Constraints handled using Triggers and Procedures:

Triggers:

1. CameraCheckoutLogTrigger

```
TRIGGER "CAMERACHECKOUTLOGTRIGGER"
after update of checkintime on CameraCheckout for each row
DECLARE
ReservationId NUMERIC(10);
CheckInTime Timestamp;
CheckOutTime Timestamp;
CameraId varchar(10);
EndTime Timestamp;
StudentNumber varchar(10);
FacultyNumber varchar(10);
Fine1 numeric(10);
date1 timestamp;
date2 timestamp;
dayt numeric(10);
hours numeric(10);
mins numeric(10);
begin
Fine1:=0;
date2:=:new.checkintime;
select c.cameraid,c.endtime,c.studentnumber,c.facultynumber into
CameraId,EndTime,StudentNumber,FacultyNumber from camerareservation c
where c.reservationid=:NEW.reservationid;
select
    extract( day from diff ),
    extract( hour from diff ),
    extract( minute from diff ) into dayt,hours,mins
from (
        select (CAST(date2 as timestamp) - CAST(endtime as timestamp))
diff
        from dual
     );
if dayt >=0 then
Fine1:=dayt*24+hours+CEIL(mins/60);
end if;
  insert into CAMERACHECKOUTLOG(
    RESERVATIONID,
    CAMERAID,
    CHECKINTIME,
    CHECKOUTTIME,
    ENDTIME,
    STUDENTNUMBER,
    FACULTYNUMBER,
    FINE
  )
    values
    (
    :NEW.reservationid,
    CAMERAID,
```

```
    :NEW.CheckInTime,
    :NEW.CheckOutTime,
    ENDTIME,
    STUDENTNUMBER,
    FACULTYNUMBER,
    Fine1
    );
end;
```

Explanation: It makes an entry into the CameraCheckoutLog table when the camera is returned (checked in) and it also calculates the fine if any on that resource transaction. If the student doesnot return the fine within 90 days, his account will be put on hold.

2. FineLogCamera

```
TRIGGER "FINELOGCAMERA"
after insert on CameraCheckoutlog for each row
declare
begin

  if :new.fine>0 then
  insert into finecollected
  values (
    :NEW.studentnumber,
    :NEW.facultynumber,
    'Camera',
    :NEW.cameraid,
    :NEW.fine,
    :NEW.checkintime,
    :NEW.checkouttime,
    :NEW.endtime
  );

end if;
end;
```

Explanation: After the insertion of entry into the CameraCheckoutLog table, if there is a fine incurred, this trigger will update the fine collected table by inserting all the values corresponding to that fine and hence record the PatronId, Checkin time, Checkout time, Endtime, CameraId and calculated fine into the table for that corresponding fine incurred.

3. CheckoutLogTrigger

```
TRIGGER "CHECKOUTLOGTRIGGER"
after update of checkintime on Checkout for each row
declare
fine1 numeric(10);
date1 timestamp;
date2 timestamp;
hold numeric(10);
begin
if updating then
fine1:=0;
date1:=:new.reservedupto;
date2:=:new.checkintime;
select CEIL((CAST (date2 AS DATE)- CAST (date1 AS DATE))) into fine1
from dual;
if fine1 >0 then
fine1:=fine1*2;
end if;
if fine1 <0 then
fine1:=0;
end if;


  insert into CHECKOUTLOG(
    RESOURCEID,
    CHECKOUTTIME,
    RESERVEDUPTO,
    CHECKINTIME,
    STUDENTNUMBER,
    FACULTYNUMBER,
    fine,Typepub,namepub
  )
  values (
    :NEW.RESOURCEID,
    :NEW.CHECKOUTTIME,
    :NEW.RESERVEDUPTO,
    :NEW.checkintime,
    :NEW.studentnumber,
    :NEW.facultynumber,
    fine1,:NEW.typepub, :new.namepub
  );

  select CEIL((CAST (date2 AS DATE)- CAST (date1 AS DATE))) into hold
from dual;
  if :new.facultynumber is null then
      if hold>=90 then
        insert into studenthold values(:new.studentnumber);
      end if;
```

```
   end if;

end if;
end;
```

Explanation: : It makes an entry into the CheckoutLog table when the publication is returned (checked in) and it also calculates the fine if any on that resource transaction. If the student doesnot return the fine within 90 days, his account will be put on hold.

