数据库系统 课程实验3
数据库设计

计科210X甘晴void 202108010XXX

目录

实验目的

掌握数据库设计的基本方法及数据库设计工具。

实验内容

掌握数据库设计的基本步骤，包括数据库概念结构设计、逻辑结构设计，物理结构设计，数据库模式 SQL 语句生成。能够使用数据库设计工具进行数据库设计。

实验重难点

实验重点：概念结构设计、逻辑结构设计。

实验难点：逻辑结构设计。逻辑结构设计虽然可以按照一定的规则从概念结构转换而来，但是由于概念结构通常比较抽象，较少考虑更多细节，因此转换而成的逻辑结构还需要进一步调整与优化。逻辑结构承接概念结构和物理结构，处于核心地位，因此是数据库设计的重点和难点。

实验环境

DBMS：8.0.33 MySQL Community Server - GPL

可视化：Navicat Premium 16.1.6

命令行：Navicat自带命令列

Powerdesigner

实验过程

（0）数据库需求描述

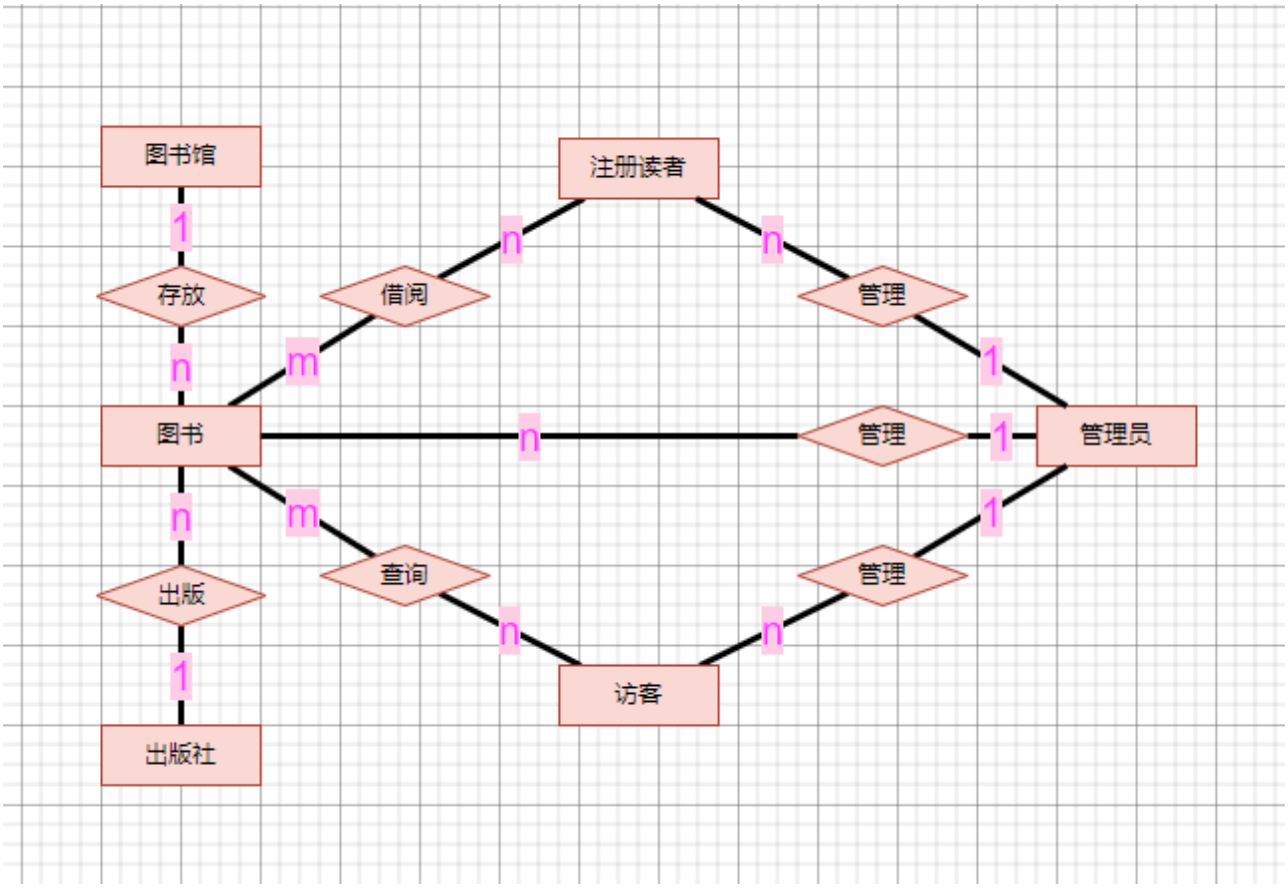
设计一个图书馆管理系统，有注册读者，访客，管理员三种身份，需要管理图书的信息以及它的出版与存放地点信息。一个注册读者能借阅多本图书，一本图书（在不同的时间）能被不同读者借阅。一个访客能查询多本图书，一本图书能被多个访客查询。一个管理员管理图书，访客，注册读者（该图书由该管理员经手，该访客或注册读者由该管理员引导）。一本图书只能放在一个图书馆里，一本图书只能由一个出版社所出版。

对于借阅，记录借阅发生的时间，应该归还的时间，是否可以续借，借阅的终端地址。对于查询，记录查询发生的时间，以及查询发生的终端地址。

（1）数据库概念结构设计

E-R图

从需求描述中抽象出用于描述关系的E-R图如下。



实体

从需求描述中识别出以下6个实体，并分别为它们赋予属性。

图书馆**library**：

- 图书馆编号：library_id
- 图书馆名：library_name
- 图书馆联系电话：library_tel
- 图书馆地址：library_address

图书**book**：

- 图书编号：book_id
- 图书名称：book_name
- 图书作者：book_writer
- 图书编号：book_pressid
- 图书出版日期：book_pressdate
- 图书ISBN：book_isbn
- 图书价格：book_price
- 图书副本数量：book_copynum

出版社**press**：

- 出版社编号：press_id
- 出版社名：press_name
- 出版社联系电话：press_tel
- 出版社地址：press_address

注册读者**reader**：

- 读者编号：reader_id
- 读者姓名：reader_name
- 读者性别：reader_sex
- 读者电话号码：reader_tel
- 读者邮箱地址：reader_email
- 读者证件类型：reader_doc
- 读者证件号码：reader_docid
- 读者注册日期：reader_registerdate
- 读者备注：reader_others

访客**visitor**：

- 访客编号：visitor_id
- 访客注册日期：visitor_registerdate

管理员**admin**：

- 管理员编号：admin_id
- 管理员姓名：admin_name
- 管理员性别：admin_sex
- 管理员电话号码：admin_tel
- 管理员邮箱地址：admin_email
- 管理员入职时间：admin_registerdate
- 管理员职位：admin_pos

关系

关系在逻辑结构设计时补入完善的ER图中，这里先列出来。

借阅**borrow**

- 借阅日期：borrow_date
- 应归还日期：borrow_expect_returndate
- 是否可续借：borrow_ifrenew
- 是否归还：borrow_ifreturn
- 最终归还日期：borrow_returndate
- 借阅时间：borrow_time
- 借阅终端地址：borrow_terminal(3位)

