Appendix 2: Queries

1. LOGICAL DATABASE DESIGN

To specify the queries we use in the challenge, first, the database schema is defined. It should be noted that the schema is only used for specifying the semantics of those queries. The dataset provided is not necessarily to be stored in a database using the schema. Actually, the dataset is not even necessarily to be stored in a relational database. Attendees are required to build a system that can return results that are identical to the results of these queries under the given schema in the given dataset.

Five tables, i.e.friendList, microblog, event, mention and retweet, are defined in the schema.The schema is defined as follows.

The following list defines the data types that are used:

- **Variable text, size N** means that the attribute must be able to hold any string of characters of a variable length with amaximum length of N. If the attribute is stored as a fixed length string and the string it holds is shorter than N characters, it must be padded with spaces.
- **Date and time**represents the data type for a date value that includes a time component. The date component must be able to hold any date between 2009-8-24 and 2011-13-31. The time component must be capable of representing the range of time values from 00:00:00 to 23:59:59 with a resolution of at least one second. Date and Time must be implemented using data types that defined by the DBMS for that use.

FRIENDLIST Table Layout

Field Name	Field Definition	Comments
UID	variable text, size 200	non-nullable
FRIENDID	variable text, size 200	non-nullable

Primary Key:(UID, FRIENDID)

Comment:FriendListdescribes the information of followshipbetween users. The attribute values of UID and FRIENDID are both userID, whichuniquely identifies auser.Each row of the table represents that UID followed FRIENDID.A user may follow several users. Similarly,a user may be followed by several users.

MICROBLOGTable Layout

Field Name	Field Definition	Comments
MID	variable text, size 64	non-nullable
UID	variable text, size 200	non-nullable
TIME	date and time	non-nullable

Primary Key:MID

Comment:Microblogdescribes some basic information about a micro-blog, including userID (UID), timestamp (TIME), and a unique identifier (MID). It contains both original tweets and re-tweeted posts.

EVENT Table Layout

Field Name	Field Definition	Comments
MID	variable text, size 64	non-nullable
TAG	variable text, size 200	non-nullable

Primary Key:(MID, TAG)

(MID)Foreign Key, referencesMICROBLOG(MID)

Comment:Event records tags ofmicro-blogs.Amicro-blogmay containseveral tags. The micro-blogcan be divided into original tweets and re-tweeted posts. If the micro-blogis original, thenthe table only recordsthe micro-blog’s tags. If the micro-blogisa re-tweeted post, then it is annotated with the tags ofthe re-tweeted original tweet.

MENTION Table Layout

Field Name	Field Definition	Comments
MID	variable text, size 64	non-nullable
UID	variable text, size 200	non-nullable

Primary Key: (MID, UID)

(MID)Foreign Key, referencesMICROBLOG (MID)

Comment: Mention recordsuser IDsthatamicro-blogmentioned.Amicro-blog maymention many users. If amicro-blog is original, then the table only recordsusers ofthe micro-blog mentioned. If amicro-blog is a re-tweeted post, then the table records users whom there-tweeted original tweet mentioned, plus user IDsthat appearing in the re-tweeting path.

RETWEET Table Layout

Field Name	Field Definition	Comments
MID	variable text, size 64	non-nullable
REMID	variable text, size 200	non-nullable
PATH	variable text, size 700	nullable

Primary Key: (MID,REMID)

(MID,REMID)Foreign Key, referencesMICROBLOG (MID,MID)

Comment: Retweet records information of re-tweeted posts. If a micro-blog is a re-tweeted post, then the table will record the post's ID as MID, the re-tweeted original post's ID as REMID, the path of the re-tweeted post as PATH. The path is a set of userID ordered by time.

2. Workload

There are nineteen typical queries. Values of variables in each query are randomly selected by BSMA performance testing tool from a predefined lists.There are five variables in all queries, including userID, timestamp, tag, top-x and timeRange. The value of user ID can be represented by symbol "A" or "B" in queries. "?"represents content of a tag, which is the label of an event."YYYY-MM-DDHH:MM:SS" means a timestamp.The possible value of x in top-x is the number of 10, 50,or 100.The timeRange may be one hour, one day, one week,or one year. We just

usetop-10 or one hour in the queries below, while BSMA may determine values of these two parameters automatically.

The specification of required output of queries is given as well. There are two parts of output: 1) parameter values, and 2) query results. The first part (parameter values) should be represented by a sequence of components, each of which is in the form of "Att=Val". Here, Att is the name of parameter,while Val is the value used in the query.Components should be separated by comma: ','.There are five related parameters. They are:userID, timestamp, tag, top-x andtimeRange.The values of top-x and timeRange can be obtainedautomatically by BSMA.The others should be outputted.

