Problem 1

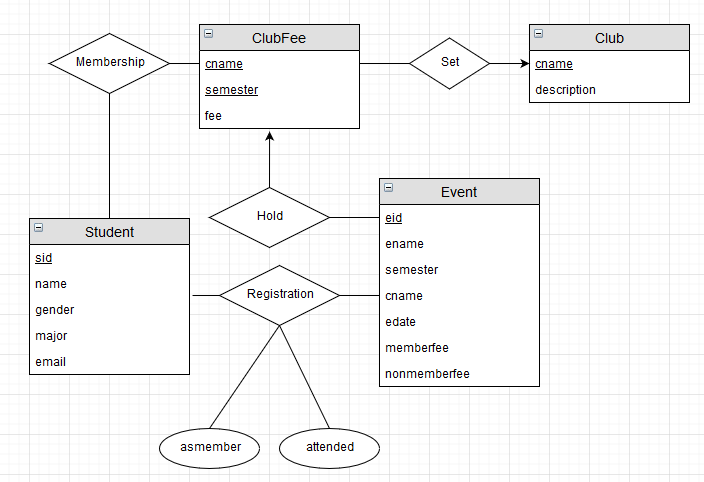
1. ClubFee: cname references Club(cname)

Membership: cname, semester references ClubFee(cname, semester)

Event: cname, semester references ClubFee(cname, semester)

Registration: sid references Student(sid), eid references Event(eid)





2. SELECT DISTINCT Membership.cname FROM Membership WHERE Membership.cname NOT IN (SELECT Membership.cname FROM Membership, Student WHERE Membership.sid = Student.sid AND Student.gender = 'female');
3. SELECT Student.`name` FROM Student WHERE EXISTS (SELECT \* FROM Registration,Event WHERE Registration.sid = Student.sid AND Registration.asmember = 0 AND Registration.attended = 1 AND Registration.eid = Event.eid AND Event.cname = 'Chess Club' AND Event.semester='Spring 2018');
4. SELECT Registration.sid FROM Event,Registration WHERE Event.eid=Registration.eid AND Event.cname='Chess Club' AND Event.semester='Spring 2018' AND Registration.attended=1 AND Registration.asmember=0 GROUP BY Registration.sid HAVING SUM(Event.nonmemberfee)>=(SELECT Clubfee.fee FROM Clubfee WHERE Clubfee.cname='Chess Club' AND Clubfee.semester='Spring 2018');
6. ∏ cname (Membership) – ∏ cname (σ Membership.sid = Student.sid ^ Student.gender = 'female' (Membership ⅹ Student) )
7. ∏ name (Student ÷ ∏ sid ( sid  count(\*)>=1 (σ Registration.asmember = false ^ Registration.attended = true ^ Registration.eid = Event.eid ^ Event.cname = 'Chess Club' ^ Event.semester='Spring 2018' (Registration ⅹ Event)) ))
8. f ← ∏ fee (σ cname='Chess Club' ^ semester='Spring 2018' (ClubFee))

∏ Registration.sid (Registration.sid  SUM(Event.nonmemberfee)>=f (σ Event.eid=Registration.eid ^ Event.cname='Chess Club' ^ Event.semester='Spring 2018' ^ Registration.attended=1 ^ Registration.asmember=0 (Event ⅹ Registration)))

Problem 2

1. List the screen names of all users who lives in Hoboken and sent some tweets on July 6, 2017.

List the uids and screen names of all users who follows the user with the screen name ‘Joe Schmoe’ and sent some tweets on November 11, 2015.

1. ∏ U.screenname (σ U.uid = T.uid ^ U.ucity = Hoboken ^ T.tdate = July 6, 2017 (ρ U (User) ⅹ ρ T (Tweet)))

∏ U.uid, U.screenname (σ U.uid = T.uid ^ U.uid = F.fuid ^ U2.uid = F.uid ^ T.tdate = November 11, 2015 ^ U2.screenname = Joe Schmoe (ρ U (User) ⅹ ρ F (Follows) ⅹ ρ T (Tweet) ⅹ ρ U2 (User)))

1. time=b\*tT+tS

=numberall \* sizeone / vT + latency

=5\*10^8\*5\*10^10\*100B/ 100MB/s + 10ms

=2.5 \* 10^13s + 0.01s

=2.5 \* 10^13s

1. tT=size node / v T = 4KB / 100MB/s = 4\*10^(-5)s

tS=10ms=0.1s

time=hi\*(tT+tS)+b\*tT

=4\*0.10004s+5\*10^7\*100B/4KB\*4\*10^(-5)s

=0.40016s+50s

=50.40016s

1. tT=size node / v T = 4KB / 100MB/s = 4\*10^(-5)s

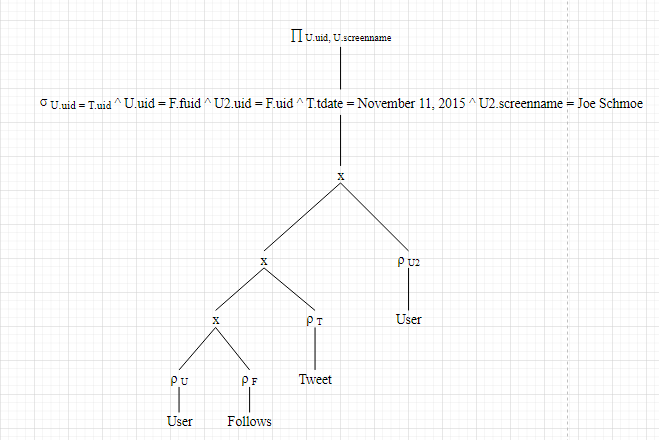
tS=10ms=0.1s

time= (hi + n) ∗ (tT + tS)

=(4+5\*10^7\*100B/4KB)\*(4\*10^(-5)s+0.1s)

=125050.4s.





1. time=b\*tT+tS

=numberall \* sizeone / vT + latency

=5\*10^8\*5\*10\*10\*5\*10^9\*5\*10^8\*100B/100MB/s + 10\*10^(-3)s

=6.25\*10^31s + 10\*10^(-3)s

=6.25\*10^31s

1. Create a sparse clustered B+-tree index on uid in Follows. By this way, we can reduce the seek times as much as possible.

Problem 3

1. FALSE

TRUE

FALSE

FALSE

TRUE

1. FALSE

FALSE

TRUE

FALSE

TRUE

1. FALSE

TRUE

TRUE

FALSE

TRUE

1. TRUE

FALSE

TRUE

TRUE

TRUE