UNIVERSITY LIBRARY
CSC 540 PROJECT REPORT

(**GROUP: 21**. Ashutosh Chaturvedi ● Ayush Gupta ● Jordy Jose ● Priysha Pradhan ● Srishti Arora)

Table of Contents

1. Problem Statement:

University Library is desktop application to assist with the administration of the library. The application allows library patrons to use the resources available in the two libraries present on campus the D.H Hill Library and the J.B. Hunt Library. Each of the libraries contains certain books, journals, E books, some study/conference rooms and devices like cameras.

2. Entities and Relationships:

2.1 Entities:

1. Users
   This entity has user id and password. This table contains user id for all faculties and students who have activated their account.

2. Students
   The entity has student number, first name, last name, home phone, work phone, date of birth, sex, nationality, department id, street, city, pin code, student classification, degree program and student program.

3. Faculty
   The entity has faculty number, first name, last name, category, nationality and department id.

4. Courses
   The entity has course id, course name and faculty no.

5. Departments
   The entity has the department id and name.

6. Addresses
   The entity has street, city and pin code.

7. Libraries
   The entity contains library id and library name.

8. Ecopy Books
   The entity has ISBN, edition, publisher, author, tittle, and year.

9. Ecopy Journals
   The entity has ISBN, author, tittle and year.

10. Ecopy Conference
    The entity has conference number, conference name, author, year and tittle.

11. Hardcopy Books
    The entity has ISBN, edition, publisher, author, tittle, number of copies and year.

12. Hardcopy Journals
    The entity has ISBN, author, number of copies, tittle and year.

13. Hardcopy Conference
    The entity has conference number, conference name, author, number of, year and tittle.

14. Study-Rooms
    The entity contains room number, floor, capacity, library id.

15. Conference-Rooms
    The entity contains room number, floor, capacity, library id.

16. Cameras
    The entity has camera id, make, model, memory, configuration, library id, is available.

17. Master Messages
    The entity has master message id, master message subject, master message text. This table contains static messages.

18. Dues
    The relationship contains user id, amount, resource id, due date, paid date, user name.

2.2 Relationships:

1. Reserves
   This relationship captures faculty and course. It has faculty number, course id, ISBN, expiry date and start date.

2. Enrollment
   The relationship has course id and student number and captures relation between courses and student.

3. Book Waitlist
   The relationship has user id, book id, date waitlisted and is faculty and it captures relation between user and book.

4. Camera Request
   The relationship has user id, start id, end date, check out time, check out end time, check in date time and it captures relation between user and cameras.

5. Camera Wait Queue
   The relationship has user id, date waitlist and expected reservation date.

6. Checks Out Ecopy Book
   The relationship has user id, ISBN, checks out date and time and return date and time it captures relation between user and book.

7. Checks Out Ecopy Conference
   The relationship has user id, conference number, checks out date and time and return date and time. It captures relation between user and conference proceedings.

8. Checks Out Ecopy Journal
   The relationship has user id, ISSN, checks out date and time and return date and time. It captures relation between user and journals.

9. Conference Room Reservation

The relationship contains user id, room number, library id, start date, end date, check out deadline, check out date and time and is void. It captures relation between faculty and conference room.

10. Hold User
    The relationship contains hold id, user id and hold date. This table contain users whose account got terminated because of dues.

11. Patron Message Notifications
    The relationship contains patron message number, patron id, master message id, message generated time and read flag. This table is queried to receive notifications specific to the user.

12. Study Room Reservation
    The relationship contains user id, room number, library id, start date, end date, check out deadline, check out date time and check in date time. It captures relation between user and conference room.

3. Users:

3.1 Student:

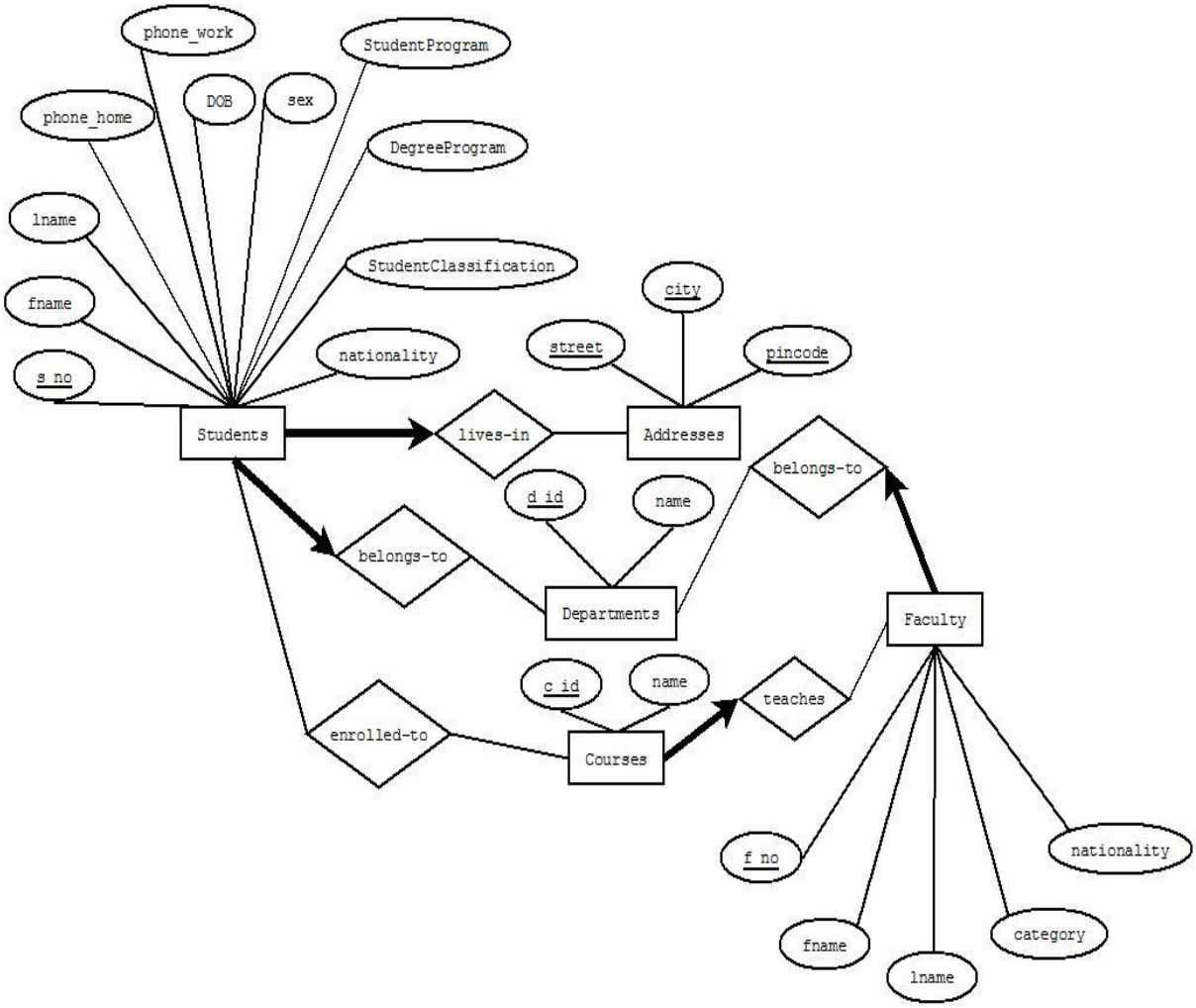
Student can request and checkout resources such as reserved books, other books, electronic publications, journals and conference proceedings, study rooms and cameras from the library.

