University of Toronto
CSC343, Winter 2015

# Assignment 1


# Chuanrui Li(1000010846) Zhenyuan Bo(999232762)


## Part 1: Queries

1. Report the user name of every student who has never worked with anyone, but has indeed submitted at least one file for at least one assignment. **Answer:**

   $Studenttogether(userName) := \Pi_{M1.userName}\sigma_{M1.gID=M2.gID \wedge M1.userName1!=M1.userName2}$
   $(\rho_{M1}(Membership) \times \rho_{M2}(Membership))$
   $Studentalone(userName) := \Pi_{userName}(Membership) - \Pi_{userName}(Studenttogether)$
   $Result(userName) := \Pi_{userName}(Studentalone \bowtie Submission)$

2. Find the graders who have marked every assignment. We will say that a grader has marked an assignment if they have given a grade on that assignment to at least one group, whether or not that grade has been released. Report the grader's userName. **Answer:**

   $Shouldhave(aID, userName) := \Pi_{aID}(Assignment) \times \Pi_{userName}(Grader)$
   $Actualhave(aID, userName) := \Pi_{aID,userName}(Grader \bowtie Group \bowtie Result)$
   $Missing(aID, userName) := Shouldhave - Actualhave$
   $Result(userName) := \Pi_{userName}(Grader) - \Pi_{userName}(Missing)$

3. Find all groups for A2 (*i.e.*, the assignment whose description is "A2") whose last submission was after the due date, but who submitted at least two different files (*i.e.*, files with two different names) before the due date. Report the group ID, the name of the first file they submitted, and when they submitted it. If there are ties for a group's first submit, report them all. **Answer:**

   $Lastsub(aID, gID, due, fileName, userName, when) := \Pi_{aID,gID,due,fileName,userName,when}$
   $\sigma_{when>due \wedge description="A2"}(Assignment \bowtie Submission \bowtie Group)$

   $Beforedue(aID, gID, when, fileName) := \Pi_{aID,gID,when,fileName}$
   $\sigma_{sub1.fileName!=sub2.fileName \wedge sub1.due>when \wedge sub2.due>when \wedge sub1.gID=sub2.gID}$
   $(\rho_{sub1}(Lastsub) \times \rho_{sub2}(Lastsub))$

   $Notfirst(aID, gID, when, fileName) := \Pi_{s2.aID,s2.gID,s2.when,s2.fileName}\sigma_{s1.when<s2.when \wedge s1.gID=s2.gID}$
   $(\rho_{s1}(Beforefue) \times \rho_{s2}(Beforedue))$

   $Result(gID, fileName, when) := \Pi_{gID,filename,when}(Beforedue) - \Pi_{gID,filename,when}(Notfirst)$

4. Find pairs of students who worked in a group together, and without any other students, on each assignment in the database that allowed groups of size two or more. Report their user names, last names, and firstnames. **Answer:**

$Leasttwo(r1, r2, gID) := \Pi_{M1.userName,M2.userName}\sigma_{M1.userName>M2.userName \wedge Mem1.gID=Mem2.gID}$
$(\rho_{M1}(Membership) \times \rho_{M2}(Membership))$

$Leastthree(r3, r4, gID) := \Pi_{M1.userName,M2.userName}$
$\sigma_{M1.userName!=M2.userName \wedge M2.userName!=M3.userName \wedge M1.userName!=M3.userName \wedge M1.gID=M2.gID=M3.gID}$
$Leastthree(r3, r4, gID) := \sigma_{M1.r3>M2.r4}(\rho_{M1}(Leastthree) \times \rho_{M2}(Leastthree))$

$Exacttwo(r1, r2, gID) := Leasttwo - Leastthree$
$Checkmax(aID, groupMax, gID) := \Pi_{aID,groupMax}Assignment \bowtie \Pi_{gID,aID}Group$
$Actualhave(r1, r2, aID) := \Pi_{r1,r2,aID}\sigma_{groupMax=>2}(Exacttwo \bowtie Checkmax)$
$Shouldhave(r1, r2, aID) := \Pi_{r1,r2,aID}(\Pi_{r1,r2}Exacttwo \times \Pi_{aID}(\sigma_{groupMax=>2}Checkmax))$
$Final(r1, r2) := \Pi_{r1,r2}(Shouldhave - Actualhave)$
$User1(r1, l1, f1) := \Pi_{userName,lastName,fisrtName}(User)$
$User2(r2, l2, f2) := \Pi_{userName,lastName,fisrtName}(User)$
$Final(r1, l1, f1, r2) := Final \bowtie User1$
$Result(r1, l1, f1, r2, l2, f2) := Final \bowtie User2$

5. Find any assignments where the highest mark given by one grader is less than the lowest mark given by another grader. In your result, include a row for each grader on each of these assignments. Report the assignment ID, the grader's userName, and their minimum and maximum grade. **Answer:**

