# Assignment 1: Sample Solutions

Note that there are multiple correct answers to all of these questions.

## Schema 1

1. Find all the users who have never liked or viewed a post or story of a user that they do *not* follow. Report their user id and “about” information. Put the information into a relation with attributes “username” and “description”.

   Solution:

   – all possible pairs of users

   $\mathit{AllUserPairs}(\mathit{user1},\mathit{user2}) := \Pi_{U1.uid,U2.uid}\ \sigma_{U1.uid>U2.uid}[\rho_{U1}\mathit{Users} \times \rho_{U2}\mathit{Users}]$

   – pairs where user1 does not follow user2

   $\mathit{NotFollowed}(\mathit{user1},\mathit{user2}) := \mathit{AllUserPairs} - \rho(\mathit{user}1,\mathit{user}2)(\Pi_{\mathit{Follower},\mathit{Followed}}\ \mathit{Follows})$

   – users who liked post of someone they don’t follow

   $\mathit{LikedStrangerPost}(\mathit{user}) := \Pi_{\mathit{user1}}\sigma_{\mathit{user2}=\mathit{uid}}\mathit{NotFollowed} \times (\mathit{Likes} \bowtie \mathit{Post})$

   – users who viewed story of someone they do not follow

   $V\ \mathit{iewedStrangerStory}(\mathit{user}) := \Pi_{\mathit{user1}}$

   $\sigma_{\mathit{NotFollowed.user1}=\mathit{Saw.viewerid} \wedge \mathit{NotFollowed.user2}=\mathit{Story.uid} \wedge \mathit{Saw.sid}=\mathit{Story.sid}}(\mathit{Saw} \times \mathit{Story} \times \mathit{NotFollowed})$

   – users who did any of the 2 things

   $\mathit{AnyStranger}(\mathit{uid}) := \mathit{LikedStrangerPost} \cup \mathit{ViewedStrangerStory}$

   – users who did NOT do either of these two things

   $\mathit{Candidates}(\mathit{uid}) := \Pi_{\mathit{uid}}\mathit{Users} - \mathit{AnyStranger}$

   – final (join and rename to get fields we need)

   $\mathit{Final}(\mathit{name},\mathit{description}) := \rho(\mathit{name},\mathit{description})(\Pi_{\mathit{name},\mathit{about}}(\mathit{Candidates} \bowtie \mathit{Users})$

2. Find every hashtag that has been mentioned in at least three post captions on every day of 2020. You may assume that there is at least one post on each day of a year.

   Solution:

   – find posts in 2020, keep day and pid

   $2020\mathit{posts}(\mathit{day},\mathit{pid}) := \Pi_{\mathit{when.day},\mathit{pid}}\ \sigma_{\mathit{when.year}=\text{“2020”}}\mathit{Post}$

   – find all days in 2020

   $2020\mathit{days}(\mathit{day}) := \Pi_{\mathit{when.day}}\ \sigma_{\mathit{when.year}=\text{“2020”}}\mathit{Post}$

   – find hashtags with their posts from 2020

$\mathit{HT}2020(\mathit{day,tag,pid}) := 2020\mathit{posts} \bowtie \mathit{Hashtag}$

– find those mentioned on 3 times on same day $\mathit{ThreeMentions}(\mathit{day,tag}) :=$

$$\Pi_{HT1.day,HT1.tag}\ \sigma_{(HT1.tag=HT2.tag=HT3.tag)\wedge(HT1.day=HT2.day=HT3.day)\wedge(HT1.pid>HT2.pid>HT3.pid)}\ [(\rho_{HT1}\mathit{HT}2020) \times (\rho_{HT2}\mathit{HT}2020) \times (\rho_{HT3}\mathit{HT}2020)]$$

– make table of all days in 2020 and all tags
$\mathit{AllDaysTags}(\mathit{day,tag}) := 2020\mathit{days} \times \Pi_{tag}\mathit{Hashtag}$

– find day,tag combos that did not have 3 mentions
$\mathit{NotMentionedThrice}(\mathit{day,tag}) := \mathit{AllDaysTags} - \mathit{ThreeMentions}$

– set difference on tags only to find final solution
$\mathit{Final}(\mathit{tag}) := \Pi_{tag}\mathit{AllDaysTags} - \Pi_{tag}\mathit{NotMentionedThrice}$

3. Let’s say that a pair of users are “reciprocal followers” if they follow each other. For each pair of reciprocal followers, find all of their “uncommon followers”: users who follow one of them but not the other. Report one row for each of the pair’s uncommon follower.
   In it, include the identifiers of the reciprocal followers, and the identifier, name and email of the uncommon follower.

