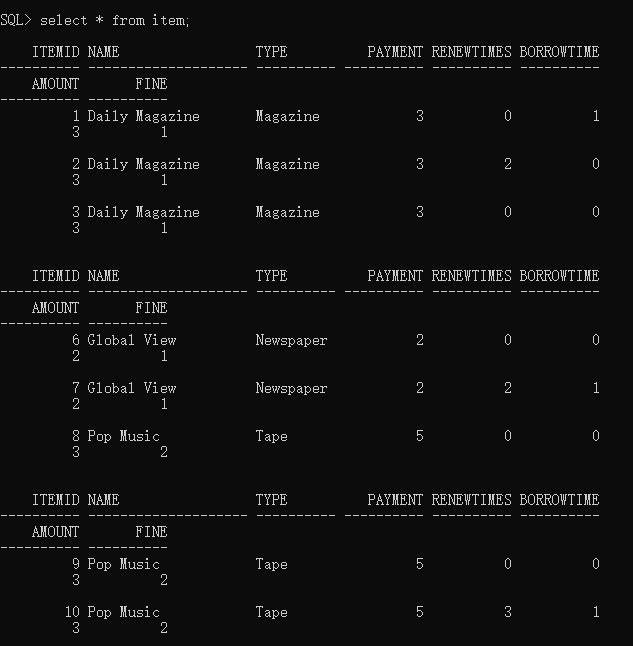
Report

I use java to implement the project. The sql script named finalprojectsql.sql under /home/classes/ece467/zxx187. My java script is under /home/classes/ece467/zxx187/finalproject/templates/finalproject. I didn’t change the name from templates given.

This project describes a library database and can implement simply borrowing action. In my design I have 4 tables in the database, which are patrons, borrow, item and request.

CREATE TABLE item(

itemID integer PRIMARY KEY NOT NULL,

name VARCHAR(20),

type VARCHAR(10),

payment integer NOT NULL,

Renewtimes integer NOT NULL,

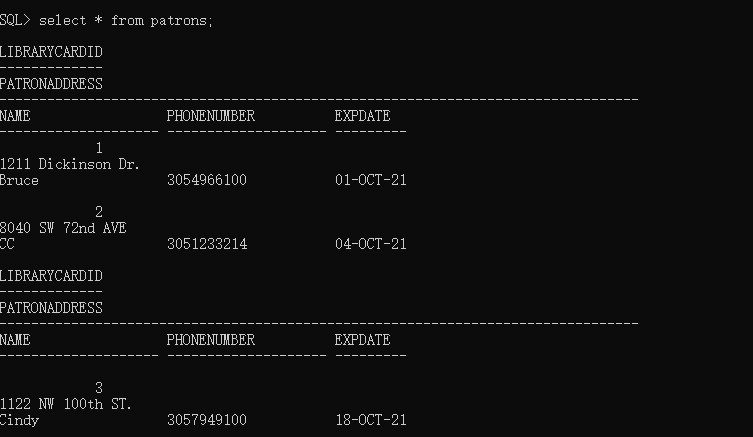
borrowtime integer not null,

Amount integer NOT NULL,

Fine integer NOT NULL

);

This is the first table and corresponding sql script. The keys’ functions are just as their names. The item table records basic information of a library item. Their names, type, payment per day, fine per day, amount ,renew times and borrow times. In my table I put into 8 values of different items.