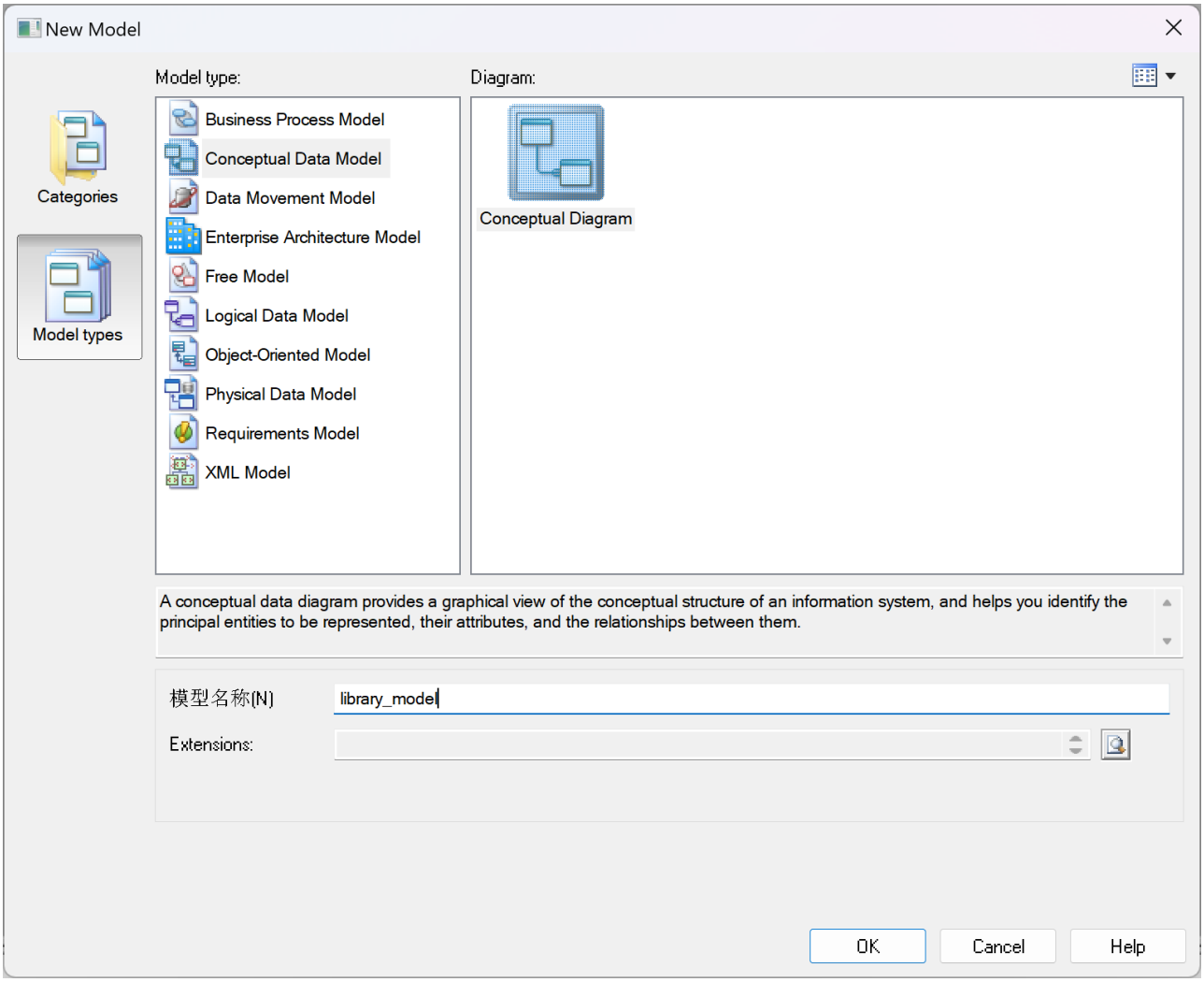
查询**query**

- 查询时间：query_time
- 查询终端地址：query_terminal

【初步使用**PowerDesigner**】

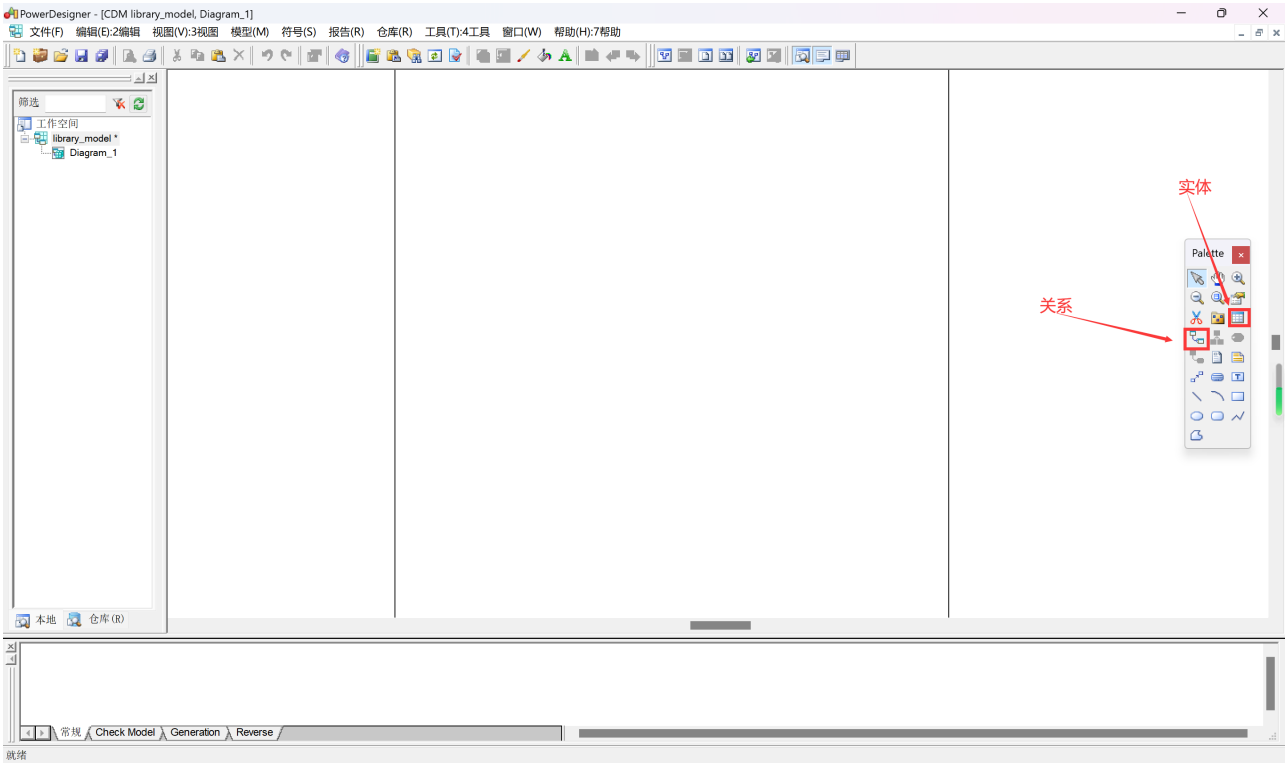
使用PowerDesigner可以很方便的绘制ER图，下面简单叙述该软件的初步操作。

①新建一个“概念模型”文件



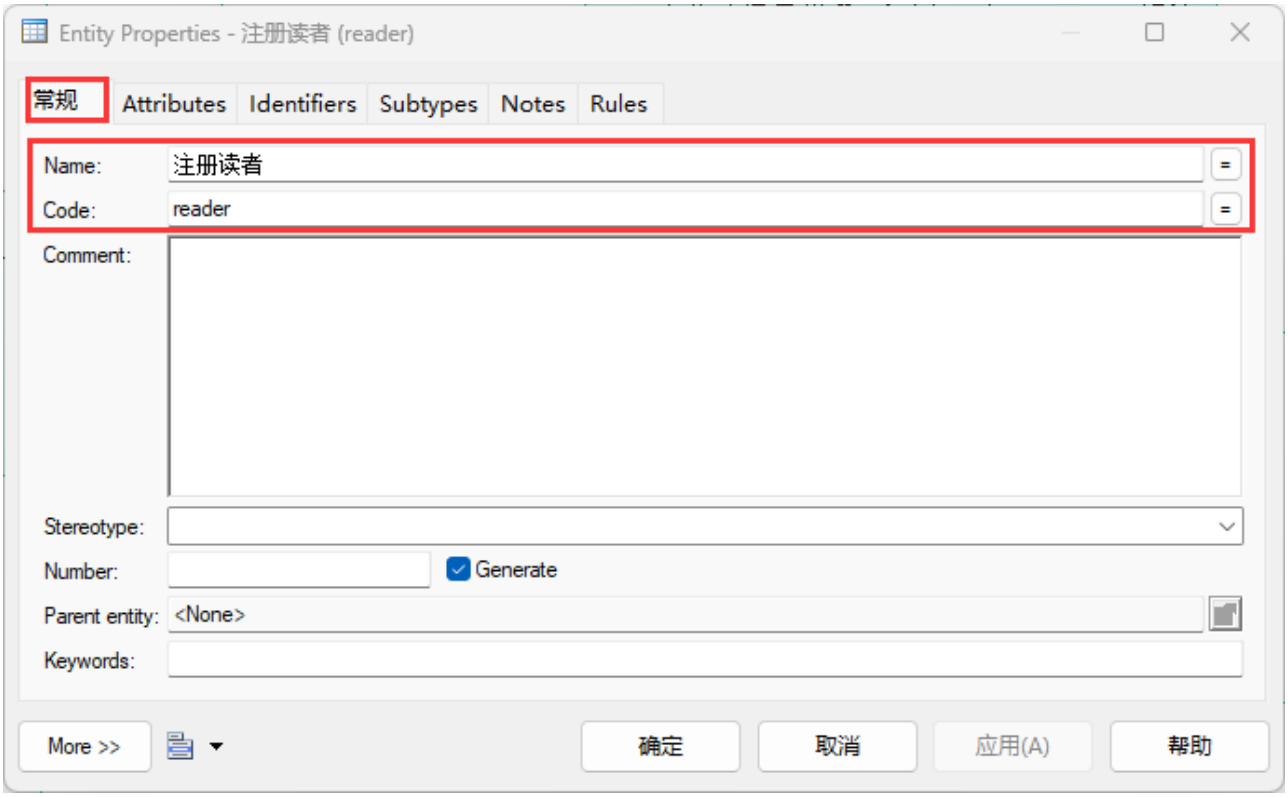
②使用工具框绘制实体并连接关系

【注意】对于相似的实体，千万不要偷懒用ctrl+cv的方法，这个复制过去的不是一个新的副本，它更像是一个链接，用人话说就是，你修改了复制本或者原本，这两个都会跟着一起变化，很烦（多花了好多时间）。一定要自己新建出一个实体去写。