   Solution:

   $\mathit{ReciprocalFollowers}(\mathit{user1,user2}) :=$

$$\Pi_{F1.follower,F2.follower\sigma F1.follower=F2.followed\wedge F1.followed=F2.follower\wedge F1.follower<F2.follower} [(\rho_{F1}\mathit{Follows}) \times (\rho_{F2}\mathit{Follows})]$$

   – find sets of followers of each of the two users
   $\mathit{FollowsUser}1(\mathit{user1,user2,follower}) :=$

$$\Pi_{RF.user1,RF.user2,F.follower\sigma RF.user1=F.followed} [(\rho_{RF}\mathit{ReciprocalFollowers}) \times (\rho_{F}\mathit{Follows})]$$

   $\mathit{FollowsUser}2(\mathit{user1,user2,follower}) :=$

$$\Pi_{RF.user1,RF.user2,F.follower}\sigma_{RF.user2=F.followed} [(\rho_{RF}\mathit{ReciprocalFollowers}) \times (\rho_{F}\mathit{Follows})]$$

   $\mathit{FollowsBoth}(\mathit{user1,user2,follower}) := \mathit{FollowsUser}1 \cap \mathit{FollowsUser}2$

   $\mathit{FollowsEither}(\mathit{user1,user2,follower}) := \mathit{FollowsUser}1 \cup \mathit{FollowsUser}2$

   $\mathit{FollowsOne}(\mathit{user1,user2,follower}) := \mathit{FollowsEither} - \mathit{FollowsBoth}$

   $\mathit{Solution}(\mathit{user1,user2,name,email}) := \Pi_{user1,user2,name,email\sigma follower=uid}[\mathit{FollowsOne} \times \mathit{User}]$

4. Find the user who has liked the most posts. Report the user’s id, name and email, and the id of the posts they have liked. If there is a tie, report them all.

Solution: Not possible with relational algebra alone.

5. Let's say a pair of users are "backscratchers" if they follow each other and like all of each others' posts.Report the user id of all users who follow some pair of backscratcher users.

Solution:

– find reciprocal follower pairs

$ReciprocalFollowers(user1,user2) :=$

$$\Pi_{F1.follower,F2.follower}\ \sigma_{F1.follower=F2.followed \wedge F1.followed=F2.follower \wedge F1.follower<F2.follower}\ [\rho_{F1}\, Follows \times \rho_{F2}\, Follows]$$

– find all posts by user1

$AllPostsU1(pid,user1,user2) := \Pi_{pid,user1,user2 \sigma uid=user1}[ReciprocalFollowers \times Post]$

– find posts by user1 liked by user2

$LikedU1Posts(pid,user1,user2) :=$

$$\Pi_{pid,user1,user2 \sigma (uid=user1) \wedge (liker=user2) \wedge (Post.pid=Likes.pid)}\ [ReciprocalFollowers \times Post \times Likes]$$

– posts by user1 not liked by user2

$NotLikedU1Posts(pid,user1,user2) := AllPostsU1 - LikedPosts$

– pairs where user1 doesn't like ALL of user2 posts

$NotReciprocalLikerU1(user1,user2) := \Pi_{user1,user2} NotLikedU1Posts$

– now do most of this again to find pairs where user2 doesn't like ALL of user1 posts

– find all posts by user2

$AllPostsU2(pid,user1,user2) := \Pi_{pid,user1,user2 \sigma uid=user2}[ReciprocalFollowers \times Post]$

– find posts by user2 liked by user1

$LikedU1Posts(pid,user1,user2) :=$

$$\Pi_{pid,user1,user2 \sigma (uid=user2) \wedge (liker=user1) \wedge (Post.pid=Likes.pid)}\ [ReciprocalFollowers \times Post \times Likes]$$

– posts by user2 not liked by user1

$NotLikedU2Posts(pid,user1,user2 := AllPostsU2 - LikedU2Posts$

– pairs where user2 doesn't like ALL of user1 posts

$NotReciprocalLikerU2(user1,user2) := \Pi_{user1,user2}\ NotLikedU2Posts$

– Backscratchers are all Reciprocal followers once re remove the NotReciprocalLikers in both

– directions

$Backscratchers(user1,user2) := ReciprocalFollowers - NotReciprocalLikerU1 - NotReciprocalLikerU2$

– Final (join Follows with Backscratchers)

$$Final(user) := \Pi_{F1.follower}\ \sigma_{(F1.Follower=F2.Follower)\wedge(F1.Followed=Backscratchers.user1)\wedge(F2.Followed=Backscratchers.user2)}$$
$$[\rho_{F1}Follows \times \rho_{F2}Follows \times Backscratchers]$$

## 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. Each user can have at most one current story.

   Solution:

   $$\sigma_{(S1.uid=S2.uid)\wedge(S1.sid<S2.sid)\wedge(S1.current=S2.current=True)}[\rho_{S1}Story \times \rho_{S2}Story] = \emptyset$$

2. Every post must include at least one picture or one video and so must every story.

   Solution:

   $$(\Pi_{pidPost} - \Pi_{pidPIncludes}) \cup (\rho_{pid}[\Pi_{sid}Story - \Pi_{sid}Sincludes]) = \emptyset$$

## Schema 2:

1. Find the clinic name and department name for all departments whose head is also head of another department.

We join relation Department with itself on attribute dHead and select tuples with different departmentIDs.