3.2 Faculty:

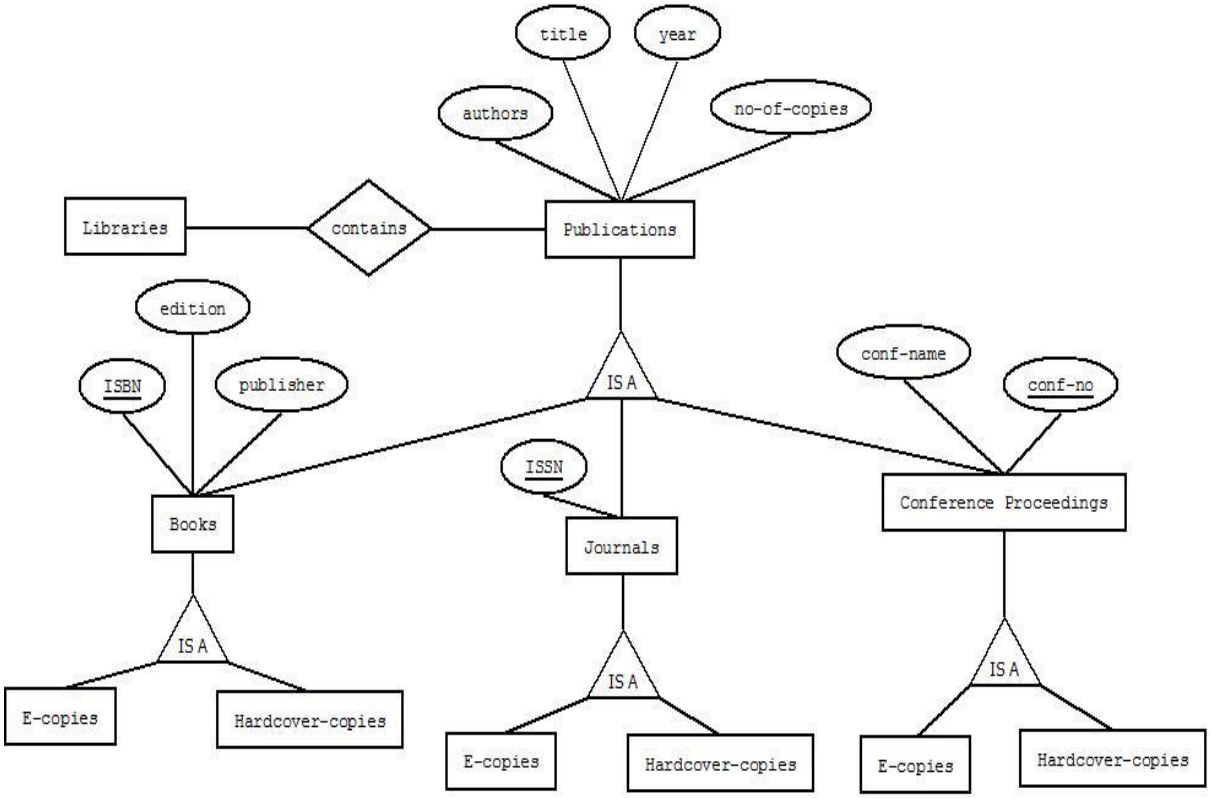
Faculty can request and checkout resources such as books, electronic publications, journals and conference proceedings, study rooms, conference rooms and cameras from the library. Faculty can also reserve a book for a course.

4. E-R Diagram:

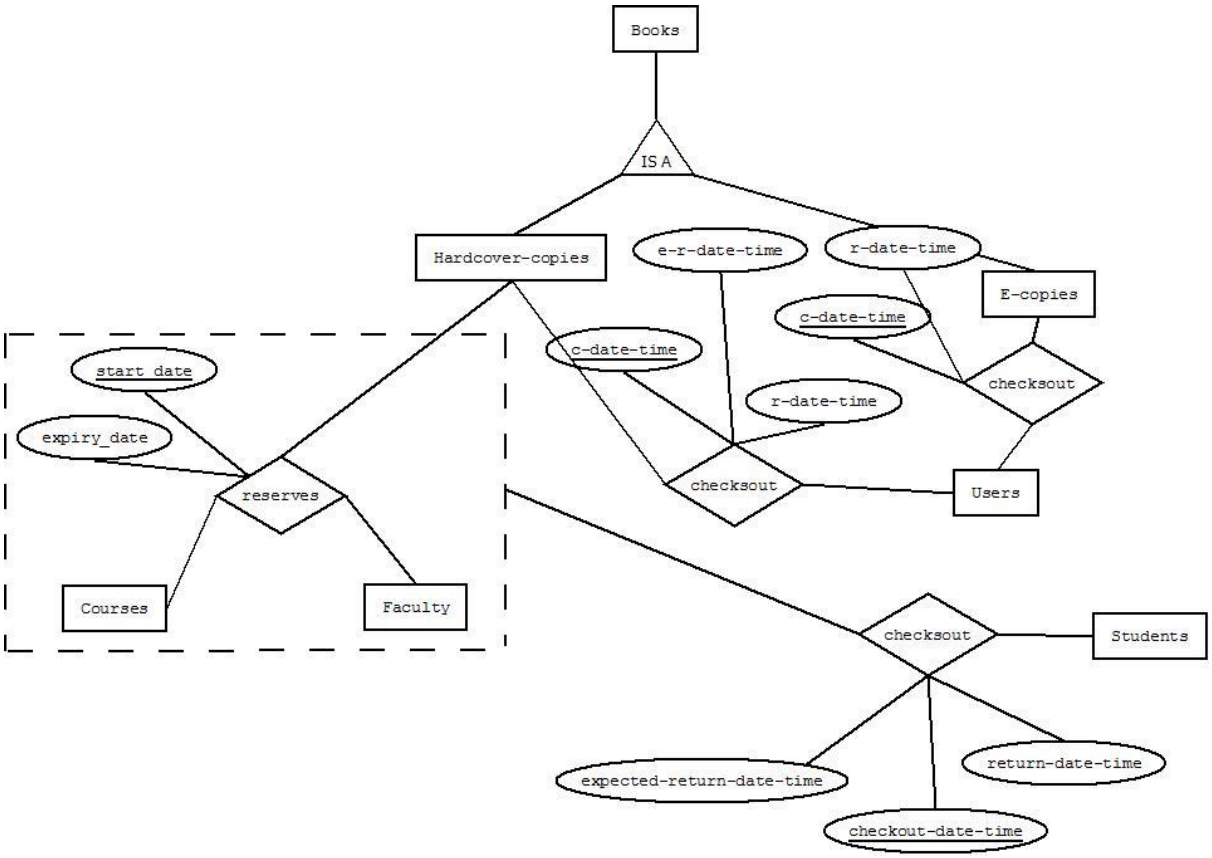
1. Library Patrons:



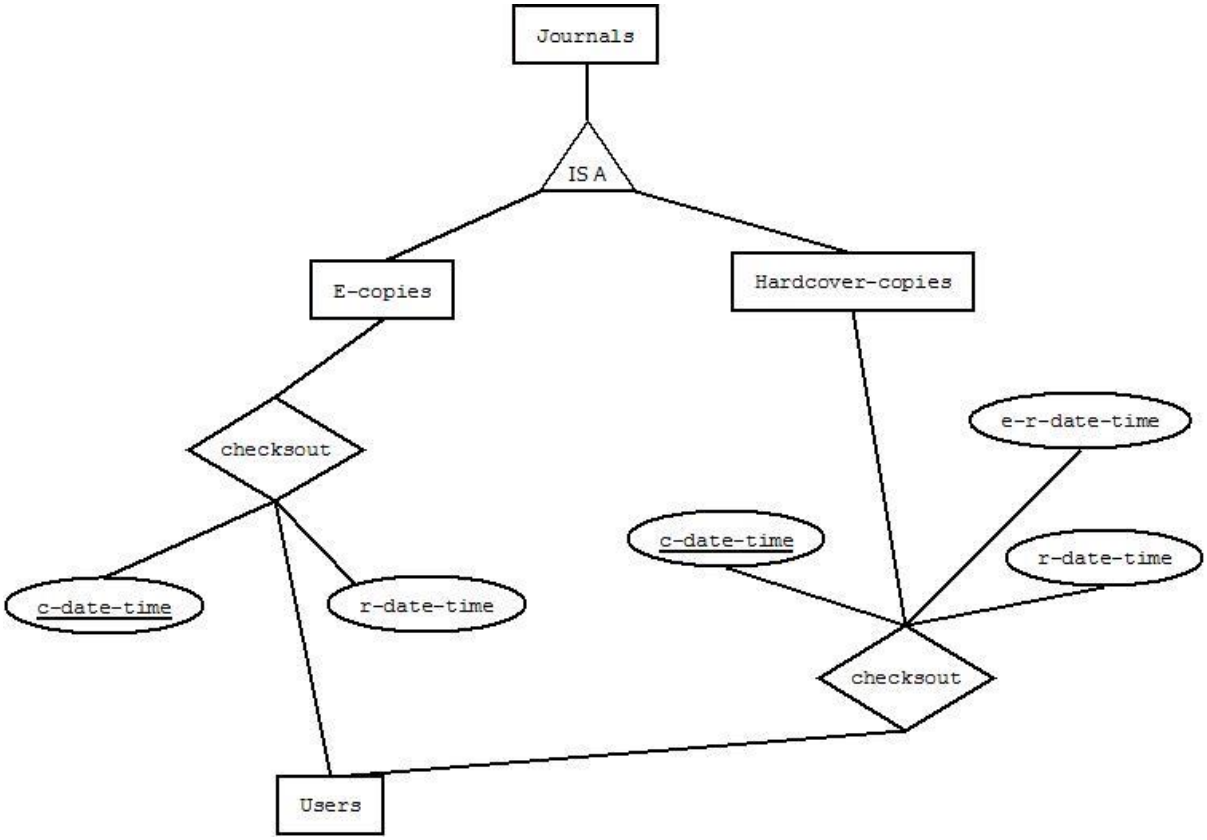
2. Publications:



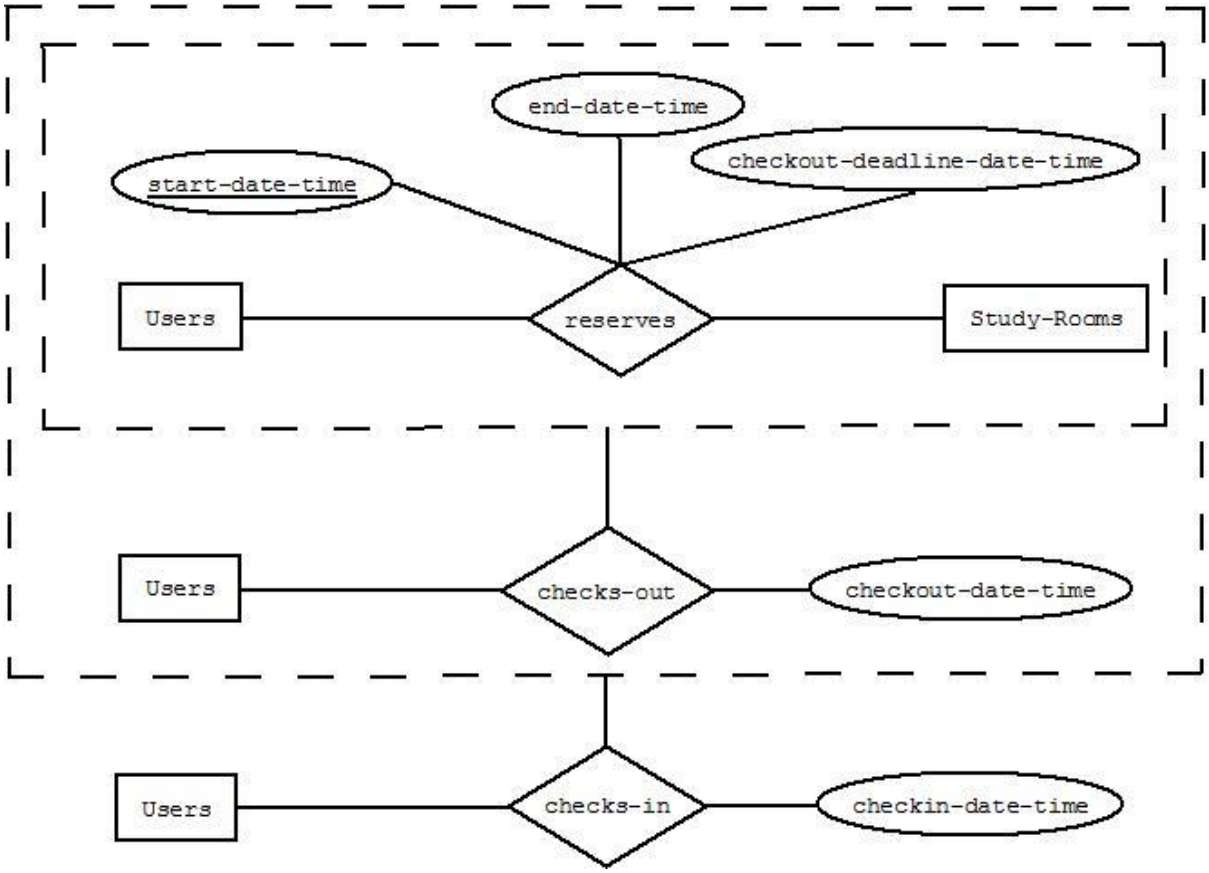
3. Books Checkout:



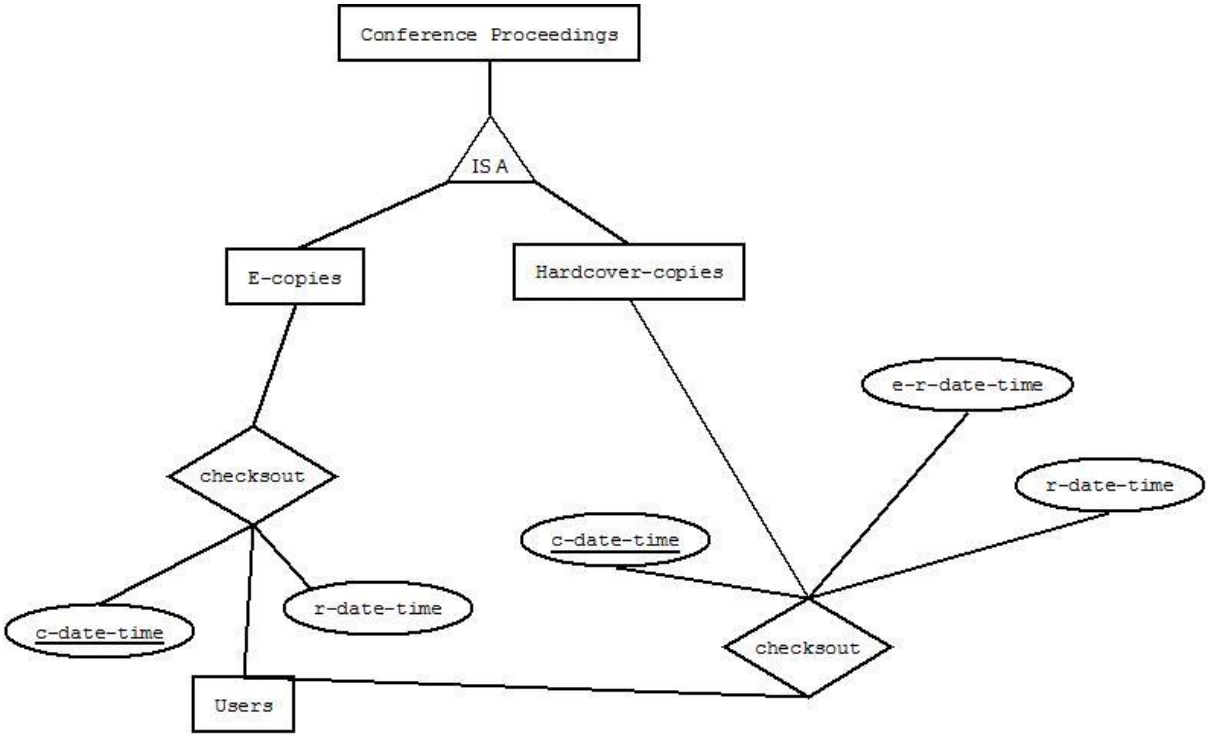
4. Journals Checkout:



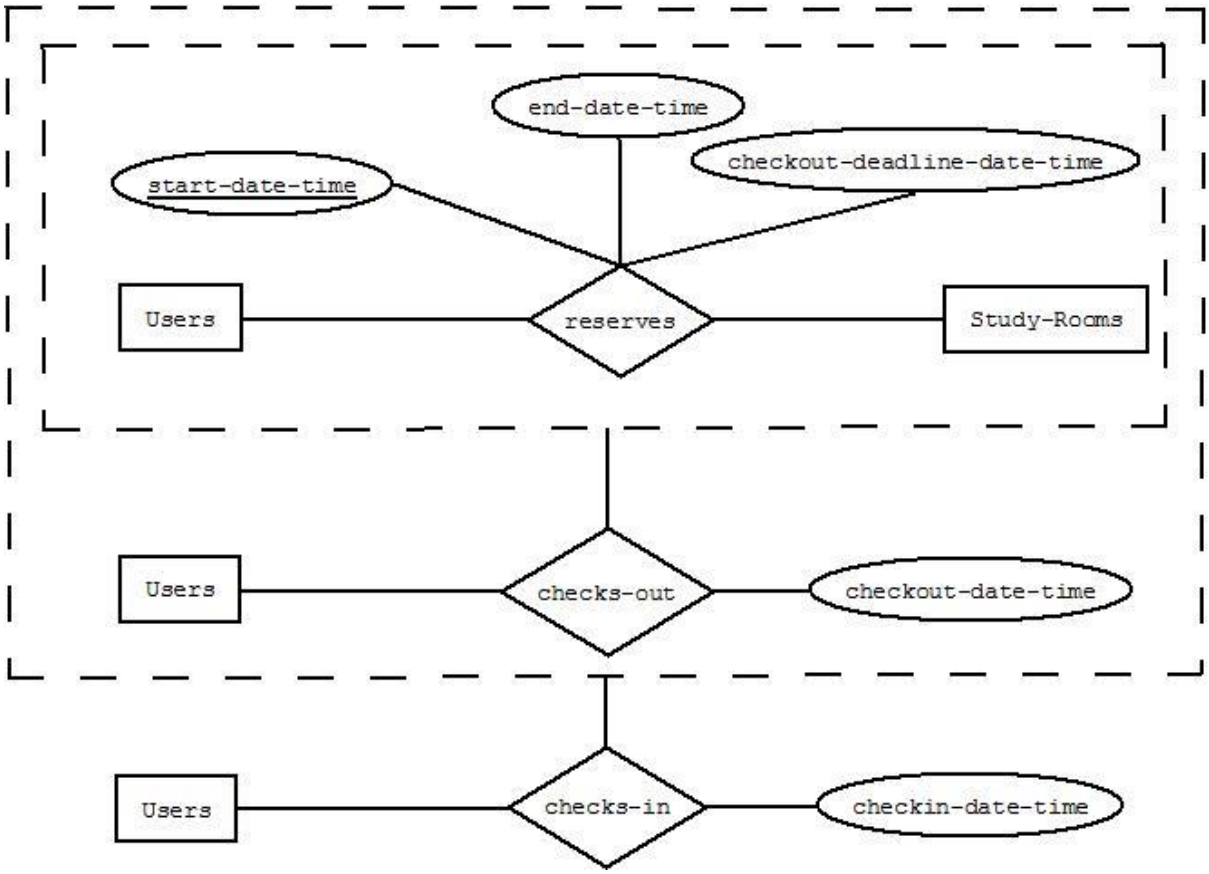
5. Study Rooms:



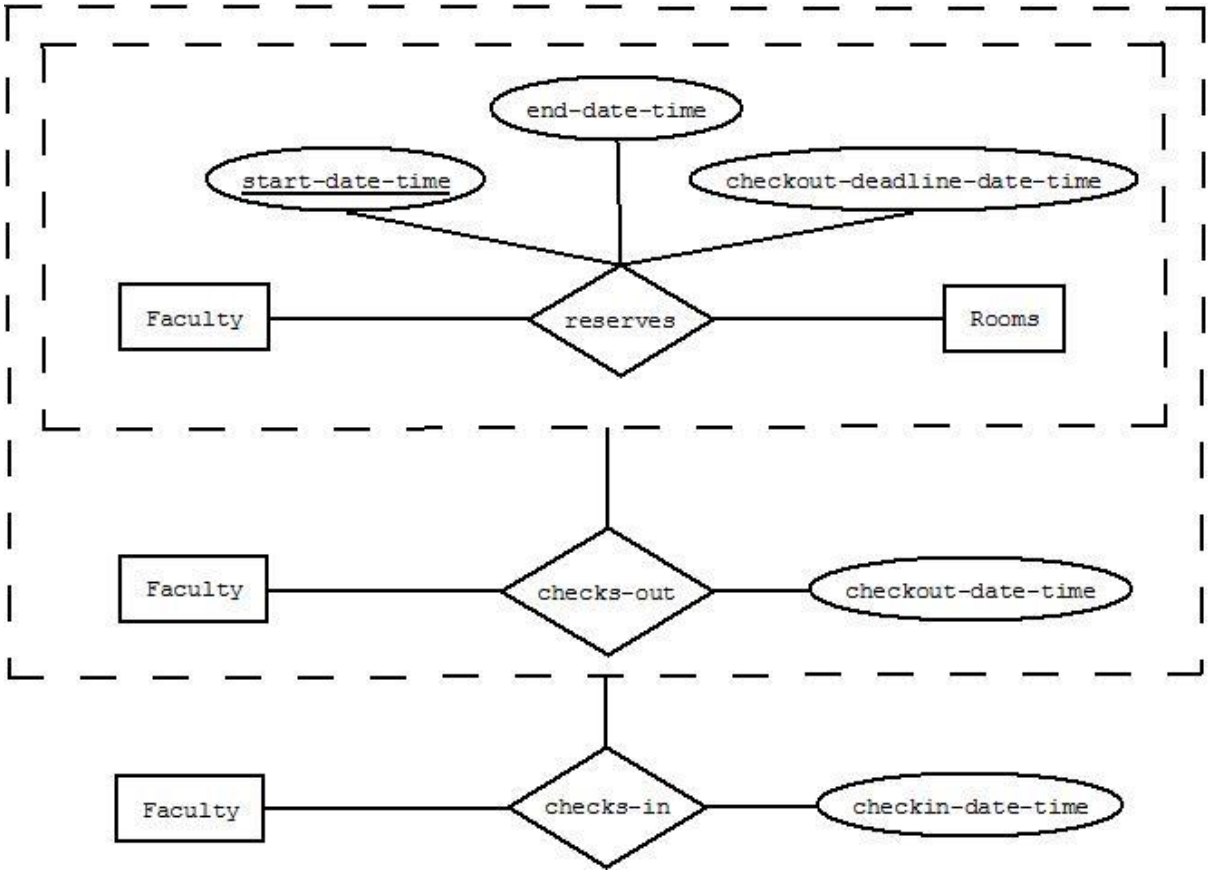
6. Conference Proceedings Checkout:



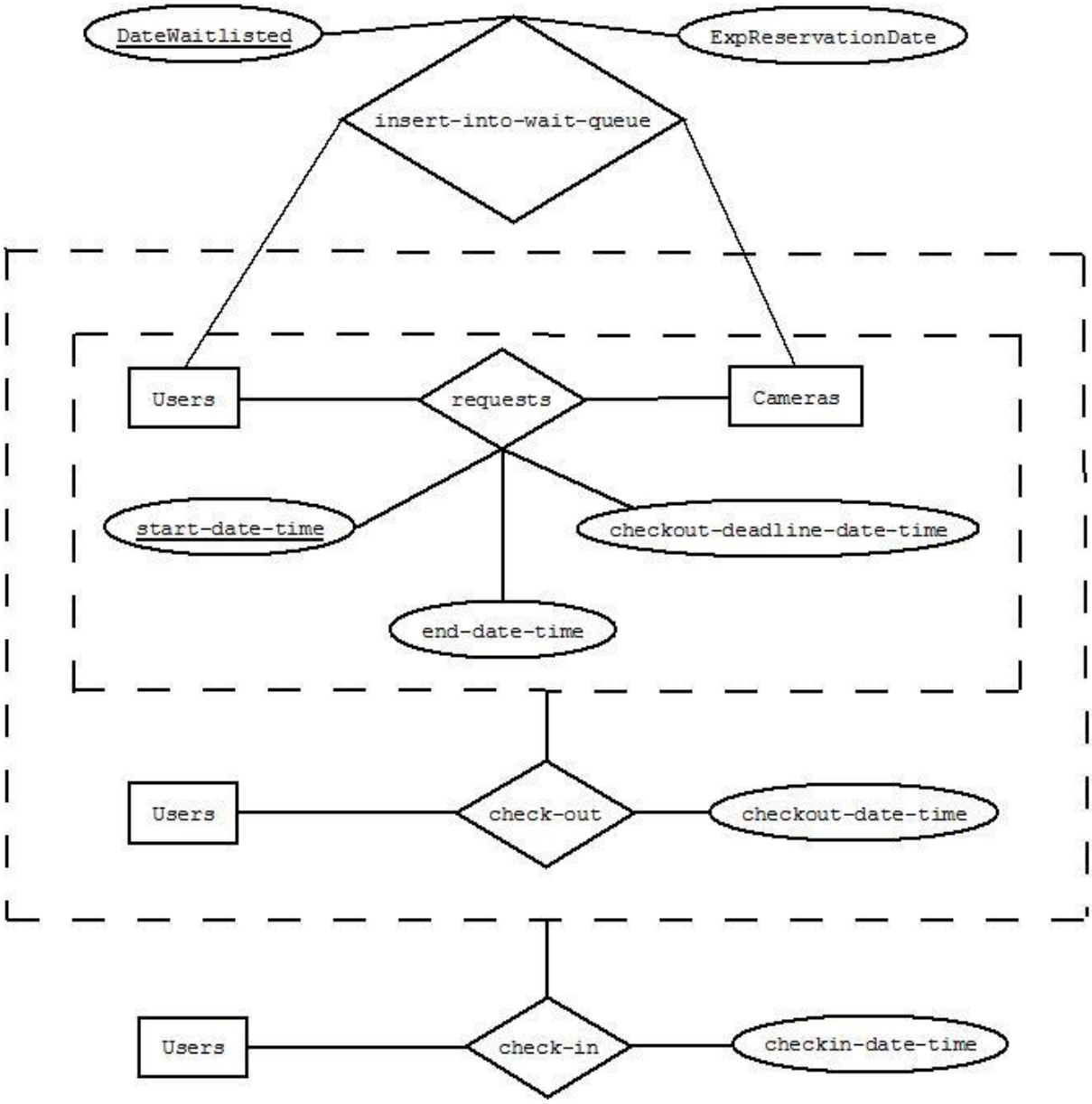
7. Study Room Reservation:



