

select * from book_borrow where return_date_actual > return_date_expected;



--creat csv file from dvdsales.customers--set search_path to dvdsales;

copy customers (firstname, lastname)
to 'D:\study\repo\neu_cs5200_db_classproject\dvd.customers.csv'
with csv header;


--populate table author--set search_path to libdb;

copy author(firstname, lastname)
from 'D:\study\repo\neu_cs5200_db_classproject\author.csv'
with CSV Header;

update author
set middlename = get_random_string(1)
from generate_series(1,10);

select * from author

--populate table bookshelf--set search_path to libdb;
insert into bookshelf (label, location_id)
select 'label'||get_random_string(5) as label, get_random_number(1,100) as location_id
from generate_series(1,10);

select * from bookshelf

--populate table building --set search_path to libdb;
copy building (id, name, address)
from 'C:\Users\Public\building.csv'
with CSV Header

select * from building

--populate table location --set search_path to libdb;

insert into location(floor,building_id)
select get_random_number(1,4) as floor, get_random_number(23,42) as building_id
```

```
from generate_series(1,100)

select * from location

--populate table penalty_reason--set search_path to libdb;

copy penalty_reason (id, desc)
from 'C:\Users\Public\penalty_reason'
with csv header;

--populate table penalty--set search_path to libdb;

insert into penalty(user_id,issue_date, clear_date, amount, book_borrow_id,
penalty_reason_id)
select get_random_number(1,10000) as user_id,
      now()::date  get_random_number(200,365) as issue_date,
      now()::date  get_random_number(0,200) as clear_date,
      get_random_number(0,100) as amount,
      get_random_number( (select min(id) from "book_borrow") ,(select max(id) from
"book_borrow") ) as book_borrow_id,
      get_random_number(1,4) as penalty_reason_id
from generate_series(1,100);

select * from penalty;


--populate table authorize--set search_path to libdb;

insert into authorize (author_id, book_id )
select
generate_series( (select min(id) from "author") ,(select max(id) from "author") ) ,
generate_series( (select min(id) from "book") ,(select max(id) from "book") )

select * from authorize;
```

Statistics

Below are table names and tuple counts.

Table Name	Live tuples
author	9
authorize	90
book	30
book_borrow	60
book_copy	150
bookshelf	10
building	20
category	6
location	100
occupation	3
penalty	60
penalty_reason	4
publisher	10
sub_category	65
user	17
wishlist	30

Two basic helper functions.

--get_random_string(integer)

```
CREATE OR REPLACE FUNCTION libdb.get_random_string(length integer)
RETURNS text AS
$BODY$
DECLARE
possible_chars TEXT := '0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ';
output TEXT := '';
i INT4;
BEGIN
FOR i IN 1..length LOOP
output := output || substr(possible_chars, get_random_number(1, length(possible_chars)), 1);
END LOOP;
RETURN output;
END;
$BODY$
LANGUAGE plpgsql VOLATILE
COST 100;
ALTER FUNCTION libdb.get_random_string(integer)
OWNER TO postgres;
```

--get_random_number(integer, integer)

```
CREATE OR REPLACE FUNCTION libdb.get_random_number(integer, integer)
RETURNS integer AS
$BODY$
DECLARE
start_int ALIAS FOR $1;
end_int ALIAS FOR $2;
BEGIN
RETURN trunc(random() * (end_int-start_int) + start_int);
END;
$BODY$
LANGUAGE plpgsql VOLATILE STRICT
COST 100;
ALTER FUNCTION libdb.get_random_number(integer, integer)
OWNER TO postgres;
```

Application for library users.

Application for library users.

--Search books.ISBN

```
CREATE OR REPLACE  FUNCTION search_isbn(_isbn varchar) RETURNS TABLE
(isbn varchar,title varchar, year int, pages int, des varchar, price real,
copy_count int, pub_name varchar) AS $$
DECLARE
BEGIN
RETURN QUERY
select B.isbn, B.title, date_part('year',B.year)::integer , B.pages::integer,
B.desc, B.price, B.copy_count::integer, P.name
from book B
join Publisher P on P.id = B.publisher_id
where B.isbn = _isbn;
END;
$$ LANGUAGE plpgsql;
--select  * from search_isbn('XDG2G8FV1TMWL0');
```

--Search books. Author or Title

