Tianshu Zhu
1002111225

Part1

Q1

“Liker” liked some post of “liked”

Liked(liker, liked):=

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

“Viewer” viewed some story of “viewed”

Viewed(viewer, viewed):=

$\Pi_{\text{Saw.viewerid, Story.sid}}(\sigma_{Saw.sid = Story.sid} (\text{Saw} \times \text{Story}))$

“Uid1” liked some post or viewed some story of “uid2” and

“Uid1” did not followed “uid2”

NotSat(uid1, uid2):=

$\text{Liked} \cup \text{Viewed} - \Pi_{\text{follower, followed}} (\text{Follows})$

“Uid” never liked some post or viewed some story of user whom he did not follow

Sat(uid):=

$\Pi_{\text{uid}} (\text{User}) - \Pi_{\text{uid1}} (\text{NotSat})$

Report

Result(username, description):=

$\Pi_{\text{User.name, User.about}} (\sigma_{User.uid = Sat.uid} (\text{User} \times \text{Sat}))$

Q2

“Pid” is posted in 2017 along with its posted date and tag

PostTag17(pid, date, tag):=

$\Pi_{\text{Post.pid, when, Hashtag.tag}}(\sigma_{\textit{when.year = 2017} \land \textit{Post.pid = Hashtag.pid}}(\text{Post} \times \text{Hashtag}))$

Join of all dates and all tags of 2017

All(date, tag):=

$\Pi_{\text{when.date}}(\sigma_{\textit{when.year = 2017}}(\text{Post})) \times \Pi_{\text{tag}}(\text{PostTag17})$

“Tag” was mentioned in 2017 but was not mentioned everyday in 2017

NotEveryday(tag):=

$\Pi_{\text{tag}}(\text{All} - \Pi_{\text{date, tag}}(\text{PostTag17}))$

“Tag” was mentioned everyday in 2017

Everyday(tag):=

$\Pi_{\text{tag}}(\text{PostTag17}) - \text{NotEveryday}$

“Tag” was mentioned at least 3 times everyday in 2017

result(tag):=

$\Pi_{\text{Everyday.tag}}(\sigma_{\textit{(p1.tag = p2.tag= p3.tag = Everyday.tag)} \land \textit{(p1.tag} \neq \textit{p2.tag} \land \textit{p1.tag} \neq \textit{p3.tag} \land \textit{p2.tag} \neq \textit{p3.tag)}}(\rho_{p1}(\text{PostTag17}) \times \rho_{p2}(\text{PostTag17}) \times \rho_{p3}(\text{PostTag17}) \times \text{Everyday}))$

Q3

“Uid1”, “uid2” are reciprocal follower and “uid1” > “uid2”

ReciprocalFollower(uid1, uid2):=

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

“Follower” followed “uid1”

uid1Follower(uid1, uid2, follower):=

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

“Follower” followed “uid2”

uid2Follower(uid1, uid2, follower):=

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

“Uid1”, “uid2” are reciprocal follower, “follower” are uncommon follower of “uid1”, “uid2”

UncommonFollower(uid1, uid2, follower):=

$\text{(uid1Follower} \cup \text{uid2Follower)} - \text{(uid1Follower} \cap \text{uid2Follower)}$

Report

result(uid1, uid2, follower, name, email):=

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

Q4

Can not be expressed

Q5

"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 eachother

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 $\cap$ ReciprocalLiker

Report "follower" who followed some pairs of backscratchers

Result(follower):=

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

Q6

"When" is the datetime of some activity of "uid"

WhenActivity(uid, when):=

$\Pi_{uid.when}(Post) \cup \Pi_{uid,\ when}(Story)$

"When" is datetime of some activity of "followed", who is followed by "follower" with "name"

WhenFollowedActivity(name, follower, followed, when):=

$\Pi_{User.name,\ User.uid,\ \mathit{WhenActivity.uid},\ \mathit{WhenActivity.when}}(\sigma_{\mathit{User.uid = Follows.follower} \wedge \mathit{Follows.followed = WhenActivity.uid}}(User \times Follows \times WhenActivity))$

For each "follower" in WhenFollowedActivity, tuple of that follower with most recent "when" is removed

NotMostRecent(name, follower, followed, when):=

$\Pi_{r1.name,\ r1.follower,\ r1.followed,\ r1.when}(\sigma_{\mathit{r1.follower = r2.follower} \wedge \mathit{r1.when < r2.when}}(\rho_{r1}(WhenFollowedActivity) \times \rho_{r2}(WhenFollowedActivity)))$

For each "follower" WhenFollowedActivity, get the tuple of that follower with most recent "when"

MostRecent(name, follower, followed, when):=

WhenFollowedActivity - NotMostRecent

Report user with the most recent user he followed

Result(followerName, followedName, followedEmail, date):=

$\Pi_{MostRecent.name,\ User.name,\ User.email,\ MostRecent.when.date}(\sigma_{MostRecent.followed = User.uid}$

$(MostRecent \times User))$

Q7

"LikeDate" is date when "uid" liked some post which was posted on "postDate"

DateLikePost(uid, likeDate, postDate):=

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

"Uid" did not satisfy question requirement:

"users who have always liked posts in the same order as the order in which they were posted"

NotSat(uid):=

$\Pi_{\text{r1.uid}} (\sigma_{(r1.likeDate > r2.likeDate) \land (r1.postDate < r2.postDate) \land (r1.uid = r2.uid)} (\rho_{\text{r1}} (\text{DateLikePost}) \times \rho_{\text{r2}} (\text{DateLikePost})))$

"Uid" satisfied question requirement

Sat(uid):=

$\Pi_{\text{uid}} (\text{User}) - \text{NotSat}$

Report

Result(name, email):=

$\Pi_{\text{name, email}} (\sigma_{User.uid = Sat.uid} (\text{Sat} \times \text{User}))$