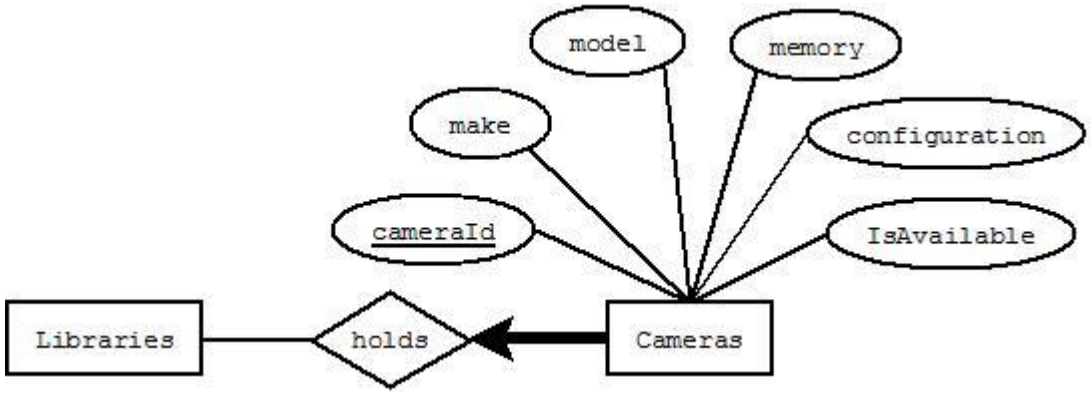
8. Faculty Room Reservation:



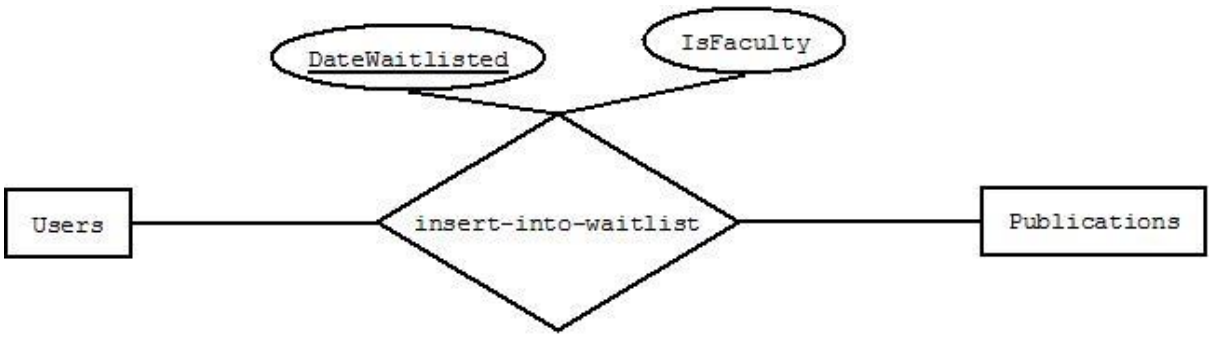
9. Student Camera Reservation:



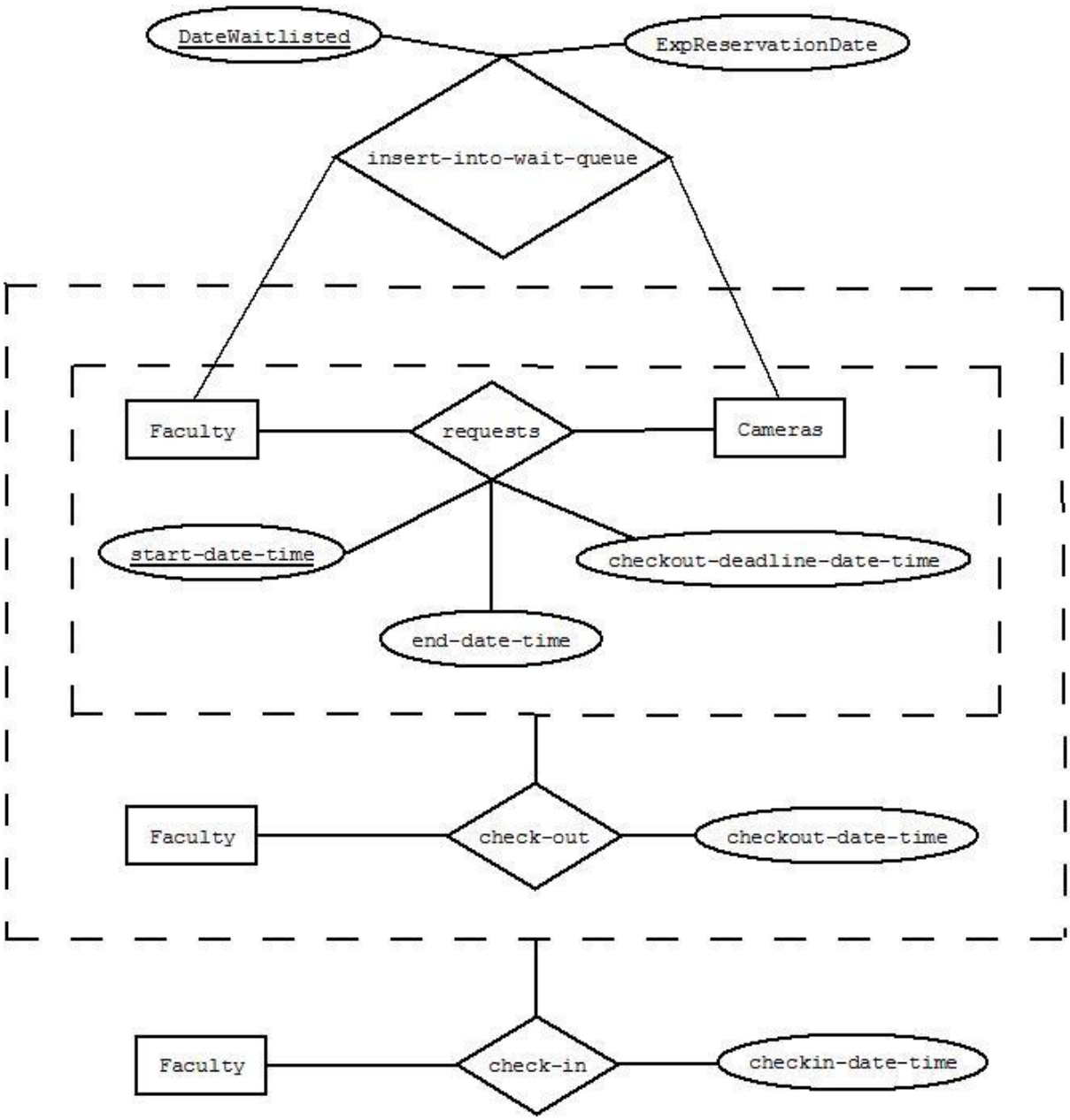
10. Camera Reservation:



11. Waitlist:



12. Faculty Camera Reservation:



6. Relationship Schema:

1. Faculty (*facultyNo:* **integer**, *firstName:* **varchar**, *lastName:* **varchar**, *category:* **varchar**, *nationality:* **integer**, *deptId:* **varchar**)
   deptId REFERENCES Departments(*deptId*)