```
CREATE OR REPLACE  FUNCTION search_author_title(varchar) RETURNS TABLE
(isbn varchar,title varchar, year int, pages int, des varchar, price real,
copy_count int, pub_name varchar) AS $$
DECLARE
_search_str varchar := '%' || $1 || '%';
BEGIN
RETURN QUERY
select B.isbn, B.title, date_part('year',B.year)::integer , B.pages::integer,
B.desc, B.price, B.copy_count::integer, P.name
from book B
join Publisher P on P.id = B.publisher_id
join authorize AZ on AZ.book_id = B.id
join author A on AZ.author_id = A.id
where
(B.title like _search_str)
or
(A.firstname like _search_str
or A.lastname like _search_str
or A.middlename like _search_str);
END;
$$ LANGUAGE plpgsql;
--select  * from search_autor_title('5P');
```

--return a book

```
CREATE OR REPLACE  FUNCTION return_a_book(_userid integer, _book_copy_id integer) RETURNS
void AS $$
DECLARE
BEGIN
update book_borrow BB
set return_date_actual = now()::date
where BB.user_id = _userid
and BB.book_copy_id = _book_copy_id
and BB.return_date_actual is null;
RETURN;
END;
$$ LANGUAGE plpgsql;
select return_a_book(12, 509);
```

-- get user borrow limit

```
CREATE OR REPLACE  FUNCTION get_borrow_limit(_userid integer) RETURNS integer AS $$
DECLARE
_limit integer;
_count integer;
BEGIN
select count(*)
from libdb.user U
where U.id = _userid
into _count;

---no such a user.
IF _count = 0 THEN
RAISE EXCEPTION 'no such a userid:% ', _userid;
END IF;
select borrow_limit
from occupation O
join libdb.user U on O.id = U.occupation_id
where U.id = _userid
into _limit;
RETURN _limit;
END;
$$ LANGUAGE plpgsql;
```

--Borrow a book

```
CREATE OR REPLACE  FUNCTION borrow_a_book(_userid integer, _book_copy_id integer, _days
integer) RETURNS void AS $$
DECLARE
_count integer;
_now date := now()::date;
```

```
_return_date date := _now + _days * '1 day'::interval;
_current_borrow_count  integer;
_borrow_limit integer :=get_borrow_limit(_userid);
BEGIN

--get current borrow count
select count(*)
from book_borrow BB
where BB.user_id = _userid and return_date_actual is null
into _current_borrow_count;

--check borrow limit
IF _current_borrow_count  >= _borrow_limit THEN
RAISE EXCEPTION 'cannot borrow more, limit %, current borrowed:
%',_borrow_limit,_current_borrow_count ;
END IF;

-- insert into book_borrow
insert into book_borrow(user_id, book_copy_id, borrow_date, return_date_expected)
values (_userid, _book_copy_id, _now, _return_date );

--change book_copy item status.
update book_copy BC
set status = 0
where id = _book_copy_id;
RETURN;
END;
$$ LANGUAGE plpgsql;
--select borrow_a_book(12, 509, 30);
```

--Claim a book loss.

```
--book_copy_id, userID, penalty_amount
CREATE OR REPLACE  FUNCTION claim_loss(bookCopyID integer, userID integer, penalty_amount
integer) RETURNS void AS $$
DECLARE
book_borrow_id_history  libdb.book_borrow.id%TYPE;
reason_str varchar :='Losing book';
_penalty_reason_id  penalty_reason.id%TYPE;
BEGIN

--find if there is such a book borrow history
select * from book_borrow BB
where BB.book_copy_id = bookCopyID and user_id = userID
order by BB.borrow_date DESC
limit 1
into book_borrow_id_history;
IF NOT FOUND THEN
RAISE EXCEPTION 'no such a book borrow history found bookCopyID:% userID:% ',
bookCopyID,userID;
END IF;
```

```
-- get penalty reason
select id from penalty_reason PR
where PR.desc = reason_str
into _penalty_reason_id;
IF NOT FOUND THEN
RAISE EXCEPTION 'no proper penalty reason found, % is searched for.', reason_str ;
END IF;

-- deduct 1 for book copy count
update book B
set copy_count = copy_count-1
where B.id in (select BC.book_id from book_copy BC where BC.id = bookCopyID);

-- set the book copy lost
update book_copy BC
set status = 2 -- book is lost permanently.
where BC.id =bookCopyID;

-- mark as the book returned now.
update book_borrow BB
set return_date_actual = now()::date
where BB.book_copy_id = bookCopyID and user_id = userID;

-- record for penalty
insert into penalty
values (userID, now()::date, null, penalty_amount,
book_borrow_id_history,_penalty_reason_id);
RAISE notice 'done';
RETURN;
END;
$$ LANGUAGE plpgsql;
```

--Pay the penalty bill.