$Marks(userName, mark, aID) := \Pi_{userName,mark,aID}(Group \bowtie Result \bowtie Grader)$
$Nhighestmark(userName, mark, aID) := \Pi_{M1.username,M1.mark,M1.aID}$
$\sigma_{M1.userName=M2.userName \wedge M1.mark<M2.mark}(\rho_{M1}(Marks) \times \rho_{M2}(Marks))$
$Highestmark(userName, mark, aID) := Marks - Nhighestmark$
$Nlowestmark(userName, mark, aID) := \Pi_{M1.userName,M1.mark,M1.aID}$
$\sigma_{M1.userName=M2.userName \wedge M1.mark>M2.mark}(\rho_{M1}(Marks) \times \rho_{M2}(Marks))$
$Lowestmark(userName, mark1, aID1) := Marks - Nlowestmark$

— Nature join for highest and lowest marks by username, then got the both highest and lowest marks for each individual

$Newmarks(userName, high, aID, low, aID1) := (Highestmark \bowtie Lowestmark)$

— Got the result but different format: aID, userName1, high1, low1, userName2, high2, low2

$Result(aID, userName1, high1, low1, userName2, high2, low2) :=$

$\Pi_{M1.userName,M1.high,M1.aID,M2.userName,M2.low,M2.high}$

$\sigma_{M1.userName>M2.userName \wedge M1.low>M2.high \wedge M1.aID1=M2.aID}(\rho_{M1}(Newmarks) \times \rho_{M2}(Newmarks))$

— Got the result with right format: aID, userName, high, low

$Final(aID, userName, high, low) := \Pi_{aID,userName1,high1,low1}(Result) \cup$

$\Pi_{aID,userName2,high2,low2}(Result)$

6. Find all students who have worked in a group with at least one other person, but have never worked with the same person twice. Report their userName. **Answer:**

$Leastone(name) := \Pi_{M1.userName}$

$\sigma_{M1.userName!=M2.userName \wedge M1.gID=M2.gID}(\rho_{M1}(Membership) \times \rho_{M2}(Membership))$

— Got sequence: M1.userName greater than M2.userName. EX: B, A

$Leasttwice2(name1, gID1, name2, gID2) := \Pi_{M1.userName,M1.gID,M2.userName,M1.gID}$

$\sigma_{M1.userName>M2.userName \wedge M1.gID=M2.gID}(\rho_{M1}(Membership) \times \rho_{M2}(Membership))$

— Got sequence: M1.userName less than M2.userName. EX: A, B

$Leasttwice3(name1, gID1, name2, gID2) := \Pi_{M1.userName,M1.gID,M2.userName,M1.gID}$

$\sigma_{M1.userName<M2.userName \wedge M1.gID=M2.gID}(\rho_{M1}(Membership) \times \rho_{M2}(Membership))$

— Leasttwice2 X Leasttwice3, then we only need to project first column to get all userName, EX:B, A, A, B

$Sametwice(userName) := \Pi_{M1.name1}$

$\sigma_{M1.name1=M2.name2 \wedge M1.name2=M2.name1 \wedge M1.gID1!=M2.gID1}(\rho_{M1}(Leasttwice2) \times \rho_{M2}(Leasttwice3))$

$Nsametwice(name) := \Pi_{userName}(Membership) - Sametwice$

$Result(name) = Leastone \cap Nsametwice$

7. Find all students who meet these two requirements: (a) their groups (whether they were working alone or with others) handed in every required file, and did so on time, for all assignments, and (b) their grades never went down from one assignment to another with a later due date. Report their userName. Note: If an assignment has no required files, it is true of any group that they handed in every required file. **Answer:**

$Shouldhave(gID, aID, fileName, userName) := \Pi_{gID,aID,fileName,userName}(Required \bowtie Group)$

$Actualhave(gID, aID, fileName, userName) := \Pi_{gID,aID,fileName,userName}((Required) \bowtie (Group) \bowtie \Pi_{gID,fileName,userName}(Submission))$

$Missing(gID, aID, fileName, userName) := Shouldhave - Actualhave$

$Result1(userName) := \Pi_{userName}(User) - \Pi_{userName}(Missing)$

$Checkdue(userName) := \Pi_{userName}\sigma_{when<due}$

$(Result1 \bowtie \Pi_{aID,due}Assignment \bowtie \Pi_{gID,userName,when}Submission \bowtie \Pi_{gID,aID}Group)$

$Student(userName, Mark, due, aID) := \Pi_{userName,Mark,due,aID}$

$(Membership \bowtie Result1 \bowtie \Pi_{gID,Mark}(Result) \bowtie \Pi_{gID,aID}(Group) \bowtie \Pi_{aID,due}(Assignment))$

$drop(userName) := \Pi_{userName}\sigma_{T1.userName=T2.userName \wedge T1.Mark<T2.Mark \wedge T1.due>T2.due}(\rho_{T1}(Student) \times \rho_{T2}(Student))$