4. FineLog

```
TRIGGER "FINELOG"
after insert on Checkoutlog for each row
declare
begin

  if :new.fine>0 then
  insert into finecollected
  values (
    :NEW.studentnumber,
    :NEW.facultynumber,
    :NEW.typepub,
    :NEW.namepub,
    :NEW.fine,
    :NEW.checkintime,
    :NEW.checkouttime,
    :NEW.reservedupto
  );

end if;
             end;
```

Explanation: After the insertion of entry into the CheckoutLog table, if there is a fine incurred, this trigger will update the fine collected table by inserting all the values corresponding to that fine and hence record the PatronId, Checkin time, Checkout time, Endtime, CameraId and calculated fine into the table for that corresponding fine incurred.

5. RoomCheckoutLogTrigger

```
TRIGGER "ROOMCHECKOUTLOGTRIGGER"
after update of checkintime on Checkoutroom for each row
DECLARE
ReservationId NUMERIC(10);
CheckInTime Timestamp;
CheckOutTime Timestamp;
RoomId varchar(10);
StartTime Timestamp;
EndTime Timestamp;
StudentNumber varchar(10);
FacultyNumber varchar(10);
begin
select c.roomid,c.starttime,c.endtime,c.studentnumber,c.facultynumber
into RoomId,StartTime,EndTime,StudentNumber,FacultyNumber from
roomreservation c where c.reservationid=:NEW.reservationid;
  insert into CheckoutRoomLog(
    RESERVATIONID,
    ROOMID,
    STARTTIME,
     ENDTIME,
    CHECKINTIME,
    CHECKOUTTIME,
    STUDENTNUMBER,
    FACULTYNUMBER
  )
    values
    (
    :NEW.reservationid,
    RoomID,
    STARTTIME,
    ENDTIME,
    :NEW.CheckInTime,
    :NEW.CheckOutTime,
    STUDENTNUMBER,
    FACULTYNUMBER
    );
end;
```

Explanation: When the room is check-in (released), this trigger will update it into the CheckoutRoomLog and records that transaction into the database.

Procedures:

1. Camera Cancellation

```
Procedure CameraCancellation
IS
CURSOR Cameras
    IS
  select * from camerareservation;
```

```
abc camerareservation%ROWTYPE;
count1 number;
timediff number;
dayt numeric;
  hours numeric;
  mins numeric;
  l_total numeric;
  time1 timestamp;
time2 timestamp;
time3 timestamp;
student varchar(10);
faculty varchar(10);
id1 numeric;
qno numeric;
id2 varchar(10);
timer timestamp;
hours1 numeric;
begin
Select cast(sysdate as timestamp) into timer from dual;
time1:=null;
  count1:=0;
     OPEN cameras;
     LOOP
        FETCH cameras INTO abc;
        EXIT WHEN cameras%NOTFOUND;
        begin
select c.checkouttime into time1 from cameracheckout c where
c.reservationid=abc.reservationid;
EXCEPTION
      WHEN NO_DATA_FOUND THEN
        time1:= NULL;
        end;
select
    extract( day from diff ),
    extract( hour from diff ),
    extract( minute from diff ) into dayt,hours,mins
from (
        select (CAST(sysdate as timestamp) - CAST(abc.starttime as timestamp))
diff
        from dual
     );
 SELECT TO_CHAR
    (SYSDATE, 'HH24') into hours1
     FROM DUAL;

if time1 is NULL then
      l_total:=hours+CEIL(mins/60);

        if (dayt>=0 and hours>=1) then
        insert into reminder
values('1',abc.studentnumber,abc.facultynumber,'Camera',abc.cameraid ||' Reservation
Cancelled',timer);
```

```
        select starttime,endtime,studentnumber,facultynumber,queueno into
time2,time3,student,faculty,qno from cameraqueue where rownum=1 and
cameraid=abc.cameraid order by queueno asc ;
      select max(reservationid)+1 into id1 from camerareservation;
       insert into reminder
values('1',abc.studentnumber,abc.facultynumber,'Camera',abc.cameraid ||' Camera
Ready for Pickup', timer);
      delete from cameraqueue where queueno=qno  and cameraid=abc.cameraid
and studentnumber=student or facultynumber=faculty;
         id2:=abc.cameraid;
           SYS.DBMS_OUTPUT.PUT_LINE(time3||time3|| 'hell');
       delete from camerareservation where  reservationid=abc.reservationid and
facultynumber=abc.facultynumber or studentnumber=abc.studentnumber;
         insert into camerareservation values (id1,id2,time2,time3,student,faculty);
          end if;

end if;


    END LOOP;
   CLOSE Cameras;
         end;
```