```
--Can only pay for the most recent one.
--If there are severla pending penalties, call this function several times.
CREATE OR REPLACE  FUNCTION pay_penalty(_userid integer) RETURNS void AS $$
DECLARE
_target_row penalty%ROWTYPE;
BEGIN
--get the target row. no key defined, therefore get the entire row and use it as key..
select *
from penalty P
where P.user_id = _userid and P.clear_date is NULL
limit 1
into _target_row;

-- not found. error.
IF NOT FOUND THEN
RAISE EXCEPTION 'ERROR: no such a target row with userID:% ', _userid;
```

```
END IF;
-- clear the pending penalty.
update penalty P
set clear_date = now()::date
where P.user_id = _target_row.user_id
and P.issue_date = _target_row.issue_date
and P.amount = _target_row.amount -- real number comparison for equal is bad coding habit.
and P.book_borrow_id = _target_row.book_borrow_id
and P.penalty_reason_id = _target_row.penalty_reason_id;
RETURN;
END;
$$ LANGUAGE plpgsql;
```

--Ask library to import a book.

```
CREATE OR REPLACE  FUNCTION ask_for_import(_userid integer, _isbn varchar) RETURNS void AS
$$
DECLARE
_the_count integer;
BEGIN

-- check existence in book table.
select count(isbn)
from book B
where B.isbn= _isbn
into _the_count;

--book exists
IF _the_count <> 0 THEN
RAISE EXCEPTION 'ERROR: the book with ISBN:% already exists. ', _isbn;
END IF;

--check the existence of user isbn combination in wishilist
_the_count := 0;
select count(isbn)
from wishlist W
where W.isbn = _isbn and W.user_id = _userid
into _the_count;

--combination exists
IF _the_count <> 0 THEN
RAISE EXCEPTION 'ERROR: record with user_id:% isbn:% already exists. ', _userid, _isbn;
END IF;

--insert the book into wishlist.
insert into wishlist(isbn,request_date,status,user_id)
values (_isbn, now()::date, 0, _userid);
RETURN;
END;
$$ LANGUAGE plpgsql;
--select ask_for_import(20, 'VERY__POPULAR');
```

Application for library manager

--import a book

```
CREATE OR REPLACE  FUNCTION import_a_book(_isbn varchar, _title varchar, _publisher_id integer,
_pub_year integer,
_pages integer, _price real, _desc varchar, _sub_category integer, _copy_count integer ) RETURNS
void AS $$
DECLARE
_AD date := '0001-01-01'::date;
_year date := _AD + _pub_year * interval '1 year'; --smart.
_book_id integer;
BEGIN
insert into book (isbn ,title, publisher_id, year, pages, price, "desc", sub_category_id,
copy_count)
values (_isbn, _title, _publisher_id, _year, _pages::smallint, _price, _desc, _sub_category,
_copy_count::smallint);
```

--get book id

```
select B.id
from book B
where B.isbn=_isbn
into _book_id;
FOR i IN 1.._copy_count LOOP
insert into book_copy(book_id, status)
values (_book_id, 1); -- 1 means available in the library.
END LOOP;
RETURN;
END;
$$ LANGUAGE plpgsql;
--select import_a_book('test-import3', 'test-title-import3', 10, 1998, 203, 102, 'this book is more
than 100 dolars.', 60, 50 );
```

--Delete a user’s account from library.

```
CREATE OR REPLACE  FUNCTION delete_lib_user(_userid integer) RETURNS void AS $$
DECLARE
BEGIN
delete from libdb.user
where id = _userid ;
RETURN;
END;
$$ LANGUAGE plpgsql;
--select delete_lib_user(20);
```

--Delete a publication from library.

```
CREATE OR REPLACE  FUNCTION delete_publication(_isbn varchar) RETURNS void AS $$
DECLARE
BEGIN
delete from libdb.book
where isbn = _isbn;
RETURN;
END;
$$ LANGUAGE plpgsql;
--select delete_publication('D0YANCXAICLP4D');
```

Reports

--The top N most borrowed books list within a given periods.

