Part 1

1

$Liked(liker, liked) :=$
$$\Pi_{Likes.liker, Post.pid}(Likes \bowtie Post)$$

$Viewed(viewer, viewed) :=$
$$\Pi_{Saw.viewerid, Story.sid}(Saw \bowtie Story)$$

“uid1” liked some post or viewed some story of “uid2” and “uid1” did not followed “uid2”
$ViewOrLikeNotFollow(uid1, uid2) :=$
$$\text{Liked} \cup \text{Viewed} - \Pi_{follower, followed}(Follows)$$

“uid” never liked some post or viewed some story of user whom he did not follow
$ViewOrLikeFollow(uid) :=$
$$\Pi_{uid}(User) - \Pi_{uid1}(ViewOrLikeNotFollow)$$

$Result(username, description) :=$
$$\Pi_{User.name, User.about}(User \bowtie ViewOrLikeFollow)$$

2

-Hashtags that are mentioned on any day in 2018
$TagIn2018(tag, when) :=$
$\Pi_{tag, when}\sigma_{when.year=2018}(Hashtag \bowtie Post)$

-Hashtags that are mentioned at least twice on any day in 2018
$AtLeastTwo(tag) :=$
$\Pi_{tag}\sigma_{T1.date=T2.date \wedge T1.pid \neq T2.pid}(\rho_{T1}(TagIn2018) \times (\rho_{T1}(TagIn2018))$

3

$ReciprocalFollower(uid1, uid2) :=$

$\Pi_{f1.followed, f1.follower}(\sigma_{(f1.follower=f2.followed) \wedge (f1.followed=f2.follower) \wedge (f1.follower>f1.followed)}(\rho_{f2}(Follows) \times \rho_{f1}(Follows)))$

“Follower” followed “uid1”
$uid1Follower(uid1, uid2, follower) :=$
$$\Pi_{uid1, uid2, follower}(\sigma_{uid1=followed}(ReciprocalFollower \times Follows))$$

“Follower” followed “uid2”
$uid1Follower(uid1, uid2, follower) :=$

$\Pi_{uid1,uid2,follower}(\sigma_{uid2=followed}(ReciprocalFollower \times Follows))$

“Uid1”, “uid2” are reciprocal follower, “follower” are uncommon follower of “uid1”, “uid2” $UncommonFollower(uid1, uid2, follower) :=$

$(uid1Follower \cup uid2Follower) - (uid1Follower \cap uid2Follower)$

$Result(uid1, uid2, follower, name, email) :=$
$\Pi_{uid1,uid2,follower,name,email}(\sigma_{follower=uid}(UncommonFollower \times User))$

4

cannot be expressed

5

“Uid1”, “uid2” are reciprocal follower and “uid1” ¿ “uid2”
$ReciprocalFollower(uid1, uid2) :=$
$\Pi_{f1.followed,f1.follower}(\sigma_{(f1.follower=f2.followed)\wedge(f1.followed=f2.follower)\wedge(f1.follower>f1.followed)}(\rho_{f2}(Follows)\times \rho_{f1}(Follows)))$

“Liker” liked some post “pid” of “liked”
‘$Liked(liker, liked, pid) :=$

$\Pi_{liker,Post.uid,Post.pid}(\sigma_{Likes.pid=Post.pid}(Likes \times Post))$

Join all liker in likes with all (uid, pid) in Post
$All(liker, liked, pid) :=$

$\Pi_{liker}(Likes) \times \Pi_{pid,uid}(Post)$

“Liker” did not like every post of “liked”
$NotLikedEvery(liker, liked) :=$

All - Liked

“Liker” liked every post of “liked”
$LikedEvery(liker, liked) :=$

$\Pi_{liker,liked}(Liked) - \Pi_{liker,liked}(NotLikedEvery)$

“Uid1”, “uid2” liked every post of each other
$ReciprocalLiker(uid1, uid2) :=$
$\Pi_{r1.liker,r1.liked}(\sigma_{(r1.liker=r2.liked)\wedge(r1.liked=r2.liker)}(\rho_{r1}(LikedEvery) \times \rho_{r2}(LikedEvery)))$

“Uid1”, “uid2” are backscratchers

$Backscratchers(uid1, uid2) :=$
ReciprocalFollower ∩ ReciprocalLiker

$Result(follower) :=$
$\Pi_{r1.follower}(\sigma_{(r1.follower=r2.follower)\wedge(r1.followed=Backscratchers.uid1)\wedge(r2.followed=Backscratchers.uid2)}(\rho_{r1}(Follows$
$\rho_{r2}(Follows) \times Backscratchers))$

6

"when" is the datetime of some activity of "uid"
$WhenActivity(uid, when) :=$
$\Pi_{uid,when}(Post) \cap \Pi_{uid,when}(Story)$

"when" is datetime of some activity of "followed", who is followed by "follower" with "name"
$WhenFollowedActivity(name, follower, followed, when) :=$
$\Pi_{User.name,User.uid,WhenActivity.uid,WhenActivity.when}(\sigma_{(User.uid=Follows.follower)\wedge(Follows.followed=WhenActivity.u}$
$Follows \times WhenActivity))$

For each "follower" in WhenFollowedActivity, tuple of that follower with most recent "when" is removed
$NotMostRecent(name, follower, followed, when) :=$
$\Pi_{r1.name,r1.follower,r1.followed,r1.when}(\sigma_{(r1.follower=r2.follower)\wedge(r1.when<r2.when)}(\rho_{r1}(WhenFollowedActivity)\times$
$\rho_{r2}(WhenFollowedActivity)))$

For each "follower" WhenFollowedActivity, get the tuple of that follower with most recent "when"
$MostRecent(name, follower, followed, when) :=$
WhenFollowedActivity - NotMostRecent

$Result(followerName, followedName, followedEmail, date) :=$
$\Pi_{MostRecent.name,User.name,User.email,MostRecent.when.date}(\sigma_{MostRecent.followed=User.uid}(MostRecent\times$
$User))$

7

cannot be expressed

8

-All (commenter, pid) in which commenter has left for post with pid except for the first one. $NotFirst(commenter, pid) :=$
$\Pi_{C1.commenter,C2.pid}\sigma_{C1.commenter=C2.commenter\wedge C1.when<C2.when}$
$(\rho_{C1}(Comment) \times \rho_{C1}(Comment))$

- The first comment left by commenter for post with pid
$First(commenter, pid) :=$
$(\Pi_{commenter,pid}Comment) - NotFirst$

-All (commenter, pid) in which commenter has left comment for post with pid except for the last one. $NotLast(commenter, pid) :=$
$\Pi_{C1.commenter,C2.pid}\sigma_{C1.commenter=C2.commenter \wedge C1.when>C2.when}$
$(\rho_{C1}(Comment) \times \rho_{C1}(Comment))$

-The last comment left by commenter for post with pid
$Last(commenter, pid) :=$
$(\Pi_{commenter,pid}Comment) - NotLast$

$Answer(commenter, pid) := First \cup Last$

Part 2

1

$\sigma_{Story.when>Saw.when}(Story \bowtie_{Story.sid=Saw.sid} Saw) = \emptyset$

2

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