Q8

Can not be expressed

Q9

"Sid" was not the last story that "uid" viewed

NotLastStory(viewerid, sid):=

$\Pi_{\text{s1.viewerid, s1.when}} (\sigma_{(s1.viewerid = s2.viewerid) \land (s1.when < s2.when)} (\rho_{s1} (Saw) \times \rho_{s2} (Saw)))$

"Sid" was the last story that "uid" viewed

LastStory(viewerid, sid):=

$\Pi_{\text{viewerid, sid}} (Saw) - NotLastStory$

"Sid" was not the first story that "uid" viewed

NotFirstStory(viewerid, sid):=

$\Pi_{\text{s1.viewerid, s1.when}} (\sigma_{(s1.viewerid = s2.viewerid) \land (s1.when > s2.when)} (\rho_{s1} (Saw) \times \rho_{s2} (Saw)))$

"Sid" was the first story that "uid" viewed

FirstStory(viewerid, sid):=

$\Pi_{\text{viewerid, sid}} (Saw) - NotFirstStory$

report uid and the id of the first and of the last story he have seen

result(viewerid, firstSid, lastSid):=

$\Pi_{FirstStory.viewerid,\ FirstStory.sid,\ LastStory.sid} (\sigma_{FirstStory.viewerid = LastStory.viewerid} (FirstStory \times LastStory))$

Q10

“Pid” has at least 3 different comments

ThreeComments(pid, commentor, when, sentiment):=

$\Pi_{\text{c1.pid, c1.commentor, c1.when, sentiment(c1.text)}} (\sigma_{(c1.pid = c2.pid = c3.pid) \land (c1.commentor \neq c2.commentor \lor c1.when \neq c2.when) \land (c1.commentor \neq c3.commentor \lor c1.when \neq c3.when) \land (c2.commentor \neq c3.commentor \lor c2.when \neq c3.when)} (\rho_{c1} (\text{Comment}) \times \rho_{c2} (\text{Comment}) \times \rho_{c3} (\text{Comment})))$

“Pid” in ThreeComments, and pid has comments of at least 2 different “when”

LeastTwoWhen(pid, commentor, when, sentiment):=

$\Pi_{\text{r1.pid, p1.commentor, r1.when, r1.sentiment}} (\sigma_{(r1.pid = r2.pid) \land (r1.when \neq r2.when)} (\rho_{r1} (\text{ThreeComments}) \times \rho_{r2} (\text{ThreeComments})))$

“Pid” in ThreeComments, and pid has comments of at least 3 different “when”

LeastThreeWhen(pid, commentor, when, sentiment):=

$\Pi_{\text{r1.pid, p1.commentor, r1.when, r1.sentiment}} (\sigma_{(r1.pid = r2.pid = r3.pid) \land (r1.when < r2.when < r3.when)} (\rho_{r1} (\text{ThreeComments}) \times \rho_{r2} (\text{ThreeComments}) \times \rho_{r3} (\text{ThreeComments})))$

"Pid" in ThreeComments, and pid has comments all of the same "when". These "pid"s can not have shift

OneWhen(pid, commentor, when, sentiment):=

ThreeComments - LeastTwoWhen

"Pid" in ThreeComments, and pid has comments of exactly 2 different "when"

TwoWhen(pid, commentor, when, sentiment):=

LeastThreeWhen - LeastTwoWhen

OneWhenNoShift:= OneWhen

When there are exactly 2 different when, get "pid" with no shift

TwoWhenNoShift(pid, commentor, when, sentiment):=

$\Pi_{r1.pid,\ p1.commentor,\ r1.when,\ r1.sentiment}(\sigma_{(r1.pid = r2.pid)\ \wedge\ (r1.when \neq r2.when)\ \wedge\ (r1.sentiment = r2.sentiment)}(\rho_{r1}(TwoWhen) \times \rho_{r2}(TwoWhen)))$

When there are at least 3 different when, get "pid" with no shift

LeastThreeWhenNoShift(pid, commentor, when, sentiment):=

$\Pi_{r1.pid,\ p1.commentor,\ r1.when,\ r1.sentiment}(\sigma_{(r1.pid = r2.pid = r3.pid)\ \wedge\ (r1.when < r2.when < r3.when)\ \wedge\ (r1.sentiment \neq r2.sentiment\ \wedge\ r2.sentiment \neq r3.sentiment)}(\rho_{r1}(LeastThreeWhen) \times \rho_{r2}(LeastThreeWhen) \times \rho_{r3}(LeastThreeWhen)))$

“Pid” had at least three comments and for which there has been a sentiment shift over time

Shift(pid, commentor, when, sentiment):=

(((ThreeComments - OneWhenNoShift) - TwoWhenNoShift) - LeastThreeWhenNoShift)

Report

Result(uid, pid, commentor, date, sentiment):=

$\Pi_{\text{Post.pid, Post.uid, Shift.commentor, Shift.when.date, Shift.sentiment}} (\sigma_{Post.pid = Shift.pid} (\text{Post} \times \text{Shift}))$

Part2

Q1

$\sigma_{(Comment.pid = Post.pid)\ \wedge\ (comment.when \leq Post.when)}$ (Comment × Post) = ∅

Q2

$\sigma_{(s1.uid = s2.uid)\ \wedge\ (s1.sid \neq s2.sid)\ \wedge\ (s1.current = true\ \wedge\ s2.current = true)}$ (ρ_{s1} (Story) × ρ_{s2} (Story)) = ∅

Q3

Π_{pid} (Post) - Π_{pid} (PIncludes) = ∅

Π_{sid} (Story) - Π_{sid} (SIncludes) = ∅