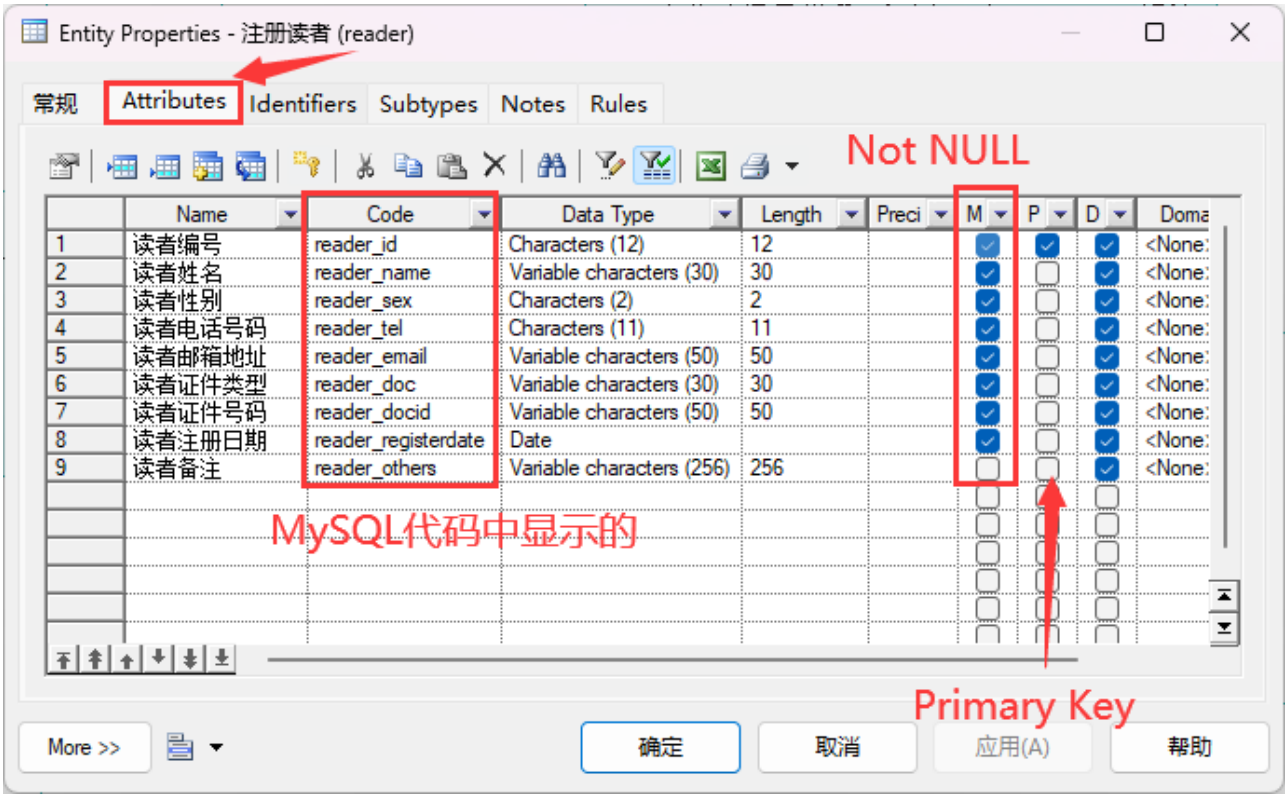


③编辑实体属性

双击实体，可以编辑实体属性，第一个Name是显示的中文，第二个Code是代码中呈现的。



点击Attributes设置实体的属性以及一些特征

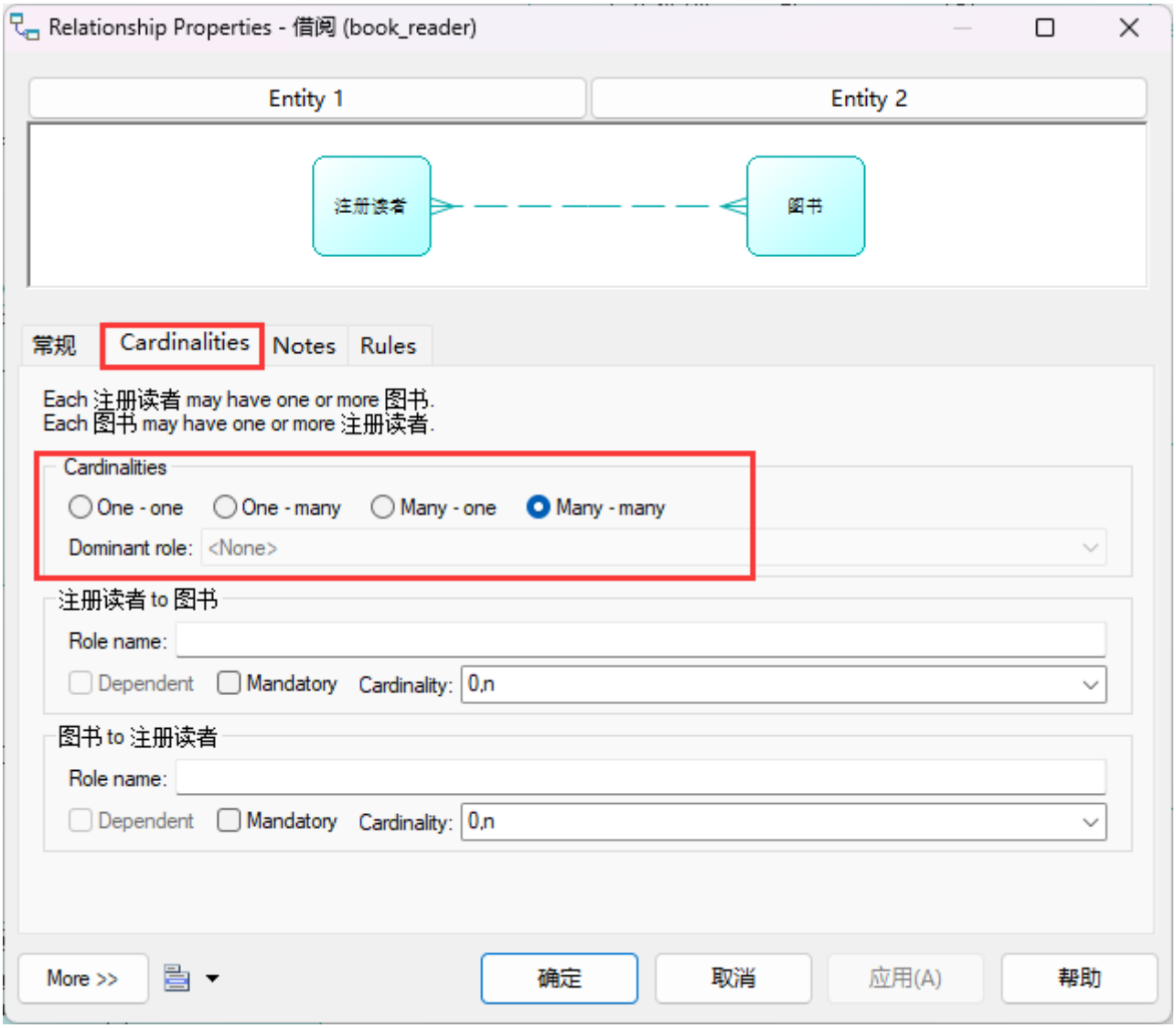


词条具体含义如下：

- Name：实体名字一般为中文（这是显示在图上给人看的）
- Code：实体代号,一般用英文（这是写在代码里的，一定要用英文，不然无法生成代码）
- Data Type：实体的数据类型
- Length：实体的数据长度
- Domain：域，表示属性取值范围如可以创建10个字符的地址域
- M：Mandatory强制属性，表示该属性必填。不能为空（相当于NOT NULL）
- P：Primary Identifer是否是主标识符，表示实体唯一标识符（相当于 PRIMARY KEY UNIQUE）
- D：Displayed显示出来，默认全部勾选（这个要选上，否则就隐藏了）

