```
-- __/\\\\\\\\\\\__/\\\\\_____/\\\__/\\\\\\\\\\\\\\\____/\\\\\_________/\\\\\\\\\
_________/\\\\\\\________/\\\\\\\________/\\\\\\\________/\\\\\\\\\\
________________/\\\\\\\\\_______/\\\\\\\\\_____
--  _\/////\\\///__\/\\\\\\___\/\\\_\/\\\///////////___/\\\///\\\
_____/\\\///////\\\_____/\\\/////\\\____/\\\/////\\\____/\\\/////\\\
____/\\\///////\\\_____________/\\\\\\\\\\\\\___/\\\///////\\\___
--   _____\/\\\_____\/\\\/\\\__\/\\\_\/\\\____________/\\\/__\///\\\__\///
______\//\\\___/\\\____\//\\\__/\\\____\//\\\__/\\\____\//\\\__\///______/\\\
_____________/\\\/////////\\\_\///______\//\\\__
--    _____\/\\\_____\/\\\//\\\_\/\\\_\/\\\\\\\\\\\___/\\\______\//\\\
___________/\\\/___\/\\\_____\/\\\_\/\\\_____\/\\\_\/\\\_____\/\\\_________/\\\//
_____________\/\\\_______\/\\\___________/\\\/___
--     _____\/\\\_____\/\\\\//\\\\/\\\_\/\\\///////___\/\\\_______\/\\\
________/\\\//_____\/\\\_____\/\\\_\/\\\_____\/\\\_\/\\\_____\/\\\________\///\\\
____________\/\\\\\\\\\\\\\\\________/\\\//_____
--      _____\/\\\_____\/\\\_\//\\\/\\\_\/\\\__________\//\\\______/\\\______/\\\//
________\/\\\_____\/\\\_\/\\\_____\/\\\_\/\\\_____\/\\\___________\//\\\
___________\/\\\/////////\\\_____/\\\//________
--       _____\/\\\_____\/\\\__\//\\\\\\_\/\\\___________\///\\\__/\\\______/\\\/
___________\//\\\____/\\\__\//\\\____/\\\__\//\\\____/\\\___/\\\______/\\\
____________\/\\\_______\/\\\___/\\\/___________
--
__/\\\\\\\\\\\_\/\\\___\//\\\\\_\/\\\_____________\///\\\\\/______/\\\\\\\\\\\\\\\
__\///\\\\\\\/____\///\\\\\\\/____\///\\\\\\\/___\///\\\\\\\\\/_____________\/\\\
_______\/\\\__/\\\\\\\\\\\\\\\_
--
_\///////////__\///_____\/////__\///________________\/////_______\///////////////
_____\///////________\///////________\///////_______\/////////_______________\///
________\///__\///////////////__

-- Your Name: Piripi Martin
-- Your Student Number: 1462615
-- By submitting, you declare that this work was completed entirely by yourself.

--
___________________________________________________________________________________
___________________________________________________________________________________
______________________________________
-- BEGIN Q1
SELECT postPermanentID, text
FROM post
WHERE post.postPermanentID
            NOT IN (
                  SELECT react.postID
                  FROM react
            );

-- END Q1
--
___________________________________________________________________________________
___________________________________________________________________________________
______________________________________
-- BEGIN Q2
SELECT modID, username, dateModStatus
FROM moderator
INNER JOIN user
      ON moderator.linkedUserID = user.userID
ORDER BY moderator.dateModStatus DESC
LIMIT 1;
```

```
-- END Q2
--
___________________________________________________________________________________
___________________________________________________________________________________
______________________________________
-- BEGIN Q3
SELECT postPermanentID, viewCount
FROM user
INNER JOIN post
      ON user.userID = post.authorID
WHERE viewCount > 9000
      AND username = 'axe';
-- END Q3
--
___________________________________________________________________________________
___________________________________________________________________________________
______________________________________
-- BEGIN Q4

SELECT postPermanentID, COUNT(originalPostID)
FROM postreply
INNER JOIN post
      ON postreply.originalPostID = post.postPermanentID
GROUP BY originalPostID
HAVING COUNT(*) = (
      SELECT MAX(comment_count)
      FROM(
            SELECT COUNT(*) AS comment_count
            FROM postreply
            GROUP BY originalPostID
      ) AS counts
) ;


-- this is probably not good quality code you need to test it

-- END Q4
--
___________________________________________________________________________________
___________________________________________________________________________________
______________________________________
-- BEGIN Q5
SELECT  dataURL, postchannel.channelID
FROM attachmentobject
INNER JOIN postchannel
      ON attachmentobject.postPermanentID = postchannel.postID
INNER JOIN channel
      ON channel.channelID = postchannel.channelID
WHERE channel.channelName LIKE '%dota2%';


-- END Q5
--
___________________________________________________________________________________
___________________________________________________________________________________
______________________________________
-- BEGIN Q6
```

```
SELECT channelName, heartcount
FROM (
  -- from all channels and there respective love counts
    SELECT postchannel.channelID, COUNT(*) AS heartcount
    FROM postchannel
    INNER JOIN react
            ON postchannel.postID = react.postID
    INNER JOIN post
            ON react.postID = post.postPermanentID
    WHERE react.emoji = 'love'
    GROUP BY postchannel.channelID
) AS channelheartcounts
-- join channel to get channelName