**Answer:**

$$\pi_{hName,dName}((\sigma_{departmentID \neq dID}(Department \bowtie$$
$$\rho_{dID\leftarrow departmentID,hID\leftarrow clinicID,dN\leftarrow dName}(Department))) \bowtie Clinic)$$

2. Find the first name and last name of all Physicians whose patients are never in the clinic for less than 10 days.

(They must have had patients admitted in the clinic at some time.)

**Answer:**

First we find physicians who had some patients in clinic for less than 10 days:

$PhywithSomePatients :=$

$$\pi_{prPhysician}\ (\sigma_{prPhysician=prInDate<10}\ (PatientRoomAssignment))$$

Then we subtract the above physicians from the set of all physicians to get the ones who never had a patient in clinic for less than 10 days:

$$PhywithnosuchPatients{:}\ = \pi_{prPhysician}\ (PatientRoomAssignment)\ - PhywithSomePatients$$

Now we join the above relation with Employee to get the names:

$$\pi_{eFirstName,eLastName}(\rho_{\ prPhysician \leftarrow\ employeeID}(\text{Employee})\ \bowtie\ PhywithnosuchPatients)$$

3. Find the first and last name of all patients who have exactly two physicians in the clinic's system.

(Note that physicians are specified in both appointments and room assignment.)

**Answer:**

First we create a temporary relation containing all the patients and their Physicians.
This information can be found in both $PatientRoomAssignment$ and $PatientAppointment$; relations:

$$PatientPhysician := \pi_{\ patientID,prPhysician}(PatientRoomAssignment) \cup \rho_{\ prPhysician \leftarrow employeeID}(\pi patientID, employeeID\ (\sigma_{dRole='Physician'}\ (Employee\ \ \bowtie PatientAppointment)))$$

We next find patients who have at least two Physicians:

$$PatientwithtwoPhy := \pi_{patientID}\ (\ \sigma_{prPhysician \neq pD}\ (PatientPhysician\ \ \bowtie\ \ \rho_{pD \leftarrow prPhysician}\ (PatientPhysician)))$$

Now we find patients with at least three Physicians:

$$PatientsatleastthreePhy := \pi_{\ patientID}(\ \sigma_{\ prPhysician \neq pD1 \wedge prPhysician \neq pD2 \wedge pD2 \neq pD1}\ (PatientPhysician\ \ \bowtie\ \ \rho_{pD1 \leftarrow prPhysician}\ (PatientPhysician)\ \ \bowtie\ \ \rho_{pD2 \leftarrow prDoctor}\ (PatientPhysician)))$$

Patients with *exactly* two physicians are those who are in $PatientwithtwoPhy$ but not in $PatientsatleastthreePhy$:

$$Final := \pi_{\ pFirstName,pLastName}\ (P\ \ \bowtie\ \ (PatientwithtwoPhy - \ PatientsatleastthreePhy))$$

4. Find the clinic and department name for the department at which there were no appointments from April 3, 2019 to April 8, 2019.

We first find departments which had some appointments between April 3, 2019 to April 8, 2019.

$$SomeAppointments\ : = \pi_{departmentID}\ (\ \sigma_{\ paDate\ >'2019-04-02' \wedge\ paDate\ <\ '2019-04-09'}(PA))$$

Departments with no appointment in that period can be found by subtracting $SomeAppointments$ from the set of all departments:

$$Final \coloneqq \pi_{hName,dName}(Clinic \bowtie Department \bowtie (\pi_{departmentID}(Department) - SomeAppointments))$$

5. Find the clinic, room number, and patient first and last names of all semi-private (size = 2) rooms which were occupied by female patients who were over 50 on May 13, 2019.

We first select rooms of size 2:
$RoomSize2 := \pi_{roomID}\ (\sigma_{rSize\ =\ 2}\ (\ Room\ ))$

Then we find rooms (and patients) in $RoomSize2$ in which there was a female patient on May 13, 2019 with over 50 years of age:

$$RoomFemalePatient50 \coloneqq \pi_{roomID,patientID} (\ \sigma_{prInDate\ <\ 2019-05-13\ \wedge\ prOutDate\ >\ 2019-05-13\ \wedge\ pSex='F'\wedge\ pDOB\ <\ 1969-08-13} (Patient \bowtie PatientRoomAssignment \bowtie RoomSize2))$$

Now we find names of the patients who occupied a room in $RoomFemalePatient50$:
$$Patients \coloneqq \pi_{hName,rNumber,pFirstName,pLastName} (H \bowtie R \bowtie P \bowtie RoomFemalePatient50))$$