CREATE TABLE Patrons(

LibrarycardID integer PRIMARY KEY NOT NULL,

PatronAdderss VARCHAR(100),

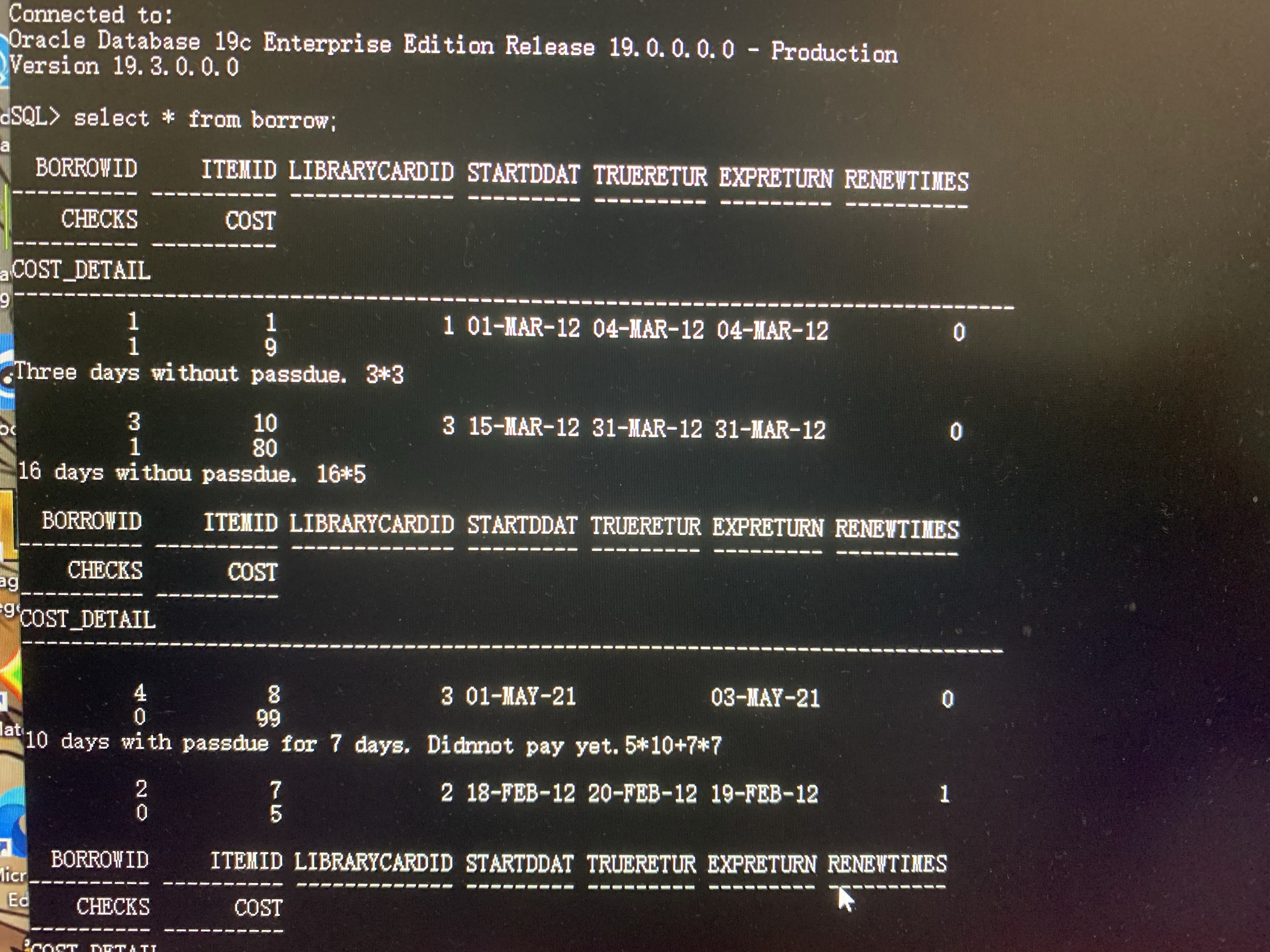
Name VARCHAR(20),

phoneNumber varchar(20),

ExpDate DATE

);

Here comes the second table which is patrons table. This table records information of a patron. In my table, I put into 3 different patrons.