INNER JOIN channel
      ON channelheartcounts.channelID = channel.channelID
-- where heartcount = the max heartconut
WHERE heartcount = (
    SELECT MAX(heartcount)
    FROM (
        SELECT COUNT(*) AS heartcount
        FROM postchannel
        INNER JOIN react
                  ON postchannel.postID = react.postID
        INNER JOIN post
                  ON react.postID = post.postPermanentID
        WHERE react.emoji = 'love'
        GROUP BY postchannel.channelID
    ) AS maxheartcounts
);

-- END Q6
--
```

```
-- BEGIN Q7

SELECT userID, reputation, reportCount as totalModeratoreports, heartCount as
totalLoveReacts
FROM (
    -- this table counts number of love
      SELECT authorID, COUNT(*) as heartCount
    FROM react
    INNER JOIN post
            ON react.postID = post.postPermanentID
      WHERE react.emoji LIKE 'love'
      GROUP BY authorID
) AS loveCounts
-- join with user to get reputation later
INNER JOIN user
    ON user.userID = loveCounts.authorID
-- join with moderator report count table
INNER JOIN (
      SELECT authorID, COUNT(*) as reportCount
    FROM moderatorreport
    INNER JOIN post
            ON moderatorreport.postPermanentID = post.postPermanentID
    GROUP by authorID
) AS reportCounts ON user.userID = reportCounts.authorID
WHERE heartCount >= 3 AND reportCount >= 1 AND reputation < 60;


-- END Q7
--
```

```
-- BEGIN Q8

SELECT channelID, virusCount FROM (

-- getting the table of channels and virus count
SELECT channelID, COUNT(*) as virusCount
FROM postchannel
      INNER JOIN (
            SELECT postPermanentID
            FROM  attachmentobject
            WHERE attachmentobject.virusScanned = 1
            GROUP BY postPermanentID
            ) AS postswithvirus
                  ON postchannel.postID = postswithvirus.postPermanentID
      GROUP BY channelID
) AS channelVirusCount

-- comparing the virus count to the minum of the highest 3 unique values of virus
count in the table
WHERE virusCount >= (
      SELECT MIN(virusCOUNT)
    FROM (
            SELECT DISTINCT viruscount
        FROM
                  (
            SELECT channelID, COUNT(*) as virusCount
                  FROM postchannel
                  INNER JOIN (
                        SELECT postPermanentID
                        FROM  attachmentobject
                        WHERE attachmentobject.virusScanned = 1
                        GROUP BY postPermanentID
                  ) AS postswithvirus
                        ON postchannel.postID = postswithvirus.postPermanentID
                  GROUP BY channelID
            ) as channelVirusCount
            ORDER BY virusCount DESC
            LIMIT 3
      )AS topUniqueCounts
)
ORDER BY virusCount DESC;




-- END Q8
--
```

```
-- BEGIN Q9

SELECT modID, COUNT(*) as numberOfDisciplinariesToRepeaters
-- getting table of mod reports so we can cross reference the user
FROM moderatorreport
INNER JOIN post
      ON moderatorreport.postPermanentID = post.postPermanentID
INNER JOIN user
      ON post.authorID = user.userID
-- choosing reports where they took action and the user was in the table of
offenders

WHERE disciplinaryAction = 1
      AND userID IN (
      -- Selecting authors that appear in at least two channels from:
            SELECT authorID
        FROM (
                  -- the list of authors and the channel they are reported in
                  SELECT authorID, channelID
                  FROM post
            INNER JOIN moderatorreport
                        ON post.postPermanentID = moderatorreport.postPermanentID
                  INNER JOIN postchannel
                        ON postchannel.postID = post.postPermanentID
                  ) AS authorReportedChannels
                  GROUP BY authorID
                  HAVING COUNT(DISTINCT channelID) >= 2
      )
GROUP BY modID;




-- END Q9
--
```

```
-- BEGIN Q10

SELECT userID
FROM user
-- table of posts in ranked grind before the date
WHERE userID NOT IN  (
      -- this is a table with all posts and replies in the channel ranked_grind
before the date
      SELECT authorID
    FROM post
    INNER JOIN postchannel
            ON post.postPermanentID = postchannel.postID
      INNER JOIN channel
            ON postchannel.channelID = channel.channelID
      WHERE channel.channelName LIKE 'ranked_grind'
            AND post.dateCreated < 01/04/2024
)

AND userID IN  (
-- table with lists of all user replys in dota2_memes after the given date
      SELECT post.authorID
    FROM postreply
    INNER JOIN postchannel
            ON postreply.replyPostID = postchannel.postID
      INNER JOIN channel
            ON postchannel.channelID = channel.channelID
      INNER JOIN post
            ON postreply.replyPostID = post.postPermanentID
      WHERE channel.channelName LIKE 'dota2_memes'
            AND post.dateCreated >= 01/04/2024
);




-- END Q10
--
```

```
-- END OF ASSIGNMENT Do not write below this line
```