④编辑关系的特征

同样地，双击关系，也可编辑相应特征。

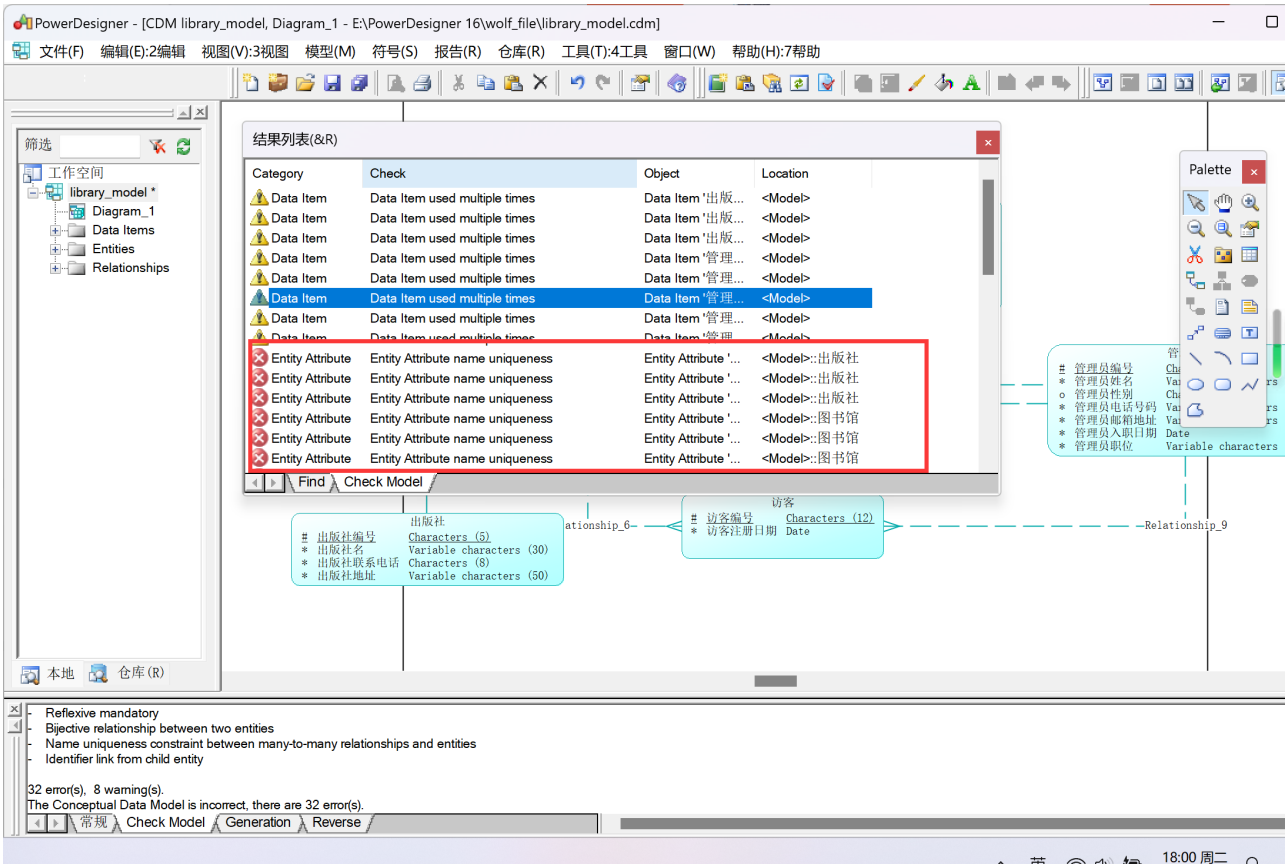


这里可以调节映射关系，是1:1还是1:n还是n:m。

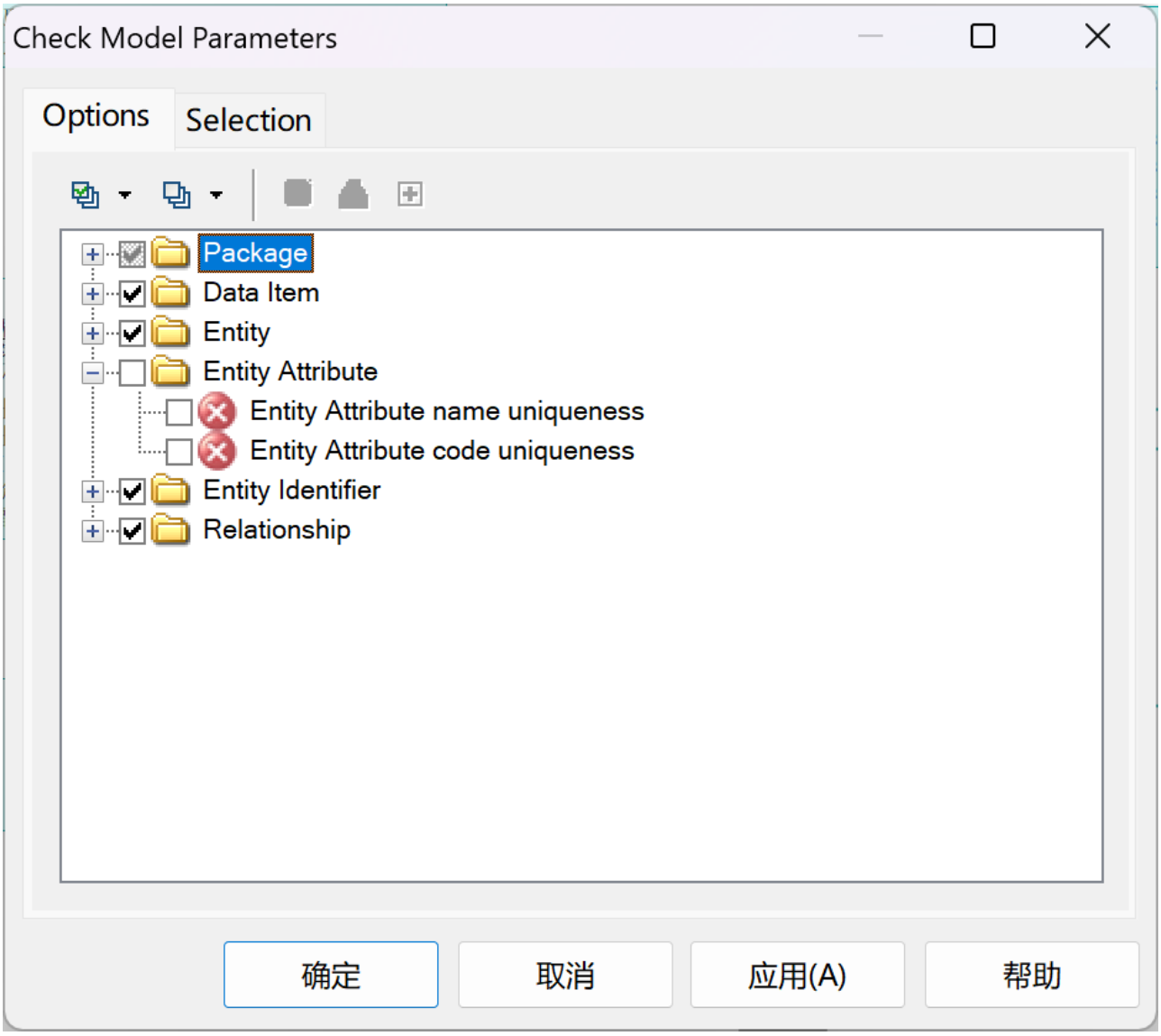
⑤检查模型

实际上可有可无，如果能正常生成，一般是没有问题的。

如果出现这样的错误

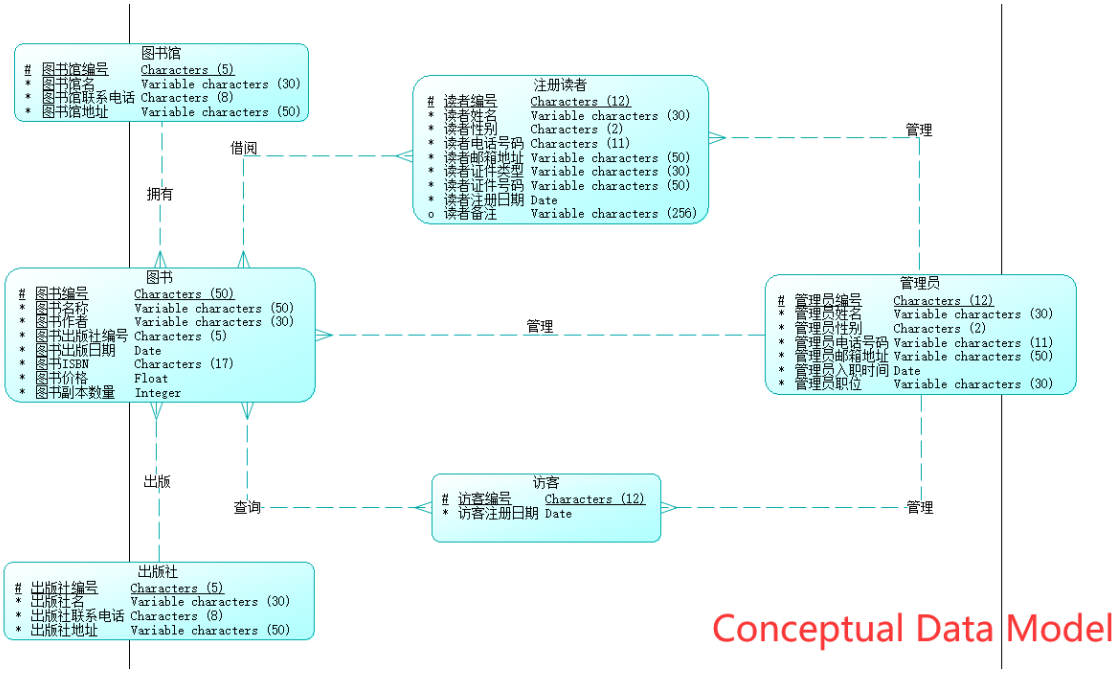


解决办法是把这两个勾去掉，不检查这两项即可。



【操作】

使用PowerDesigner绘制数据库概念模型图如下



（2）数据库逻辑结构设计

根据上述E-R图以及实体属性使用PowerDesigner设计数据库逻辑结构。

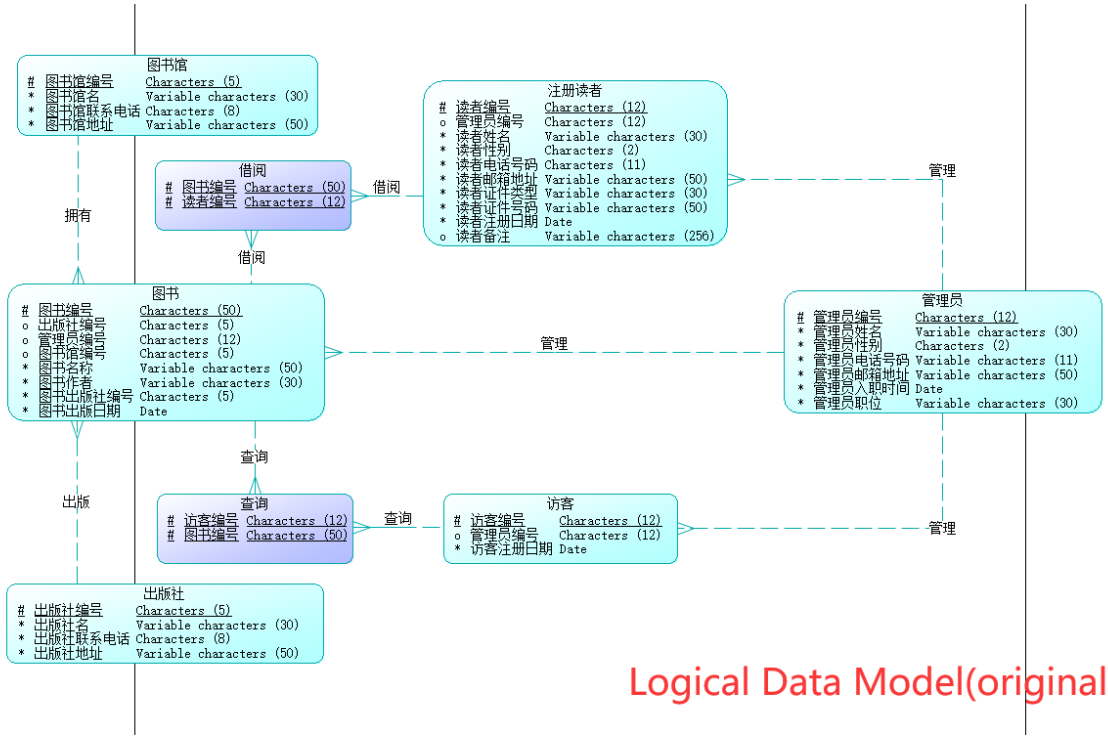
E-R图向关系模型的转换规则

- 一个1:1联系可以转换为一个独立的关系模式，也可以与任意一端对应的关系模式合并。
- 一个1:n联系可以转换为一个独立的关系模式，也可以与n端对应的关系模式合并。
- 一个m:n联系只能转换为一个关系模式三个或三个以上实体间的一个多元联系转换为一个关系模式。
- 具有相同码的关系模式可合并

（事实上，powerdesigner就是按照这个来的，最终它会合并所有1:1和1:n的关系，并为所有m:n的关系生成一个新的关系表）

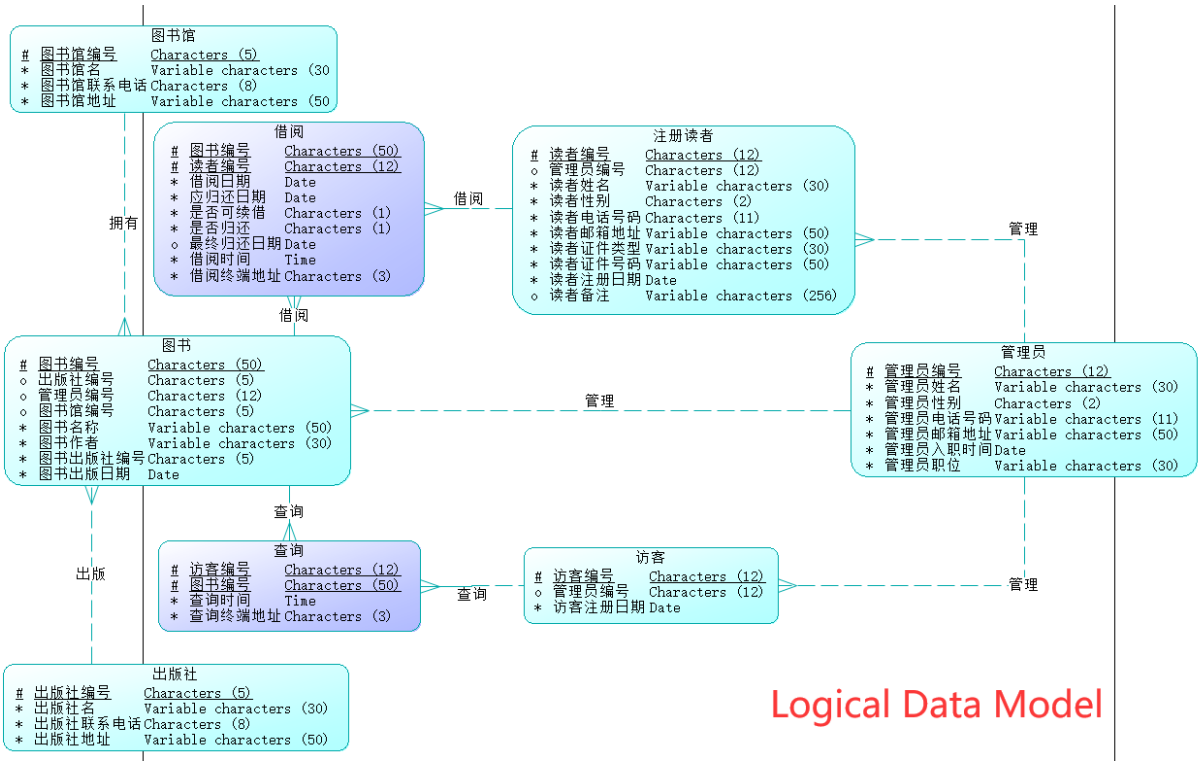
【操作】

使用PowerDesigner创建数据库逻辑模型：Tool>Generate Logical Data Model即可生成数据库逻辑模型，直接点击生成即可，不需要做其它修改。生成如下：