```
CREATE OR REPLACE  FUNCTION top_n_most_borrowed(_from date, _to date,_count int)
RETURNS table (isbn varchar, title varchar, borrow_count bigint) AS $$
DECLARE
BEGIN
IF _from > _to THEN
RAISE EXCEPTION 'from date : %d > to date: %d !  ', _from,_to;
END IF;
RETURN QUERY
select B.isbn, B.title, count(B.isbn) as borrow_count
from book_borrow BB
join book_copy BC on BB.book_copy_id = BC.id
join book B on B.id = BC.book_id
where borrow_date between '2010-01-01'::date and now()::date
group by B.title, B.isbn
order by borrow_count desc
limit _count;
END;
$$ LANGUAGE plpgsql;
--select top_n_most_borrowed('2010-01-01'::date, now()::date,10);
```

--The top N most wanted books at the moment.

```
CREATE OR REPLACE  FUNCTION top_n_most_wanted(_n int)
RETURNS table (_isbn varchar, _wanted_count int) AS $$
DECLARE
BEGIN
RETURN QUERY
select W.isbn, count(W.isbn)::int as _wanted_count
from wishlist W
group by W.isbn
order by _wanted_count desc
limit _n;
END;
$$ LANGUAGE plpgsql;
--select top_n_most_wanted(10);
```

--A user’s borrow history

```
CREATE OR REPLACE  FUNCTION borrow_history(_userid int)
RETURNS table (user_id integer, borrow_date date, return_date_expected date, return_date_actual
date
,isbn varchar, title varchar, author varchar) AS $$
DECLARE
BEGIN
RETURN QUERY
```

```
select BB.user_id, BB.borrow_date,BB.return_date_expected, BB.return_date_actual,
B.isbn, B.title, (A.firstname || ' ' || A.lastname)::varchar as author
from book_borrow BB
join book_copy BC on BB.book_copy_id = BC.id
join book B on B.id = BC.book_id
join authorize AZ on AZ.book_id = B.id
join author A on A.id = AZ.author_id
where BB.user_id = _userid
order by BB.borrow_date desc;
END;
$$ LANGUAGE plpgsql;
--select borrow_history(12);
```

--A user’s penalty status.

```
CREATE OR REPLACE  FUNCTION penalty_status(_userid int)
RETURNS table (issue_date date, clear_date date, amount real
,des varchar, title varchar) AS $$
DECLARE
BEGIN
RETURN QUERY
select P.issue_date, P.clear_date, P.amount, PR.desc, B.title
from penalty P
join penalty_reason PR on PR.id = P.penalty_reason_id
join book_borrow BB on P.book_borrow_id = BB.id
join book_copy BC on BC.id = BB.book_copy_id
join book B on B.id = BC.book_id
where P.user_id = _userid
order by P.issue_date desc;
END;
$$ LANGUAGE plpgsql;

--select penalty_status(12);
```

Roles and Privileges

```
create role web_admin; /*admin with web interface*/
create role local_admin;/*admin with local machine*/
create role web_client; /*users from web interface*/

/*Schema level control. Roles can use schema.*/
GRANT USAGE ON SCHEMA libdb
to web_admin, local_admin, web_client;

/* table level for admin. Admin can do anything.*/
GRANT  ALL ON TABLE
publisher, sub_category, category, penalty_reason,
book, book_copy, penalty, authorize, bookshelf,
author,location, book_borrow, libdb.user,
building, occupation, wishlist
TO web_admin, local_admin;

/* table level for web_client. The web_client has some more restrictions*/
GRANT select ON TABLE
publisher, sub_category, category, penalty_reason,
book, book_copy, penalty, authorize, bookshelf,
author,location, book_borrow, libdb.user,
building, occupation, wishlist
TO web_client;

/*can only update two tables*/
GRANT update on TABLE
penalty, book_borrow
TO web_client;

/*can only insert into three tables*/
GRANT insert on TABLE
penalty, book_borrow, wishlist
TO web_client;

/* ENTIRE  TABLE LIST:
publisher, sub_category, category, penalty_reason,
book, book_copy, penalty, authorize, bookshelf,
author,location book_borrow, user,
building, occupation, wishlist
*/
```