2. Students (*studentNo:* **intxeger**, *firstName:* **varchar**, *lastName:* **varchar**, *homePhone:* **integer**, *workPhone:* **integer**, *dob:* **datetime**, *sex:* **varchar**, *nationality:* **varchar**, *deptId:* **varchar,** *street:* **varchar**, *city:* **varchar**, *pincode:* **varchar**)
   deptId REFERENCES Departments(*deptId*)
   street REFERENCES Addresses(*street*)
   city REFERENCES Addresses(*city*)
   pincode REFERENCES Addresses(*pincode*)

3. Addresses (*street:* **varchar**, *city:* **varchar**, *pincode:* **varchar**)

4. Departments (*deptId:* **varchar**, *name:* **varchar**)
5. Libraries (*libraryId:* **integer**, *name:* **varchar**)
6. Courses (*courseId:* **varchar**, *courseName:* **varchar**, *name:* **varchar**, *facultyNo:* **integer**)
   facultyNo REFERENCES Faculty (*facultyNo*)

7. EcopyBooks (*ISBN:* **integer**, *edition:* **varchar**, *publisher:* **varchar**, *author:* **varchar**, *title:* **varchar**, *noOfCopies:* **integer**, *year:* **integer**)

8. HardcopyBooks (*ISBN:* **integer**, *edition:* **varchar**, *publisher:* **varchar**, *author:* **varchar**, *title:* **varchar**, *noOfCopies:* **integer**, *year:* **integer**)

9. EcopyJournals (*ISSN:* **integer**, *author:* **varchar**, *title:* **varchar**, *noOfCopies:* **integer**, *year:* **integer)**

10. HardcopyJournals (*ISSN:* **integer**, *author:* **varchar**, *title:* **varchar**, *noOfCopies:* **integer**, *year:* **integer)**

11. EcopyConf (*confNo:* **integer**, *confName:* **varchar**, *author:* **varchar**, *title:* **varchar**, *noOfCopies:* **integer**, *year:* **integer**)

12. HardcopyConf (*confNo:* **integer**, *confName:* **varchar**, *author:* **varchar**, *title:* **varchar**, *noOfCopies:* **integer**, *year:* **integer**)

13. Camera (*cameraId:* **integer**, *make:* **varchar**, *model:* **varchar**, *memory:* **varchar**, *configuration:* **varchar**, *isAvailable:* **varchar**, *libraryId:* **integer**)
    libraryId REFERENCES Libraries (*libraryId*)

14. StudyRooms (*roomNo:* **varchar**, *floor:* **integer**, *capacity:* **integer**, *libraryId:* **integer**)
    libraryId REFERENCES Libraries (*libraryId*)

15. StudyRoomsReservation (userId: **varchar**, *roomNo:* **varchar**, *libraryId:* **integer,** *startDate:* **date,** *endDate:* **date**)
roomNo, libraryId REFERENCES StudyRooms (*roomNo, libraryId*)
userId REFERENCES User (*userId*)

16. ConferenceRooms (*roomNo:* **varchar**, *floor:* **integer**, *capacity:* **integer**, *libraryId:* **integer**)
libraryId REFERENCES Libraries (*libraryId*)

17. Duration (*startDate:* **datetime**, *endDate:* **datetime**)

18. Reserves (*facultyNo:* **integer**, *courseId:* **integer**, *ISBN:* **integer**, *expiryDate:* **datetime**, *startDate:* **datetime**)
facultyNo REFERENCES Faculty(facultyNo)
courseId REFERENCES Courses(*courseId)*
ISBN REFERENCES HardcopyBooks(*ISBN)*

19. Camera Request (*userId:* **varchar**, *cameraId:* **varchar,** *startDate:* **date,** *endDate:* **date,** *:* **date,** *checkoutEndTime***: date,** *checkinDateTime:* **date**)
cameraId REFERENCES Cameras(*cameraId*)
userId REFERENCES User (*userId*)

20. CameraWaitQueue (*userId:* **varchar**, *cameraId:* **varchar,** *dateWaitListed:* **date,** *expReservationDate:* **date**)
cameraId REFERENCES Cameras(*cameraId*)
userId REFERENCES User (*userId*)

21. ChecksoutEcopyBook (*userId:* **varchar**, *ISBN:* **varchar,** *cDateTime:* **date,** *rDateTime:* **date**)
ISBN REFERENCES EcopyBooks (*ISBN*)
userId REFERENCES User (*userId*)

22. ChecksoutEcopyConf (*userId:* **varchar**, *confNo:* **varchar,** *cDateTime:* **date,** *rDateTime:* **date**)
confNo REFERENCES EcopyConf (*confNo*)
userId REFERENCES User (*userId*)

23. ChecksoutEcopyJournal (*userId:* **varchar**, *ISSN:* **varchar,** *cDateTime:* **date,** *rDateTime:* **date**)
ISSN REFERENCES HardcopyJournals(*ISSN*)
userId REFERENCES User (*userId*)

24. ChecksoutHardcopyBook (*userId:* **varchar**, *ISBN:* **varchar,** *cDateTime:* **date,** *rDateTime:* **date,** *erDateTime:* **date**)
ISBN REFERENCES HardcopyBooks (*ISBN*)
userId REFERENCES User (*userId*)

25. ChecksoutHardcopyConf (*userId:* **varchar**, *confNo:* **varchar,** *cDateTime:* **date,** *rDateTime:* **date,** *erDateTime:* **date**)
confNo REFERENCES HardcopyConf (*confNo*)

userId REFERENCES User (*userId*)

26. ChecksoutHardcopyJournal (*userId:* **varchar**, *ISSN:* **varchar,** *cDateTime:* **date,** *rDateTime:* **date,** *erDateTime:* **date**)
ISSN REFERENCES HardcopyJournals(*ISSN*)
userId REFERENCES User (*userId*)

27. ConferenceRoomReservation (*userId:* **varchar**, *roomNo:* **varchar,** *libraryId:* **varchar,** *startDate:* **date,** *endDate:* **date,** *checkoutDeadline:* **date,** *checkoutDateTime:* **date,** *checkinDateTime:* **date,** *isVoid:* **varchar**)
roomNo, libraryId REFERENCES StudyRooms (*roomNo, libraryId*)
userId REFERENCES User (*userId*)

28. Enrollment (*courseId:* **varchar**, *studentNo:* **integer**)
courseId REFERENCES Courses (*courseId*)

29. BookWaitlist (*userId:* **varchar**, *bookId:* **varchar,** *dateWaitlisted:* **date,** *isFaculty:* **char**)
userId REFERENCES User (*userId*)

30. Dues (*userId:* **varchar**, *resourceId:* **varchar,** *Amount:* **integer,** *dueDate:* **date,** *paidDate:* **date, ,** *isPaid:* **char**)

31. HoldUser (*userId:* **varchar**, *holdId:* **integer,** *holdDate:* **date**)
userId REFERENCES User (*userId*)

32. MasterMessages (*masterMessageId:* **varchar**, *masterMessageSubject:* **varchar,** *masterMessageText:* **varchar**)

33. PatronMessageNotifications (patronMessageNo: **integer,** *patronId:* **varchar**, *masterMessageId:* **varchar,** *readFlag:* **varchar**)
masterMessageId REFERENCES MasterMessages (*masterMessageId*)

7. Functional Dependencies:

RELATION	FUNCTIONAL DEPENDENCY
STUDENTS	studentNo, deptId, street, city, pincode -> firstName, lastName, homePhone, workPhone, dob, sex, nationality
COURSES	courseId, facultyNo -> name
FACULTY	facultyNo, deptId -> fisrtName, lastName, category, nationality
DEPARTMENTS	deptId -> name
CONFERENCE PROCEEDINGS	confNo -> confName
ECOPYBOOKS	ISBN -> edition, publisher, author, title, noOfCopies, year
HARDCOPYBOOKS	ISBN -> edition, publisher, author, title, noOfCopies, year
ECOPYJOURNALS	ISSN -> edition, publisher, author, title, noOfCopies, year
HARDCOPYJOURNALS	ISSN -> edition, publisher, author, title, noOfCopies, year
ECOPYCONF	confNo -> confName, author, title, noOfCopies, year
HARDCOPYCONF	confNo -> confName, author, title, noOfCopies, year
STUDYROOMS	roomNo, libraryId -> floor, capacity
STUDYROOMSRESERVATION	roomNo, libraryId, userId, startDate -> endDate
CONFERENCE ROOMS	roomNo, libraryId -> floor, capacity
LIBRARIES	Id -> name
BOOKWAITLIST	facultyId, dateWailisted, userId -> isFaculty
CAMERA	cameraId, libraryId -> make, model, memory, configuration, isAvailable,
CAMERAREQUESTS	userId, cameraId, startdate -> userId, cameraId , endDate, checkoutEndTime, checkinDateTime
CAMERAWAITQUEUE	userId, cameraId, daitWaitlisted -> expReservationDate
STUDYROOMRESERVATION	userId,
LIBRARIES	LibraryId - > name
RESERVES	facultyNo , courseId, ISBN -> expiryDate, startDate
CHECKSOUTECOPYBOOK	userId, ISBN, cDateTime -> rDateTime
CHECKSOUTECOPYCONF	userId, confNo, cDateTime -> rDateTime
CHECKSOUTECOPYJOURNAL	userId, ISSN, cDateTime -> rDateTime
CHECKSOUTHARDCOPYBOOK	userId, ISBN, cDateTime -> rDateTime, erDateTime