CREATE TABLE Borrow (

borrowid integer PRIMARY KEY,

itemid integer NOT NULL,

LibrarycardID integer NOT NULL,

StartDate DATE,

TrueReturnDate DATE,

ExpReturnDate DATE,

renewtimes integer default 0,

checks integer default 0,

cost integer NOT NULL,

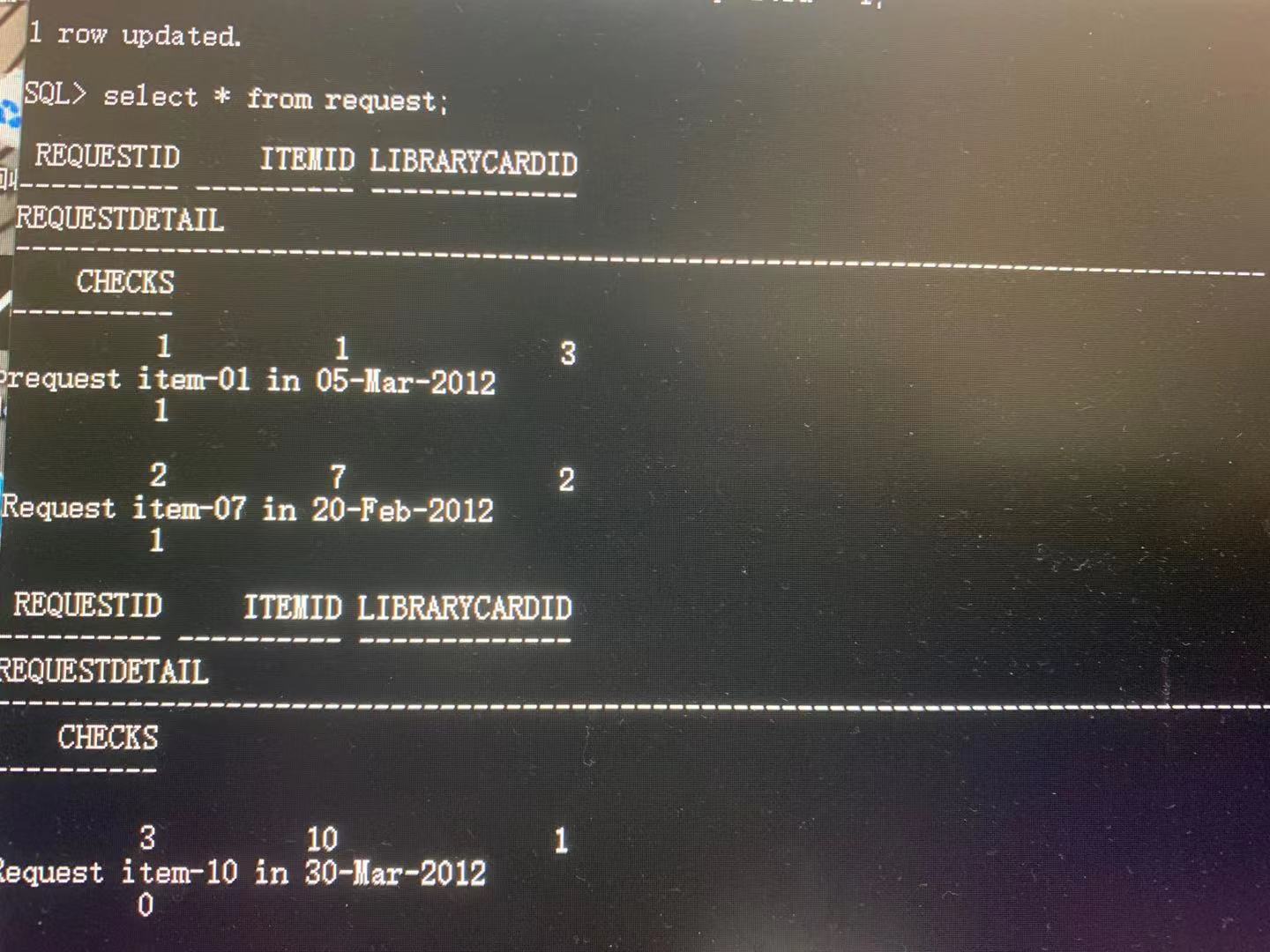
cost\_detail varchar(255),

FOREIGN KEY(itemID) REFERENCES item(itemID),

FOREIGN KEY(LibrarycardID) REFERENCES Patrons(LibrarycardID)