Explanation: On Friday 10:00 AM if the requested camera is not checked out, then it sends a reminder to the patron, who was supposed to check it out, about the cancellation. Then a notification is sent to the patron who is next in the CameraQueue table about the Camera's availability and that needs to be picked up.

2. CameraConfirmation

```
Procedure CameraConfirmation
IS
CURSOR cam
     IS
   select * from cameraqueue order by queueno;
abc cameraqueue%ROWTYPE;
a1 numeric;
b1 numeric;
c1 numeric;
dayt numeric;
  hours numeric;
  mins numeric;
timer timestamp;
begin
Select cast(sysdate as timestamp) into timer from dual;
    OPEN cam;
    LOOP
      FETCH cam INTO abc;
```

```
        EXIT WHEN cam%NOTFOUND;
              select
    extract( day from diff ),
    extract( hour from diff ),
    extract( minute from diff ) into dayt,hours,mins
from (
        select (CAST(SYSDATE as timestamp) - CAST(abc.starttime as timestamp))
diff
        from dual
     );

SELECT TO_CHAR
    (SYSDATE, 'HH24') into hours
     FROM DUAL;
          if ((dayt>=0 and hours>=-1) OR (dayt>0))then
        insert into reminder
values('1',abc.studentnumber,abc.facultynumber,'Camera',abc.cameraid ||' Camera is
not available till 8 am',timer);
      delete from cameraqueue where queueno=abc.queueno and
cameraid=abc.cameraid;
        end if;


END LOOP;
   CLOSE cam;
end;
```

Explanation: On Friday 8:00 AM, if the CameraRequest is not fulfilled a reminder is sent to the patrons in the CameraQueue table that by 8:00 AM the request hasnot been cleared.