这是它为我们生成的，我们可以看到多出来两个深色的部分，这就是为我们生成的两个新的关系表（这基本上只有在n:m的时候才会生成）。但实际上这还是不符合我们的预期的，比如对于借阅，我肯定要知道借阅发生的时间，应该归还的时间，以及是否可以续借等等这些额外的附加信息，对于查询，我可能想知道查询发生的时间，以及查询发生的终端地址等等，这些就需要我们手动对深色部分的模型作出更改。

对这两个关系表进行更改后，得到模型如下：



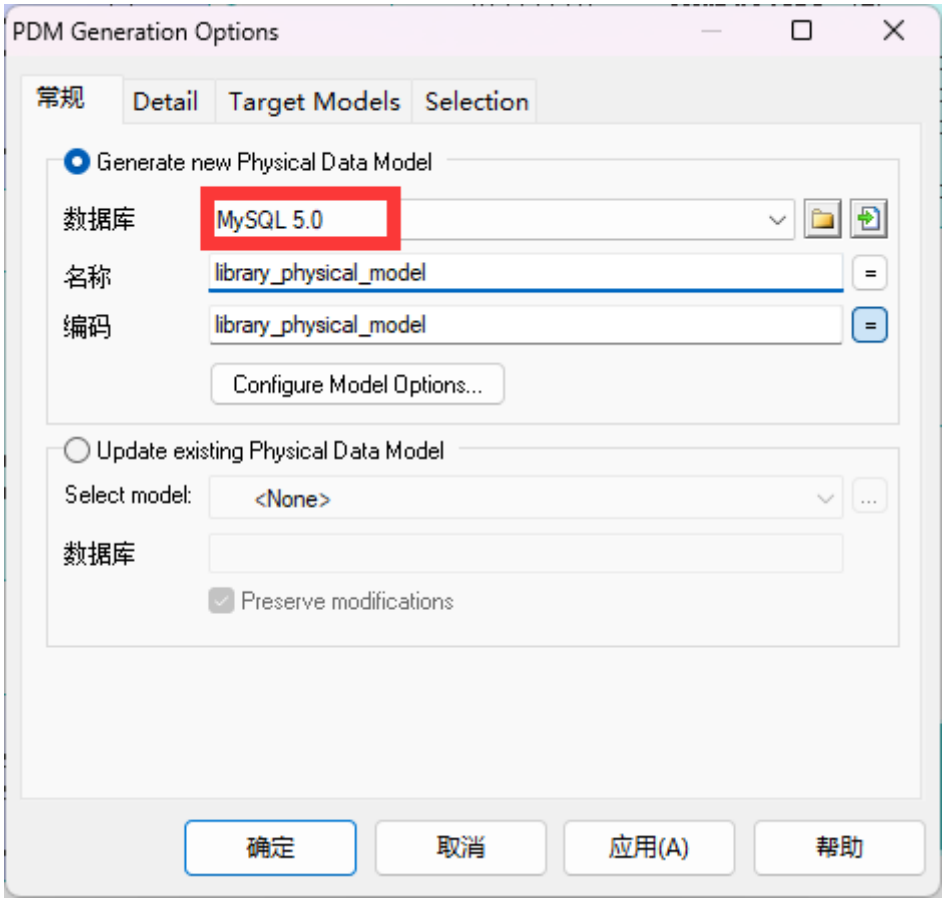
（3）数据库物理结构设计

数据库物理结构首先根据逻辑结构自动转换生成，然后根据应用需求设计数据库的索引结构、存储结构。

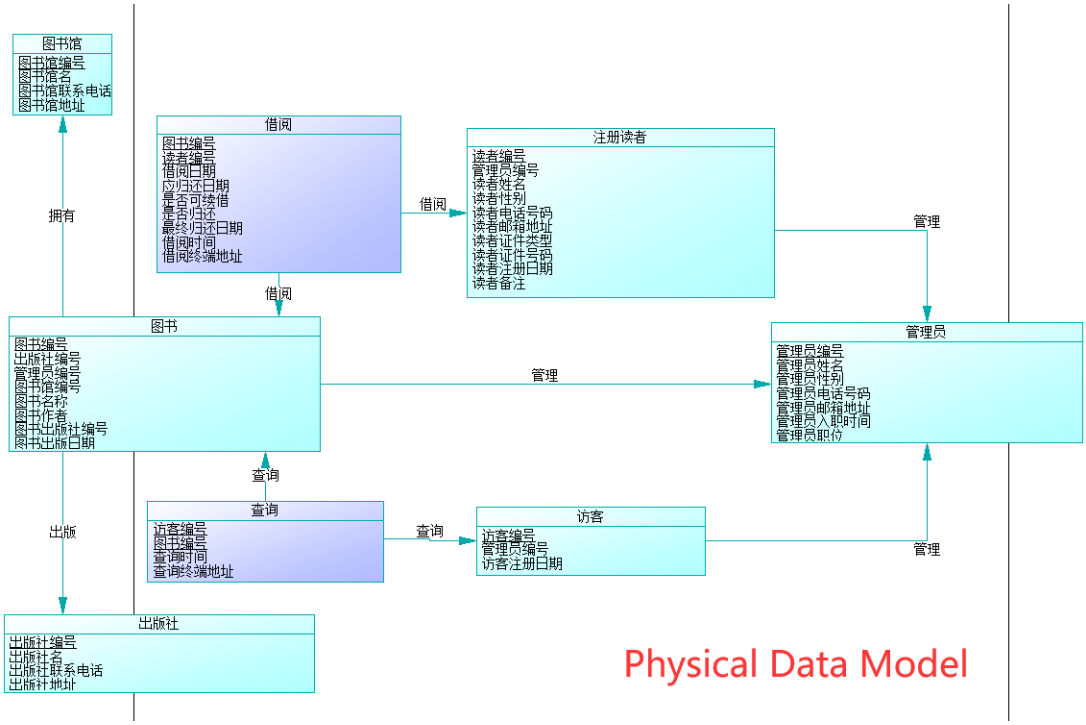
选择索引存取方法，数据库会自动为每个关系的主码建立索引。对于该系统，不需要建立其他索引。

【操作】

使用PowerDesigner创建数据库逻辑模型：Tool>Generate Physical Data Model即可生成数据库逻辑模型，这里要注意修改这个数据库为我们自己用的数据库（话说这个MySQL5.0是不是有点老了）。