);

This is my borrow table. It records a borrowing action which includes start date, actual return date(true return date) and expected return date ( expreturndate), item id and so on. The checks works for recording whether a borrowing action’s payment has been done(works for the query 3 and 4). If a payment has done the check should be 1. Otherwise it should be 0. Cost\_detail records detail in borrowing action. This table also has two foreign keys and I have 4 borrow values.



CREATE TABLE Request(

RequestID integer PRIMARY KEY,

itemid integer NOT NULL,

LibrarycardID integer NOT NULL,

RequestDetail varchar(255),

checks integer default 0,

FOREIGN KEY(itemID) REFERENCES item(itemID),

FOREIGN KEY(LibrarycardID) REFERENCES Patrons(LibrarycardID)

);

Request is for patrons to pre-borrowing (booking) an item. But if it conflicts with exiting borrowing action recorded in borrow table, the checks will be 0. For example in my requestid= 3 row. The patron 1 wants to get item 10 in 30-Mar-2012. In the meantime, item 10 was not return and the true return date was 31-Mar-2012. So the checks becomes 0.

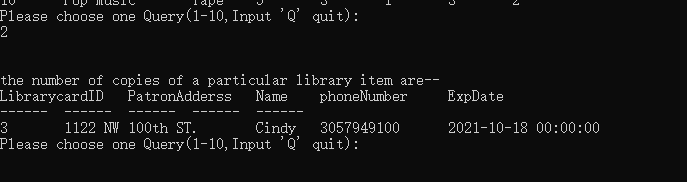
Implementation result in java



select \*

from item

1,It list the numbers and copies of each item in database.



select p.\*

from borrow b

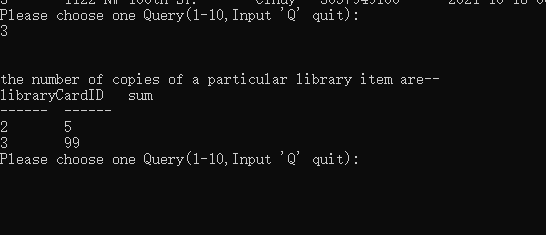
left join patrons p

on p.libraryCardID = b.libraryCardID

where TrueReturnDate is null

and ExpReturnDate < sysdate;

2, For finding details of the patrons who have at least an overdue library item today. We need to satisfy two conditions which is it hasn’t been return and it is already overdue. So in sql script it was last two line from my corresponding script.



select b.libraryCardID,sum(cost) as sum

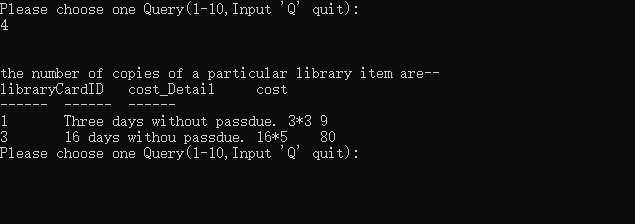
from borrow b, patrons p

where p.libraryCardID = b.libraryCardID

and b.checks = 0

group by b.libraryCardID;

3, In order to find the fine. We need to output the total cost for those borrow’s checks be 0. As the details said in borrow 2 and 4. Those patrons hasn’t paid yet. So the corresponding checks are 0 and their fine generated by such formula : borrowing days(in expected) \* payment + borrowing days(passdue)\*(payment + fine).