3. PublicationPriority

```
Procedure PublicationPriority
IS
CURSOR publications
     IS
    select * from checkoutwait order by facultynumber,timestamp asc;
abc checkoutwait%ROWTYPE;
count1 number;
timediff number;
begin
  count1:=0;
     OPEN publications;
     LOOP
        FETCH publications INTO abc;
        EXIT WHEN publications%NOTFOUND;

IF abc.typepub='Book' then
SYS.DBMS_OUTPUT.PUT_LINE('hello');
```

```
select availablecopies into count1 from book where resourceid=abc.resourceid;
select ceil(CAST (abc.reservedupto AS DATE)- CAST (sysdate AS DATE)) into
timediff from dual;
if count1>0 then
SYS.DBMS_OUTPUT.PUT_LINE(count1||timediff);
        if timediff>0 then
        SYS.DBMS_OUTPUT.PUT_LINE(count1||timediff);
        insert into checkout values
(abc.resourceid,abc.timestamp,abc.reservedupto,null,abc.studentnumber,abc.facultynu
mber,abc.typepub,abc.namepub);
        update book set availablecopies=availablecopies-1 where
resourceid=abc.resourceid;
        end if;
        delete from checkoutwait where resourceid=abc.resourceid and
facultynumber=abc.facultynumber or studentnumber=abc.studentnumber;
end if;
if timediff<=0 then
delete from checkoutwait where resourceid=abc.resourceid and
facultynumber=abc.facultynumber or studentnumber=abc.studentnumber;
end if;
elsif abc.typepub='Journal' then
select availablecopies into count1 from journal where resourceid=abc.resourceid;
select ceil(CAST (abc.reservedupto AS DATE)- CAST (sysdate AS DATE)) into
timediff from dual;
if count1>0 then
SYS.DBMS_OUTPUT.PUT_LINE(count1||timediff);
        if timediff>0 then
        SYS.DBMS_OUTPUT.PUT_LINE(count1||timediff);
        insert into checkout values
(abc.resourceid,abc.timestamp,abc.reservedupto,null,abc.studentnumber,abc.facultynu
mber,abc.typepub,abc.namepub);
        update journal set availablecopies=availablecopies-1 where
resourceid=abc.resourceid;
        end if;
        delete from checkoutwait where resourceid=abc.resourceid and
facultynumber=abc.facultynumber or studentnumber=abc.studentnumber;
end if;
if timediff<=0 then
delete from checkoutwait where resourceid=abc.resourceid and
facultynumber=abc.facultynumber or studentnumber=abc.studentnumber;
end if;
SYS.DBMS_OUTPUT.PUT_LINE('hello1');
elsif abc.typepub='Conference' then
select availablecopies into count1 from conference where resourceid=abc.resourceid;
select ceil(CAST (abc.reservedupto AS DATE)- CAST (sysdate AS DATE)) into
timediff from dual;
if count1>0 then
SYS.DBMS_OUTPUT.PUT_LINE(count1||timediff);
        if timediff>0 then
        SYS.DBMS_OUTPUT.PUT_LINE(count1||timediff);
        insert into checkout values
(abc.resourceid,abc.timestamp,abc.reservedupto,null,abc.studentnumber,abc.facultynu
mber,abc.typepub,abc.namepub);
```

```
        update conference set availablecopies=availablecopies-1 where
resourceid=abc.resourceid;
        end if;
        delete from checkoutwait where resourceid=abc.resourceid and
facultynumber=abc.facultynumber or studentnumber=abc.studentnumber;
end if;
if timediff<=0 then
delete from checkoutwait where resourceid=abc.resourceid and
facultynumber=abc.facultynumber or studentnumber=abc.studentnumber;
end if;
SYS.DBMS_OUTPUT.PUT_LINE('hello2');
end if;

    END LOOP;
   CLOSE publications;
end;
```

Explanation: In the CheckoutWait table the priority to get the publication is checked with every new request. The priority is as suggested, if the patron who requests the publication is a faculty then irrespective of what the timestamp is, the highest priority will be given to that patron. Next for all the students, the priority is given on the basis of the timestamp of request which works on the first come first serve basis.