结果如下：

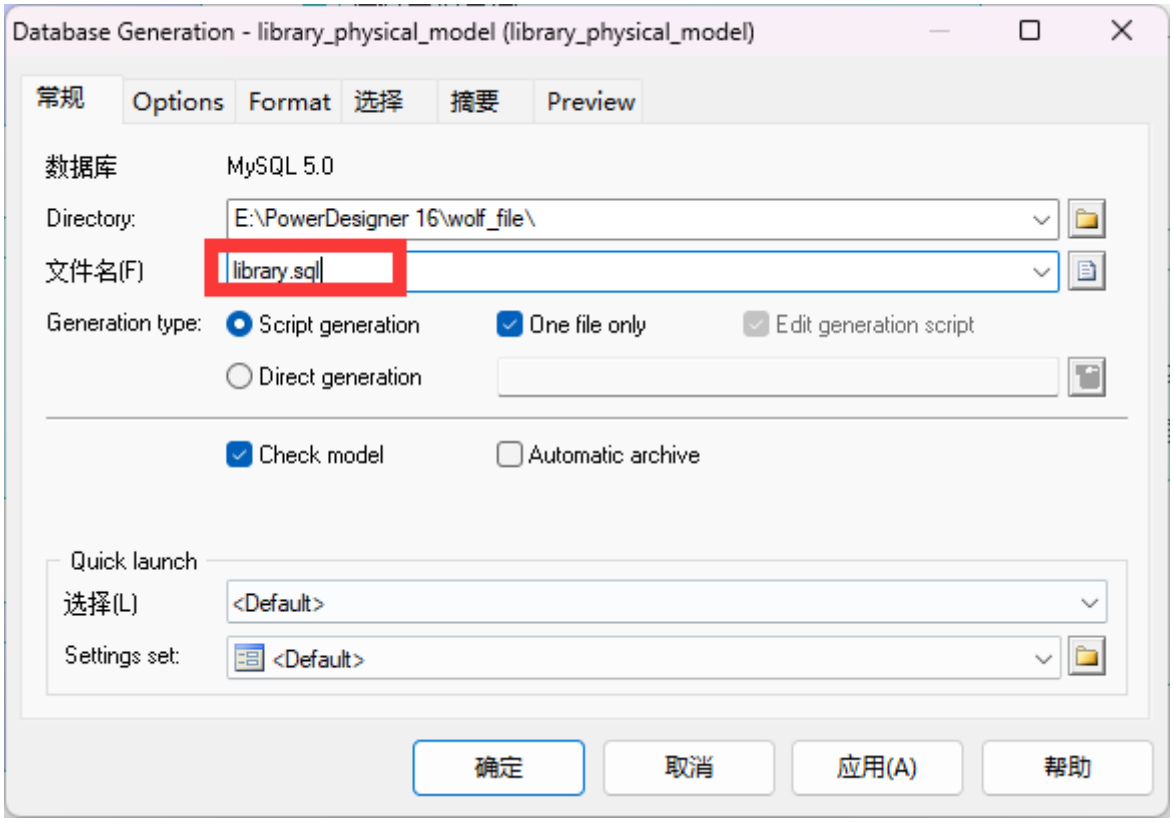


（4）SQL语句生成

这一步可以自动生成SQL代码。

【操作】

使用PowerDesigner创建数据库逻辑模型：Database>GenerateDatabase，即可生成。这里可以设置文件的保存路径和文件名。



代码

最后生成的代码如下。

```
/*==============================================================*/
/* DBMS name:      MySQL 5.0                                    */
/* Created on:     2023/12/12 21:41:42                          */
/*==============================================================*/


drop table if exists admin;

drop table if exists book;

drop table if exists book_reader;

drop table if exists book_visitor;

drop table if exists library;

drop table if exists press;

drop table if exists reader;
```

```
drop table if exists visitor;

/*==============================================================*/
/* Table: admin                                                 */
/*==============================================================*/
create table admin
(
   admin_id             char(12) not null,
   admin_name           varchar(30) not null,
   admin_sex            char(2) not null,
   admin_tel            varchar(11) not null,
   admin_email          varchar(50) not null,
   admin_registerdate   date not null,
   admin_pos            varchar(30) not null,
   primary key (admin_id)
);

/*==============================================================*/
/* Table: book                                                  */
/*==============================================================*/
create table book
(
   book_id              char(50) not null,
   press_id             char(5),
   admin_id             char(12),
   library_id           char(5),
   book_name            varchar(50) not null,
   book_writer          varchar(30) not null,
   book_pressid         char(5) not null,
   book_pressdate       date not null,
   book_isbn            char(17) not null,
   book_price           float not null,
   book_copynum         int not null,
   primary key (book_id)
);

/*==============================================================*/
/* Table: book_reader                                           */
/*==============================================================*/
create table book_reader
(
```

```
   book_id              char(50) not null,
   reader_id            char(12) not null,
   borrow_date          date not null,
   borrow_expect_returndate date not null,
   borrow_ifrenew       char(1) not null,
   borrow_ifreturn      char(1) not null,
   borrow_returndate    date,
   borrow_time          time not null,
   borrow_terminal      char(3) not null,
   primary key (book_id, reader_id)
);

/*==============================================================*/
/* Table: book_visitor                                          */
/*==============================================================*/
create table book_visitor
(
   vistor_id            char(12) not null,
   book_id              char(50) not null,
   query_time           time not null,
   query_terminal       char(3) not null,
   primary key (vistor_id, book_id)
);

/*==============================================================*/
/* Table: library                                               */
/*==============================================================*/
create table library
(
   library_id           char(5) not null,
   library_name         varchar(30) not null,
   library_tel          char(8) not null,
   library_address      varchar(50) not null,
   primary key (library_id)
);

/*==============================================================*/
/* Table: press                                                 */
/*==============================================================*/
create table press
(
   press_id             char(5) not null,
```

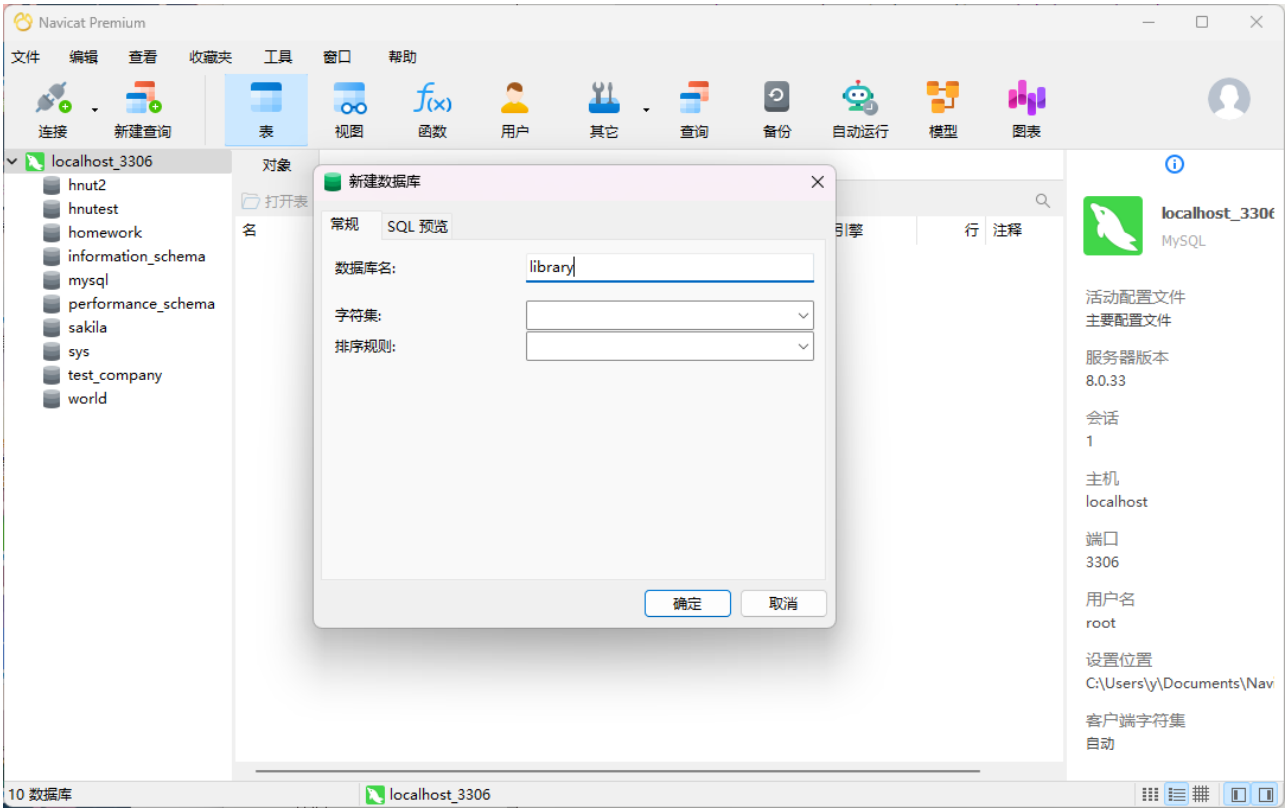
```
   press_name           varchar(30) not null,
   press_tel            char(8) not null,
   press_address        varchar(50) not null,
   primary key (press_id)
);

/*==============================================================*/
/* Table: reader                                                */
/*==============================================================*/
create table reader
(
   reader_id            char(12) not null,
   admin_id             char(12),
   reader_name          varchar(30) not null,
   reader_sex           char(2) not null,
   reader_tel           char(11) not null,
   reader_email         varchar(50) not null,
   reader_doc           varchar(30) not null,
   reader_docid         varchar(50) not null,
   reader_registerdate  date not null,
   reader_others        varchar(256),
   primary key (reader_id)
);

/*==============================================================*/
/* Table: visitor                                               */
/*==============================================================*/
create table visitor
(
   vistor_id            char(12) not null,
   admin_id             char(12),
   vistor_registerdate  date not null,
   primary key (vistor_id)
);

alter table book add constraint FK_admin_book foreign key
(admin_id)
      references admin (admin_id) on delete restrict on update
restrict;

alter table book add constraint FK_book_library foreign key
(library_id)
```

```
      references library (library_id) on delete restrict on update
restrict;

alter table book add constraint FK_book_press foreign key
(press_id)
      references press (press_id) on delete restrict on update
restrict;

alter table book_reader add constraint FK_book_reader foreign key
(book_id)
      references book (book_id) on delete restrict on update
restrict;

alter table book_reader add constraint FK_book_reader2 foreign key
(reader_id)
      references reader (reader_id) on delete restrict on update
restrict;

alter table book_visitor add constraint FK_book_visitor foreign key
(vistor_id)
      references visitor (vistor_id) on delete restrict on update
restrict;

alter table book_visitor add constraint FK_book_visitor2 foreign
key (book_id)
      references book (book_id) on delete restrict on update
restrict;

alter table reader add constraint FK_admin_reader foreign key
(admin_id)
      references admin (admin_id) on delete restrict on update
restrict;

alter table visitor add constraint FK_admin_visitor foreign key
(admin_id)
      references admin (admin_id) on delete restrict on update
restrict;
```