CHECKSOUTHARDCOPYCONF	userId, confNo, cDateTime -> rDateTime, erDateTime
CHECKSOUTHARDCOPYJOURNAL	userId, ISSN, cDateTime -> rDateTime, erDateTime
CONFERENCEROOMRESERVATION	userId, roomNo, startDate -> libraryId, endDate, checkoutDeadline, checkoutDateTime, checkinDateTime, isVoid
DUES	userId, resourceId, dueDate -> Amount paidDate, isPaid
BOOKWAITLIST	userId, dateWaitlisted -> bookId, isFaculty
HOLDUSER	UserId -> holdId, holdDate
MASTERMESSAGES	MasterMessageId -> masterMessageSubject, masterMessageText
PATRONMESSAGENOTIFICATIONS	patronMessageNo, MasterMessageId -> patronId, readFlag

All the FD of all the relation are either trivial or the attributes on left side is the primary keys of the relation. So we can conclude that all the relations are in BCNF.

8. Constraints:

There are few constraints which were implemented in Java rather than in the DBMS. They are as following:

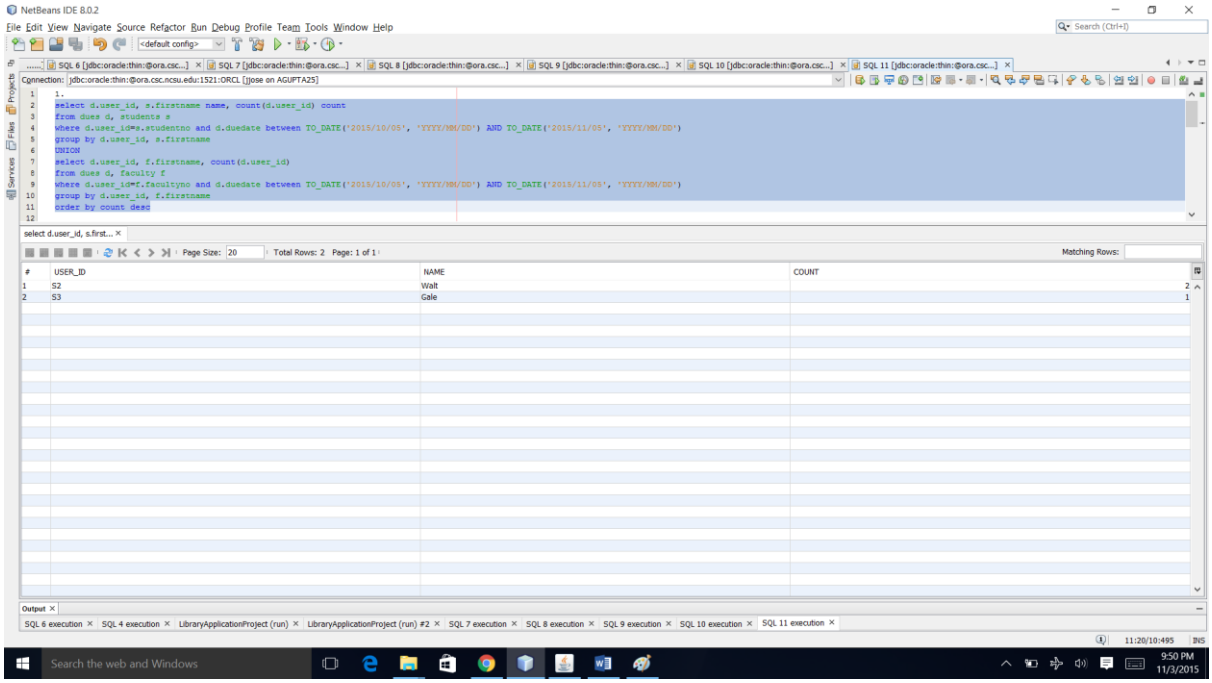
1. A camera can be reserved only on a Friday.

   A patron can reserve cameras. However the reservations are allowed only on Fridays. One reason not to implement this constraint in the DBMS was, this is not a very intuitive use case and the user of the application would like to receive a notification immediately if he/ she attempts to reserve a camera on a day other than Friday. Implementing the constraint in the DBMS means the java code will have to catch the right exception, interpret it and show the right error message.

9. SQL Queries:

1. **List in descending order the names of the patrons and the number of times they failed to return a resource on time from Oct 5th to Nov 5th, 2015.**

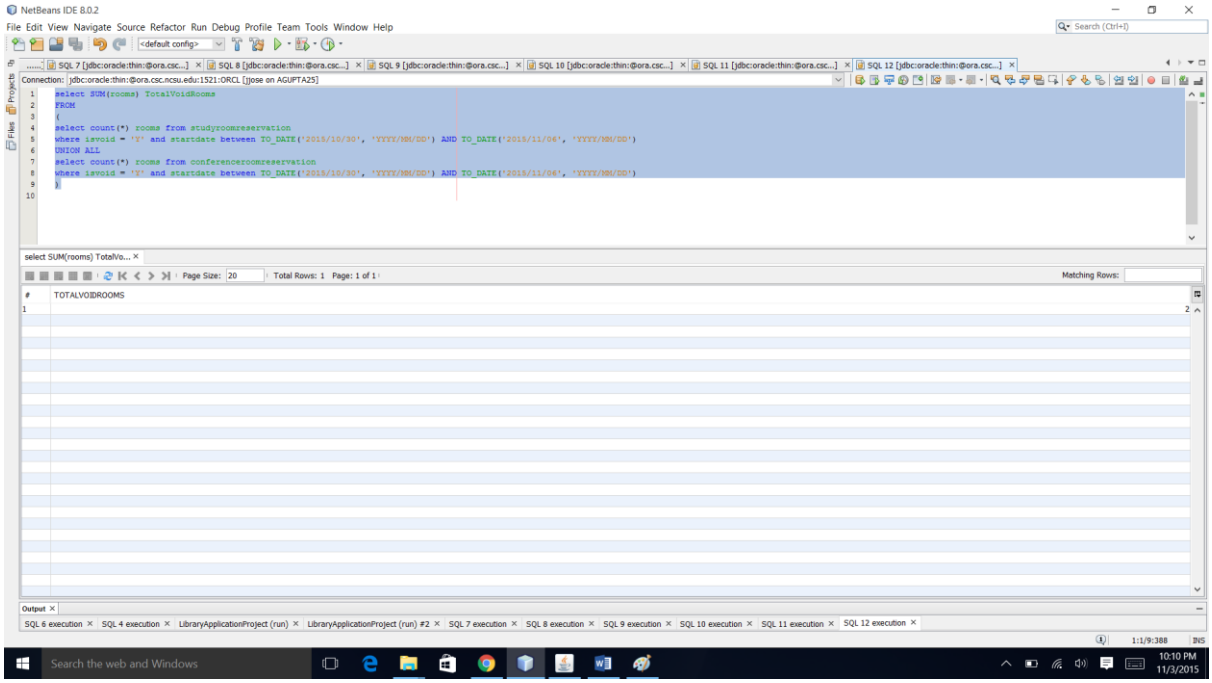
```
select d.user_id, s.firstname name, count(d.user_id) count
from dues d, students s
where d.user_id=s.studentno and d.duedate between TO_DATE('2015/10/05',
'YYYY/MM/DD') AND TO_DATE('2015/11/05', 'YYYY/MM/DD')
group by d.user_id, s.firstname
UNION
select d.user_id, f.firstname, count(d.user_id)
from dues d, faculty f
where d.user_id=f.facultyno and d.duedate between TO_DATE('2015/10/05',
'YYYY/MM/DD') AND TO_DATE('2015/11/05', 'YYYY/MM/DD')
group by d.user_id, f.firstname
order by count desc;
```