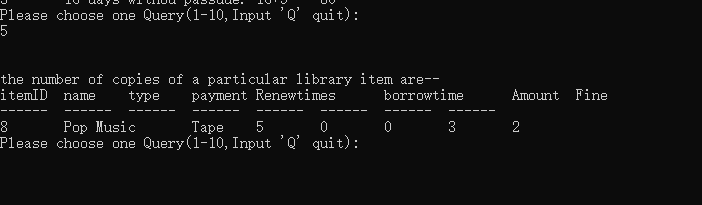
select b.libraryCardID,b.cost\_Detail,b.cost

from borrow b, patrons p

where p.libraryCardID = b.libraryCardID

and b.checks = 1;

4, To find out a patron’s payment. First we must make sure this guy has done the payment. Which means their checks should be 1. Then we output their total cost.



select i.\*

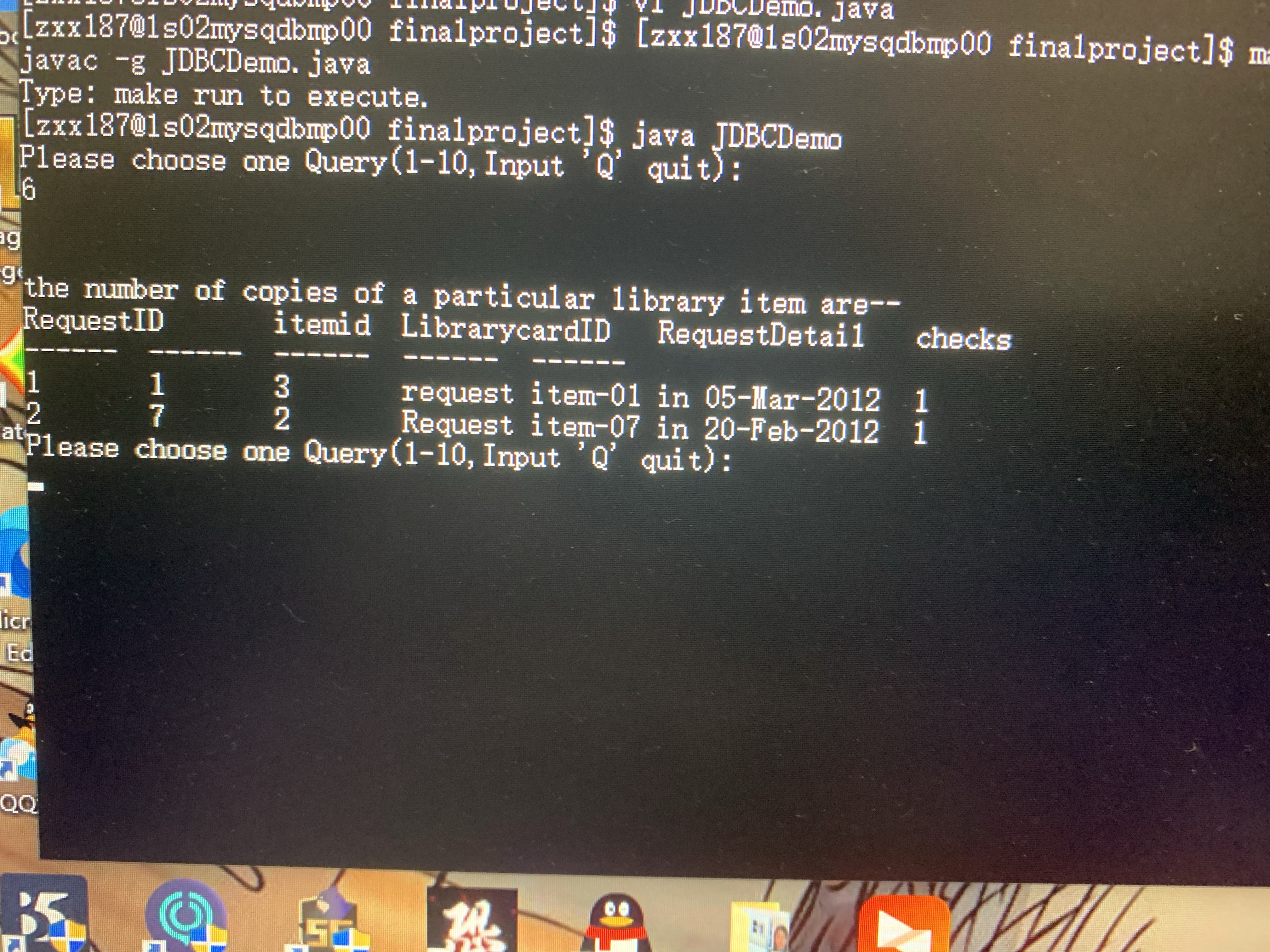
from borrow b,item i

where i.itemid = b.itemid

and TrueReturnDate is null

and ExpReturnDate < sysdate;

5, In order to find an over due item, we did same as in query2. We first check the truereturndate is null, which means item hasn’t been return yet. Second we must make sure it is over due, which means the expreturndate smaller than sys date in script.

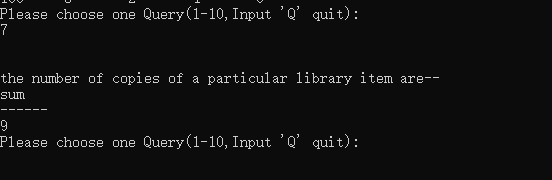


select \*

from request

where checks = 1;

6, To finding pending request, we must firstly make sure the checks be 1. Because if the checks is 0, as I discussed before, that means this request conflicts with the existing borrowing in borrow table. In this way, we can filter “useless request” from patrons and output actual requests. In this case the request 3 was denied by the system.