4. UpdateReminderPublications

```
Procedure UpdateReminderPublications
  ( student IN varchar,faculty in varchar )
IS
     l_total       INTEGER := 10000;
  dayt numeric;
  hours numeric;
  mins numeric;
  timer timestamp;
     CURSOR checkoutinfostudent
     IS
          SELECT * FROM checkout where studentnumber=student;
  CURSOR checkoutinfofaculty
     IS
          SELECT * FROM checkout where facultynumber=faculty;
    abc checkout%ROWTYPE;
  BEGIN
  delete from reminder where remindermessage like'B%';
  Select cast(sysdate as timestamp) into timer from dual;
   SYS.DBMS_OUTPUT.PUT_LINE('hello');
  if faculty is NULL then
    OPEN checkoutinfostudent;
    LOOP
      FETCH checkoutinfostudent INTO abc;
      EXIT WHEN checkoutinfostudent%NOTFOUND;
```

```
        select
    extract( day from diff ),
    extract( hour from diff ),
    extract( minute from diff ) into dayt,hours,mins
from (
        select (CAST(abc.reservedupto as timestamp) - CAST(SYSDATE as
timestamp)) diff
        from dual
     );
     l_total:=dayt+CEIL(hours/24);
     SYS.DBMS_OUTPUT.PUT_LINE(l_total);

     if l_total>=90 then
     insert into reminder values('1',student,null,abc.typepub,abc.namepub || ' 1rd late
fees Reminder',timer);
      insert into reminder values('2',student,null,abc.typepub,abc.namepub || ' 2rd late
fees Reminder',timer);
     insert into reminder values('3',student,null,abc.typepub,abc.namepub || ' 3rd late
fees Reminder',timer);
   elsif l_total>=60 and l_total<90 then
   insert into reminder values('1',student,null,abc.typepub,abc.namepub || ' 1rd late
fees Reminder',timer);
 insert into reminder values('2',student,null,abc.typepub,abc.namepub || ' 2rd late
fees Reminder',timer);
    elsif l_total>=30 and l_total<60 then
     insert into reminder values('1',student,null,abc.typepub,abc.namepub || ' 1rd late
fees Reminder',timer);
      end if;

select
    extract( day from diff ) into dayt
from (
        select (CAST(abc.reservedupto as timestamp) - CAST(abc.checkouttime as
timestamp)) diff
        from dual
     );
     l_total:=dayt;

if dayt!=0 then
    select
    extract( day from diff ),
    extract( hour from diff ),
    extract( minute from diff ) into dayt,hours,mins
from (
        select (CAST(SYSDATE as timestamp) - CAST(abc.reservedupto as
timestamp)) diff
        from dual
     );
     l_total:=dayt*-1;
if l_total=0 then
     insert into reminder values('1',student, null,abc.typepub,abc.namepub || ' 1st
Return Date Reminder',timer);
```

```
     insert into reminder values('2',student, null,abc.typepub,abc.namepub || ' 2nd
Return Date Reminder',timer);
   elsif l_total<=2 and l_total>0 then
   insert into reminder values('1',student, null,abc.typepub,abc.namepub || ' 1st
Return Date Reminder',timer);
     end if;
end if;

     END LOOP;
    CLOSE checkoutinfostudent;
    end if;

    if student is NULL then
     OPEN checkoutinfofaculty;
     LOOP
        FETCH checkoutinfofaculty INTO abc;
        EXIT WHEN checkoutinfofaculty%NOTFOUND;
        select
    extract( day from diff ),
    extract( hour from diff ),
    extract( minute from diff ) into dayt,hours,mins
from (
       select (CAST(abc.reservedupto as timestamp) - CAST(SYSDATE as
timestamp)) diff
       from dual
     );
     l_total:=dayt+CEIL(hours/24);
     SYS.DBMS_OUTPUT.PUT_LINE(l_total);

     if l_total>=90 then
     insert into reminder values('1',null,faculty,abc.typepub,abc.namepub || ' 1st late
fees Reminder',timer);
      insert into reminder values('2',null,faculty,abc.typepub,abc.namepub || ' 2nd late
fees Reminder',timer);
     insert into reminder values('3',null,faculty,abc.typepub,abc.namepub || ' 3rd late
fees Reminder',timer);
   elsif l_total>=60 and l_total<90 then
   insert into reminder values('1',null,faculty,abc.typepub,abc.namepub || ' 1 late fees
Reminder',timer);
 insert into reminder values('2',null,faculty,abc.typepub,abc.namepub || ' 2late fees
Reminder',timer);
    elsif l_total>=30 and l_total<60 then
     insert into reminder values('1',null,faculty,abc.typepub,abc.namepub || ' 1 late
fees Reminder',timer);
      end if;

 select
    extract( day from diff ) into dayt
from (
       select (CAST(abc.reservedupto as timestamp) - CAST(abc.checkouttime as
timestamp)) diff
       from dual
     );
```

```
    l_total:=dayt;

if dayt!=0 then
   select
   extract( day from diff ),
   extract( hour from diff ),
   extract( minute from diff ) into dayt,hours,mins
from (
       select (CAST(SYSDATE as timestamp) - CAST(abc.reservedupto as
timestamp)) diff
       from dual
     );
     l_total:=dayt*-1;
if l_total=0 then
     insert into reminder values('1',null,faculty,abc.typepub,abc.namepub || ' 1st
Return Date Reminder',timer);
      insert into reminder values('2',null,faculty,abc.typepub,abc.namepub || ' 2nd
Return Date Reminder',timer);
   elsif l_total<=2 and l_total>0 then
   insert into reminder values('1',null,faculty,abc.typepub,abc.namepub || ' 1st Return
Date Reminder',timer);
      end if;
    end if;


     END LOOP;
    CLOSE checkoutinfofaculty;
    end if;
  END;
```

Explanation: Here reminders are sent as per the situation. For instance, for the late return fees, a reminder is sent after 30, 60, 90 days before the student account gets held. The other kind of reminder is to update the patron about the due date which is sent 3 days before the due date of the book and one more reminder is sent 1 day before the due date in the form of the notification into their account.