2. **List the number of room reservations from Oct 30th to Nov 6th, 2015 that were rendered null and void because the patron failed to check in within the first one hour.**

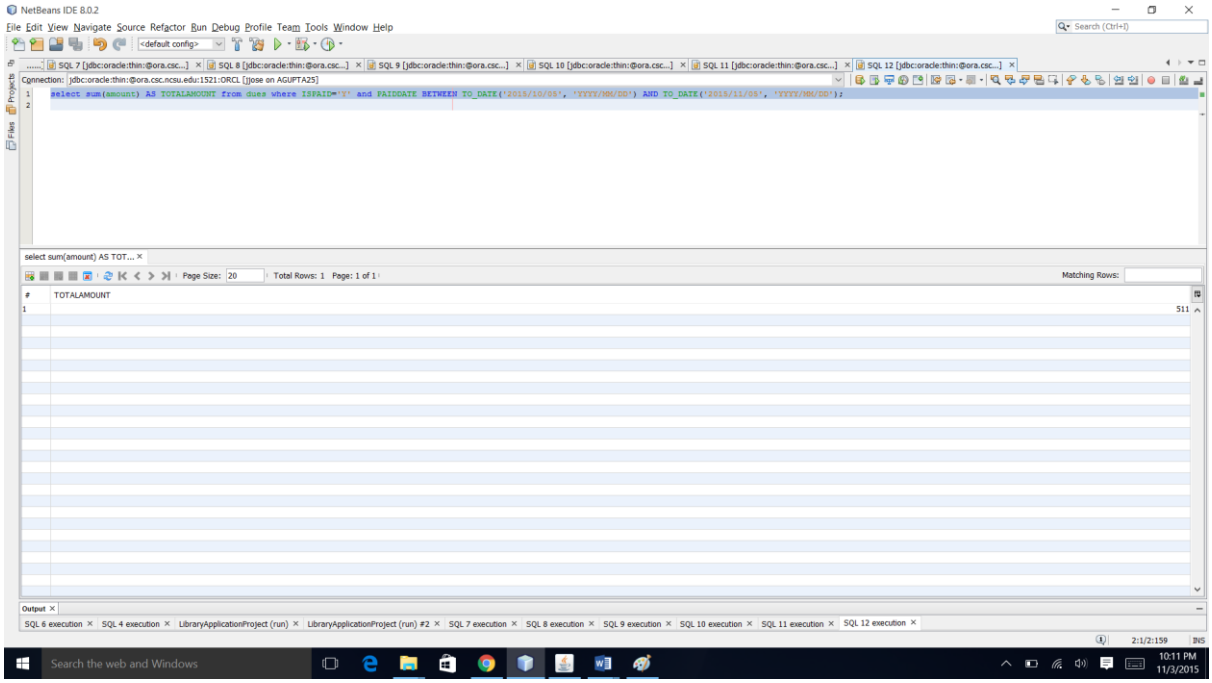
```
select SUM(rooms) TotalVoidRooms
FROM
```

```
(
select count(*) rooms from studyroomreservation
where isvoid = 'Y' and startdate between TO_DATE('2015/10/30', 'YYYY/MM/DD')
AND TO_DATE('2015/11/06', 'YYYY/MM/DD')
UNION ALL
select count(*) rooms from conferenceroomreservation
where isvoid = 'Y' and startdate between TO_DATE('2015/10/30', 'YYYY/MM/DD')
AND TO_DATE('2015/11/06', 'YYYY/MM/DD')
);
```



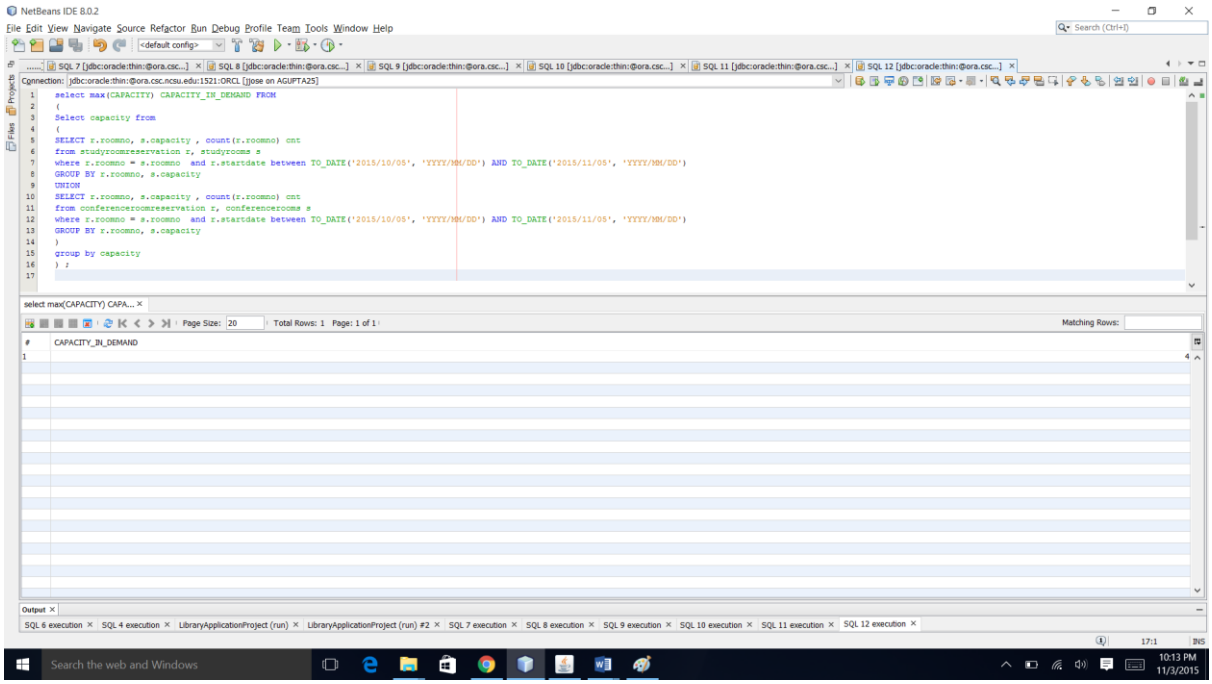
3. **What is the total amount of money that the library received between Oct 5th to Nov 5th, 2015 in terms of late fees for all resources from all patrons.**

select sum(amount) AS TOTALAMOUNT from dues where ISPAID='Y' and PAIDDATE BETWEEN TO_DATE('2015/10/05', 'YYYY/MM/DD') AND TO_DATE('2015/11/05', 'YYYY/MM/DD');



4. Using the number of requests between Oct 5th to Nov 5th, 2015 as a parameter determine what capacity rooms are in maximum demand.

```
select max(CAPACITY) CAPACITY_IN_DEMAND FROM
(
Select capacity from
(
SELECT r.roomno, s.capacity , count(r.roomno) cnt
from studyroomreservation r, studyrooms s
where r.roomno = s.roomno and r.startdate between TO_DATE('2015/10/05', 'YYYY/MM/DD') AND TO_DATE('2015/11/05', 'YYYY/MM/DD')
GROUP BY r.roomno, s.capacity
UNION
SELECT r.roomno, s.capacity , count(r.roomno) cnt
from conferenceroomreservation r, conferencerooms s
where r.roomno = s.roomno and r.startdate between TO_DATE('2015/10/05', 'YYYY/MM/DD') AND TO_DATE('2015/11/05', 'YYYY/MM/DD')
GROUP BY r.roomno, s.capacity
)
group by capacity
) ;
```


NetBeans IDE 8.0.2
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
select max(CAPACITY) CAPACITY_IN_DEMAND FROM
(
Select capacity from
(
SELECT r.roomno, s.capacity , count(r.roomno) cnt
from studyroomreservation r, studyrooms s
where r.roomno = s.roomno and r.startdate between TO_DATE('2015/10/05', 'YYYY/MM/DD') AND TO_DATE('2015/11/05', 'YYYY/MM/DD')
GROUP BY r.roomno, s.capacity
UNION
SELECT r.roomno, s.capacity , count(r.roomno) cnt
from conferenceroomreservation r, conferencerooms s
where r.roomno = s.roomno and r.startdate between TO_DATE('2015/10/05', 'YYYY/MM/DD') AND TO_DATE('2015/11/05', 'YYYY/MM/DD')
GROUP BY r.roomno, s.capacity
)
group by capacity
) ;
CAPACITY_IN_DEMAND
4
Search the web and Windows
10:13 PM
11/3/2015