select sum(cost) as sum

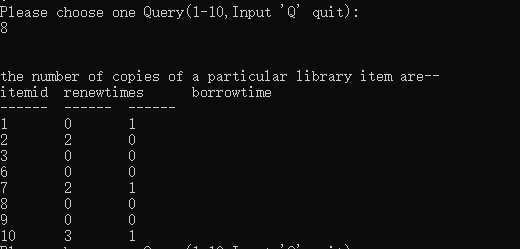
from borrow b

where b.checks = 1;

and b.TrueReturnDate > to\_date('2012-03-01','yyyy-mm-dd')

and b.TrueReturnDate < to\_date('2012-03-21','yyyy-mm-dd') ;

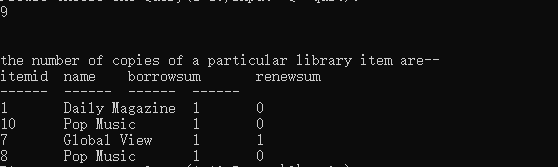
7, Still, we need to check the checks be 1 to confirm the payment has done and the library received the payment first. Which was b.checks = 1. Then we input the request date mentioned in pdf.



select itemid,renewtimes,borrowtime

from item;

8, It directly list the renew times and borrow times for each item.



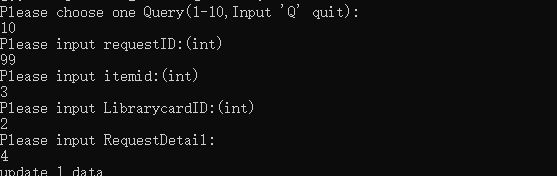
select i.itemid,i.name,count(\*) as borrowsum,sum(b.renewtimes) as renewsum

from borrow b,item i

where b.itemid = i.itemid

group by i.itemid,i.name;

9, It lists the total number and the details of library items that are checked out and renewed by a patron.



10, It can insert a request from user. However, if the request id conflicts with the exiting request id, the system will throw an error. Same case if the librarycardid is beyond the existing table.

This report fully explain the details of the whole project and the principles of implementation.

EER diagram.