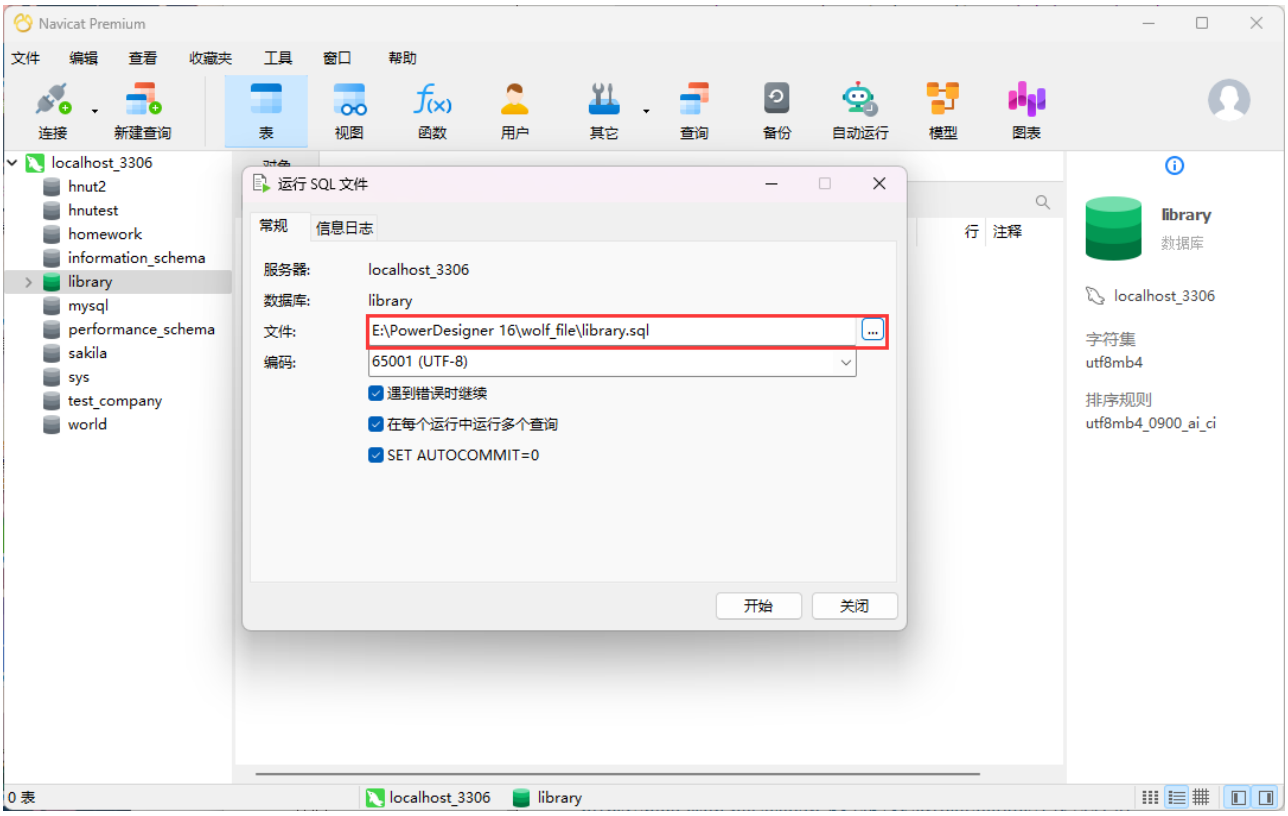
验证

使用Navicat访问该sql。

在Navicat左边栏连接本地数据库，新建一个library数据库

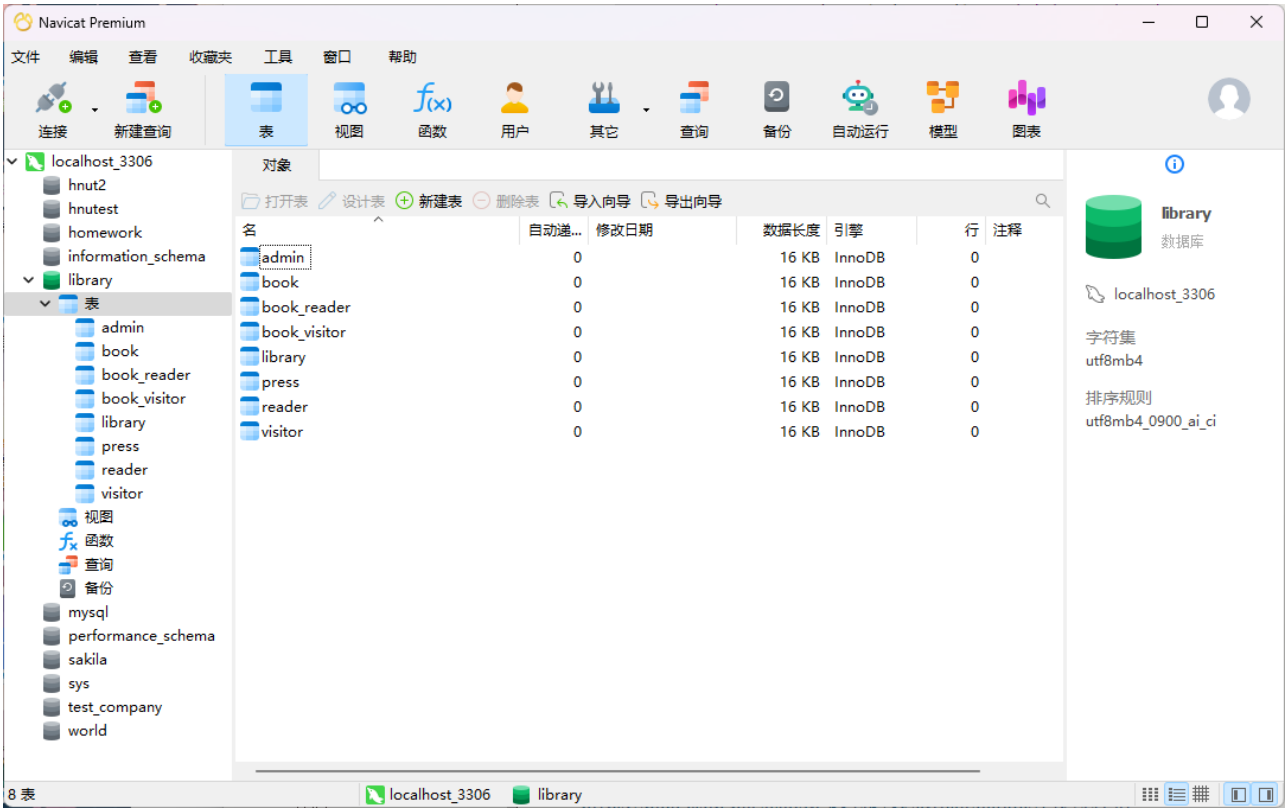


在左边栏右击library>运行SQL文件...



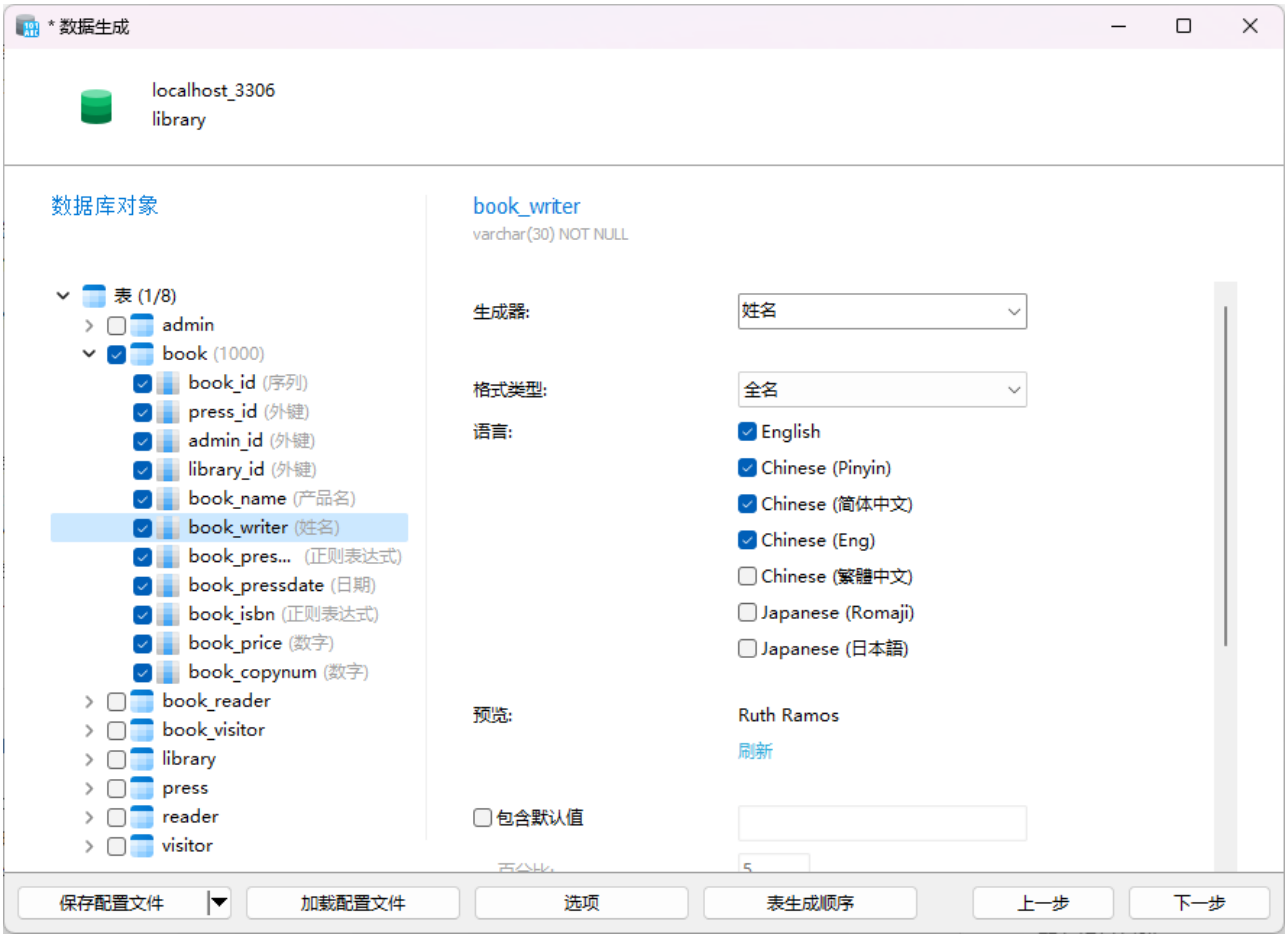
设置在遇到“错误”继续运行，可能有错误，这是正常现象，因为我们设置了很多非空，而实际上我们还没有导入数据。

成功运行之后可以见到界面如下，我们成功建立了一个数据库。

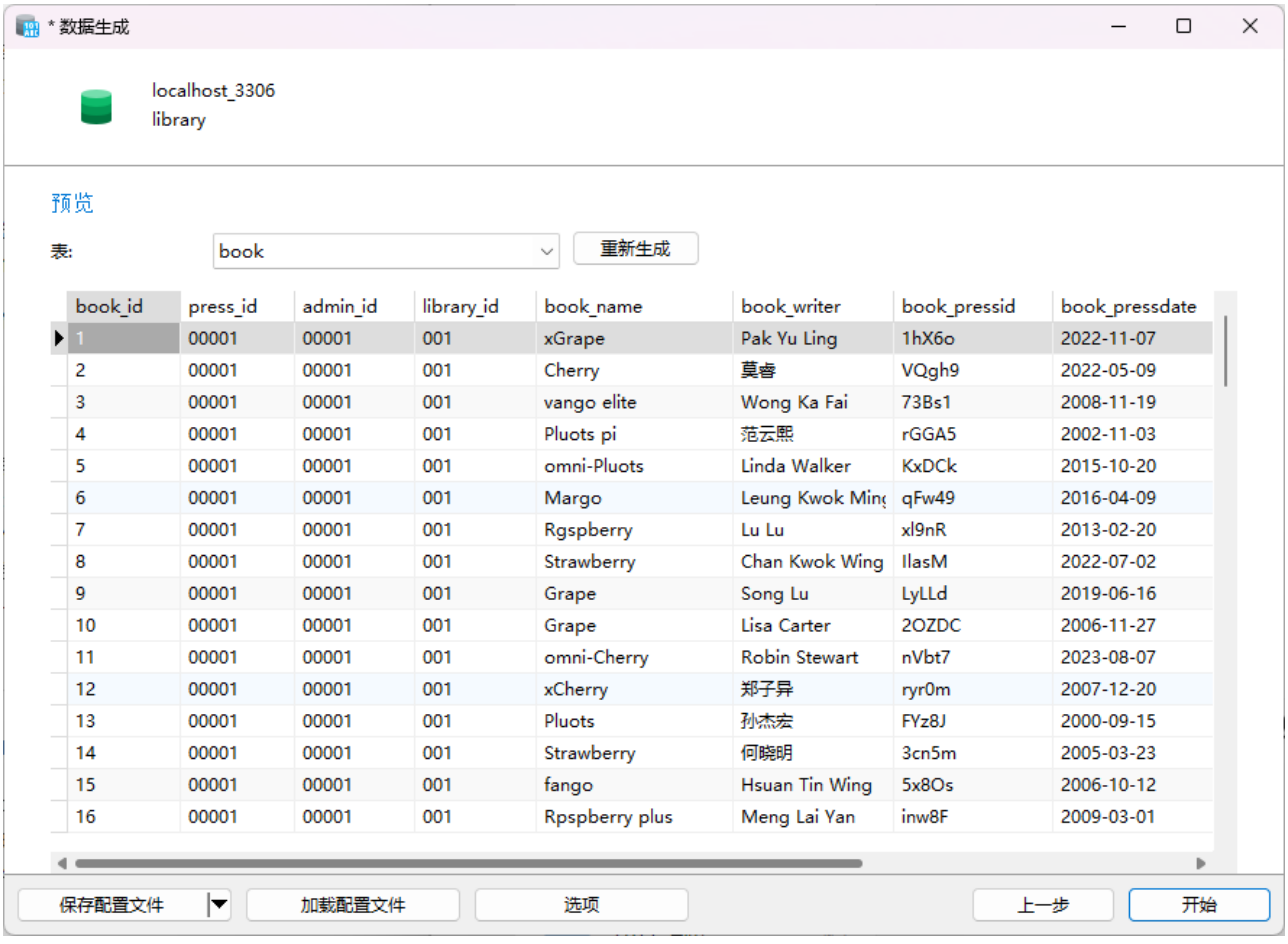


使用Navicat自带的数据生成功能生成数据，下面是生成书的作者名字的一个示例。这些每个属性值都是可以我们自己指定规则或者给予引导的，也可以用正则表达式或者枚举来限制生成的结果。