$Result(userName) := \Pi_{userName}Student - drop$

8. Find all assignments that have one or more groups with no grade or with a grade that has not been released. Report the assignment ID and description. **Answer:**

   $Nograde(gID) := \Pi_{gID}(Group) - \Pi_{gID}(Result)$

   $Withgrade(gID) := \Pi_{gID}\sigma_{released=false}(Result \bowtie Group)$

   $Result(aID, description) := \Pi_{aID,description}((Nograde \cup Withgrade) \bowtie Group \bowtie \Pi_{aID,description}(Assignment))$

9. Assignments may have required files, but students can also hand in other files that are not required. Find all groups that never handed in a file that was not required. Report the group ID. **Answer:**

   $Actualfile(gID, aID, filename) := \Pi_{gID,aID}(Group) \bowtie \Pi_{gID,fileName}(Submission)$

   $Nrequired(gID) := \Pi_{gID}\sigma_{Actualfile.aID=Requried.aID \wedge Actualfile.fileName!=Requried.fileName}(Actualfile \times Requried)$

   $Result(gID) := \Pi_{gID}(Group) - Nrequired$

## Part 2: Additional Integrity Constraints

Express the following integrity constraints with the notation $R = \emptyset$, where $R$ is an expression of relational algebra. You are welcome to define intermediate results with assignment and then use them in an integrity constraint.

1. No grades can be released for an assignment unless every group has been given a grade on that assignment (whether or not it has been released). **Answer:**

   $Actualhave(gID, aID) := \Pi_{gID,aID}(Group) \bowtie (Result)$

   $Shouldhave(gID, aID) := \Pi_{gID,aID}(Group)$

$Missing(aID) := \Pi_{aID}(Shouldhave - Actualhave)$

$\sigma_{released=true}(Missing \bowtie Group \bowtie Result) = \emptyset$

2. A TA can't give a grade to any groups on an assignment unless they have completed marking (*i.e.,* they have given a grade, whether or not it has been released) for every group they were assigned to grade on every assignment with an earlier due date. **Answer:**

$Marked(aID, userName) := \Pi_{aID,userName}(Grader \bowtie Group \bowtie Result)$

$Unmarked(aID, userName) := \Pi_{aID,userName}(Group \bowtie Grader) - \Pi_{aID,userName}(Marked)$

$\sigma_{Marked.due<Unmarked.due \wedge Marked.userName=Unmarked.userName \wedge Marked.type=Unmarked.type="TA"}$
$(\rho_{Marked}(Marked \bowtie \Pi_{aID,due}(Assignment) \bowtie \Pi_{userName,type}(User)) \times \rho_{Unmarked}(Unmarked \bowtie \Pi_{aID,due}(Assignment) \bowtie \Pi_{userName,type}(User))) = \emptyset$

3. A TA can't be assigned to grade a group of size two or more unless he or she has already given a grade (that has been released) to at least 3 students (each working in a group of size 1) on an assignment with an earlier due date. **Answer:**

$Assign(aID, gID, name) := \Pi_{aID,gID,userName}(Membership \bowtie Group)$

$Leasttwo(aID, gID, userName) := \Pi_{r1.aID,r1.gID,r1.userName} \sigma_{r1.gID=r2.gID \wedge r1.name!=r2.name}$
$(\rho_{r1}(Assign) \times \rho_{r2}(Assign))$

$Exactone(aID, gID, user) := Assign - Leasttwo$

$Required(aID, gID, userName) := \Pi_{r1.aID,r1.gID,r1.user}$
$\sigma_{r1.gID!=r2.gID \wedge r2.gID!=r3.gID \wedge r1.gID!=r3.gID \wedge r1.user!=r2.user \wedge r2.user!=r3.user \wedge r1.user!=r3.user \wedge r1.aID=r2.aID=r3.aID}$
$(\rho_{r1}(Exactone) \times \rho_{r2}(Exactone) \times \rho_{r2}(Exactone))$

$TA1(gID, userName) := \Pi_{gID,userName} \sigma_{type="TA" \wedge released=True}(Required \bowtie \Pi_{gID,released}(Result) \bowtie \Pi_{userName,type}(User))$

$Required2(gID, userName) := \Pi_{M1.gID,M1.userName} \sigma_{M1.gID=M2.gID \wedge M1.userName>M2.userName}$
$(\rho_{M1}(Membership) \times \rho_{M2}(Membership))$

$TA2(gID, userName) := \Pi_{gID,userName} \sigma_{type="TA"}((Required2) \bowtie (User))$

$Allowed(gID, userName) := \Pi_{gID,userName} \sigma_{TA1.due<TA2.due}(TA1 \bowtie group \bowtie Assignment \cap TA2 \bowtie group \bowtie Assignment)$

$Ungraded(gID, userName) := TA2 \bowtie (Grader) \bowtie (\Pi_{gID}(Group) - \Pi_{gID}(Result))$

— Ungraded is subset of assigned, assigned is a subset of Allowed. subset - set = empty
$Ungraded - Allowed = \emptyset$