5. UpdateRoomReservations

```
Procedure UpdateRoomReservations
IS
     l_total       INTEGER := 10000;
  hours numeric;
  dayt numeric;
     CURSOR checkoutinfostudent
     IS
   select * from TEMP1;
   abc TEMP1%ROWTYPE;
   abc1 roomreservation%ROWTYPE;
     BEGIN
```

```
    OPEN checkoutinfostudent;
    LOOP
      FETCH checkoutinfostudent INTO abc;
      EXIT WHEN checkoutinfostudent%NOTFOUND;
      select
   extract( hour from diff ),extract(day from diff) into hours,dayt
from (
      select (CAST(SYSDATE as timestamp) - CAST(abc.starttime as timestamp))
diff
      from dual
    );
   SYS.DBMS_OUTPUT.PUT_LINE(hours);
   select * into abc1 from roomreservation where reservationid=abc.reservationid;

    if abc.checkouttime IS NULL and ((hours>0 and dayt=0) or dayt>0) then
    insert into roomreservationnulllog values
(abc1.roomid,abc1.starttime,abc1.endtime, abc1.studentnumber, abc1.facultynumber);
    delete from roomreservation where reservationid=abc.reservationid;
    elsif abc.checkouttime IS NOT NULL and ((hours>2 and dayt=0) or dayt>0) then
    update checkoutroom set checkintime=CAST(SYSDATE as timestamp) where
reservationid=abc.reservationid;
     insert into roomreservationlog values (abc1.roomid,abc1.starttime,abc1.endtime,
abc1.studentnumber, abc1.facultynumber);
    delete from roomreservation where reservationid=abc.reservationid;
    elsif ((hours>2 and dayt=0) or dayt>0) then
    insert into roomreservationnulllog values
(abc1.roomid,abc1.starttime,abc1.endtime, abc1.studentnumber, abc1.facultynumber);
    delete from roomreservation where reservationid=abc.reservationid;
    end if;

    END LOOP;
   CLOSE checkoutinfostudent;

 END;
```

Explanation : Automatically the room is checked in into the database and becomes available once the time is over and updates the RoomReservationLog table.

6. UpdateRoomReservation1

```
Procedure UpdateRoomReservations1
IS
     l_total       INTEGER := 10000;
 hours numeric;
 dayt numeric;
     CURSOR checkoutinfostudent1
     IS
   select * from roomreservation;
   abc roomreservation%ROWTYPE;
   abc1 timestamp;
```

```
 BEGIN

 OPEN checkoutinfostudent1;
 LOOP
   FETCH checkoutinfostudent1 INTO abc;
   EXIT WHEN checkoutinfostudent1%NOTFOUND;
   select
extract( hour from diff ),
extract( day from diff ) into hours, dayt
from (
   select (CAST(SYSDATE as timestamp) - CAST(abc.starttime as timestamp))
diff
   from dual
 );
 begin
 select checkouttime into abc1 from checkoutroom where
RESERVATIONID=abc.reservationid;
EXCEPTION
  WHEN NO_DATA_FOUND THEN
   abc1:= NULL;
   end;

 if abc1 IS NULL and ((hours>0 and dayt=0) or dayt>0) then
 insert into roomreservationnulllog values (abc.roomid,abc.starttime,abc.endtime,
abc.studentnumber, abc.facultynumber);
 delete from roomreservation where reservationid=abc.reservationid;
  end if;

 END LOOP;
CLOSE checkoutinfostudent1;

            END;
```

Explanation: If the room is not checked in by the patron post one hour according to the schedule, the reservation gets cancelled and an entry is made into the RoomReservationNullLog.

Constraints handled by the application code:

Which publication can be checked out for what interval of time is handled in the application code.