【注意】由于书本我们定义了外键，所以要先人为写好至少一条admin,press,library的信息，否则book没法生成，因为外键是NOT NULL且没法被填充。

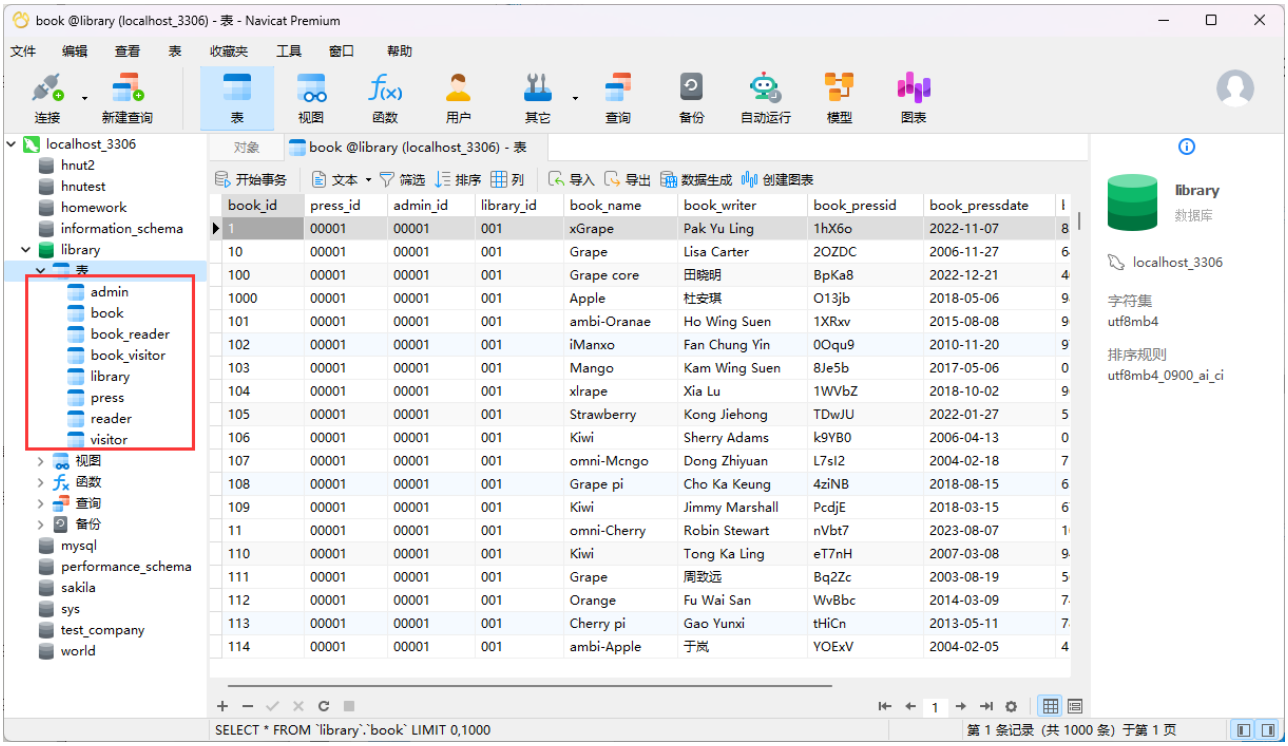


根据对应的规则生成出来关于书的数据（部分）如下（这里就没有去考虑一些细节，只是做做一个范例）。



book_id	press_id	admin_id	library_id	book_name	book_writer	book_pressid	book_pressdate
1	00001	00001	001	xGrape	Pak Yu Ling	1hX6o	2022-11-07
2	00001	00001	001	Cherry	莫睿	VQgh9	2022-05-09
3	00001	00001	001	vango elite	Wong Ka Fai	73Bs1	2008-11-19
4	00001	00001	001	Pluots pi	范云熙	rGGA5	2002-11-03
5	00001	00001	001	omni-Pluots	Linda Walker	KxDCk	2015-10-20
6	00001	00001	001	Margo	Leung Kwok Min	qFw49	2016-04-09
7	00001	00001	001	Rgspberry	Lu Lu	xl9nR	2013-02-20
8	00001	00001	001	Strawberry	Chan Kwok Wing	IlasM	2022-07-02
9	00001	00001	001	Grape	Song Lu	LyLLd	2019-06-16
10	00001	00001	001	Grape	Lisa Carter	2OZDC	2006-11-27
11	00001	00001	001	omni-Cherry	Robin Stewart	nVbt7	2023-08-07
12	00001	00001	001	xCherry	郑子异	ryr0m	2007-12-20
13	00001	00001	001	Pluots	孙杰宏	FYz8J	2000-09-15
14	00001	00001	001	Strawberry	何晓明	3cn5m	2005-03-23
15	00001	00001	001	fango	Hsuan Tin Wing	5x8Os	2006-10-12
16	00001	00001	001	Rpspberry plus	Meng Lai Yan	inw8F	2009-03-01

对于左边的这些表格都可以做数据生成。



然后就可以在这个基础上做数据查询更新等等操作了。

如果导入的是真实的数据，就可以做一个真正的图书馆数据库系统。

当然还要加上java或php写的前端，这就不是本次实验涉及的内容了，是大作业需要考虑的了。

参考文献

https://blog.csdn.net/weixin_63246738/article/details/128744120

https://www.imangodoc.com/pFUil7ts.html

https://blog.csdn.net/Aaron503/article/details/128280233

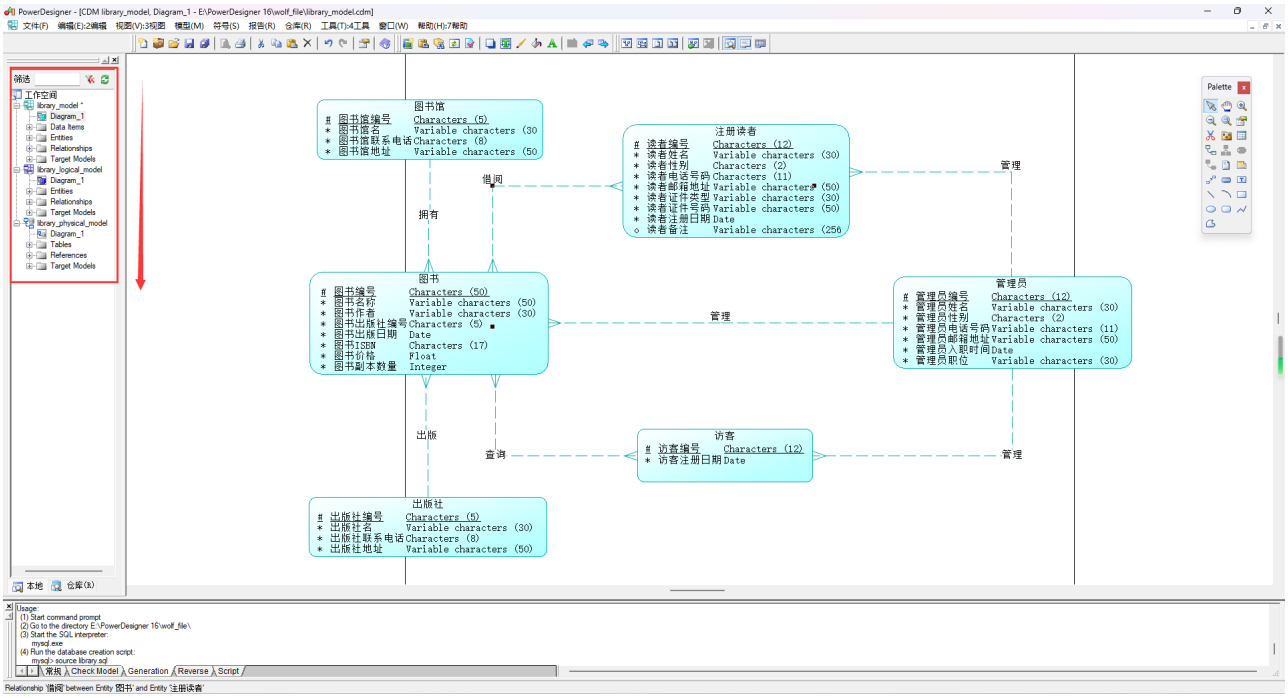
https://blog.csdn.net/qq_51684393/article/details/128413966

实验感悟

实验没有太大的难度，主要的时间花在思考我要做什么，以及探索PowerDesigner的使用上。有几点感悟如下。

需求分析太重要。不得不说一个明确的需求分析真的是太重要太重要了，如果知道想要做什么，有明确的需求的话，完成几个部分的设计真的不是很难的事情。但如果需求分析没有做好，后面真的就没办法开展。

PowerDesigner很好用。这个工具真的是太强大了，首先是很用户友好，对于一个纯新手，看看教程短时间可以上手。此外是逻辑很清晰，看看这张图，左边的栏从上往下依次就是“概念模型”，“逻辑模型”，“物理模型”，最后再生成SQL代码。工具清晰加上思路清晰，就能很快完成任务。



实验过程有意义。算是完整地走了一遍很简单的数据库雏形设计，挺有价值，为以后大作业奠定了一些基础。