There are some terms that are explained as follows:

Followee:
User A's followees are users who were followed by A.
Follower:
User A's followers are users who followed A.
r-friend:
User A's r-friends are users who followed A and were followed by A

1. Find top-x (x may be 10, 50, or 100) suggested followees for user A: Those people that are currently not users A's r-friends but are r-friends of many of A'sr-friends. Get all r-friends of A's r-friends, order them by the number of people in A's r-friends list connecting to them.

```
SELECT f1.uid
FROM
    (SELECT a.friendID AS uid
     FROM friendList AS a JOIN friendList AS b
         ON a.friendID = b.uid
     WHERE a.uid = b.friendID AND a.uid = "A") AS f,
    friendList AS f1 JOIN friendList AS f2
    ON f1.friendID = f2.uid
WHERE f2.uid = f.uid AND
       f1.friendID = f2.uid AND
       f1.uid = f2.friendID AND
       f1.uid<>"A" AND
    f1.uid<>f.uid
GROUP BY f1.uid
ORDER BY COUNT(f2.uid)DESC
```

```
LIMIT 10;
```

2. Find out the top-x (x may be 10, 50, or 100) users for user A:Those people that are currently not user A's followees but are followers of many of A'sfollowees. Get all followers of A's followees. Order them bythe number of people in A's followeeslist being followed by them.

```
SELECT f1.uid
FROM friendList AS f1,
     (SELECT friendID
   FROM friendList
   WHERE uid = "A") AS f2
WHERE f1.uid <> "A" AND
      f1.friendID = f2.friendID AND
      f1.uid<> f2.friendID
GROUP BY f1.uid
ORDER BY COUNT(f1.friendID)DESC
LIMIT 10;
```

3. Find out thetop-x (x may be 10, 50, or 100) users for user A:Those people that are currently not user A's followees but are followees of many of A'sfollowees.Getallfollowees of A's followees.Order them by the number of people in A's followees list following them.

```
SELECT f.friendID
FROM
    (SELECT *
     FROM friendList
     WHERE uid IN
            (SELECT friendID
           FROM friendlist
           WHERE uid = "A")) AS f
WHERE f.friendID<> "A" AND
      f.friendID NOT IN
        (SELECT friendID
       FROM friendList
       WHERE uid = "A")
GROUP BY f.friendID
ORDER BY COUNT(f.uid)DESC
LIMIT 10;
```

4. Find out users who be followed by user A and B together.

```
SELECT DISTINCT f1.friendID
FROM
    (SELECT friendID
```

```
    FROM friendList
    WHERE uid = "A") AS f1,
   (SELECT friendID
   FROM friendList
   WHERE uid = "B") AS f2
WHERE f1.friendID = f2.friendID;
```

5. Find out users who are B's followers and user A's followees.

```
SELECT uid
FROM friendList
WHERE friendID = "B"AND
      uid IN
        (SELECT friendID
       FROM friendList AS a
       WHERE a.uid = "A");
```

Comment: the query contains two user IDs(user A and B), the order of user IDscannot be reversed, A should be front of B.

6. Find out top-x (x may be 10, 50, or 100) users: Those people would be all people except user A. Get all people except A.Order them by the number of people who be mentioned by A's micro-blogs being mentioned by their micro-blogs.And the time of their micro-blogs should be in a timeRange from a timestamp.

```
SELECT microblog.uid
FROM microblog,mention,
   (SELECT DISTINCT mention.uid AS uid
    FROM microblog,mention
WHERE microblog.mid = mention.mid AND
         microblog.uid = "A") AS x
WHERE microblog.mid = mention.mid AND
     mention.uid = x.uid AND
     microblog.uid<> "A" AND
     microblog.time BETWEEN TO_DAYS('YYYY-MM-DDHH:MM:SS')
     AND DATE_ADD('YYYY-MM-DD HH:MM:SS',INTERVAL 1HOUR)
GROUP BY microblog.uid
ORDER BY COUNT(mention.uid) DESC
LIMIT 10;
```

7. Find out the top-x (x may be 10, 50, or 100) usersordered by the number of being mentioned by all micro-blogs.And the time of the micro-blogs should be in a timeRange from a timestamp.

```
SELECT mention.uid
FROM microblog,mention
WHERE microblog.mid = mention.mid AND
     microblog.time BETWEEN TO_DAYS('YYYY-MM-DDHH:MM:SS')
     AND DATE_ADD('YYYY-MM-DD HH:MM:SS',INTERVAL 1HOUR)
GROUP BY mention.uid
ORDER BY COUNT(*)DESC
LIMIT 10;
```

8. Show top-x (x may be 10, 50, or 100) latest micro-blogs from user A's followeesor the followees of them (order by the posting time)

```
SELECT mid
FROM microblog
WHERE microblog.uid IN
        (SELECT friendID
       FROM friendList
       WHERE uid = "A" OR
         uid IN
           (SELECT friendID
   FROM friendList
      WHERE uid = "A"))
ORDER BY microblog.time DESC
LIMIT 10;
```

9. Find out top-x (x may be 10, 50, or 100)users who most interested the tag "?": Those people are A's followees or followees of them. Get all A's followees and followees of them.Order them by the number of people who mentioned "?" in their micro-blogs.And the time of their micro-blogs should be in a timeRange from a timestamp.

```
SELECT microblog.uid
FROM microblog,event
WHERE microblog.mid = event.mid AND
event.tag = "?" AND
microblog.uid IN
       (SELECT friendID
        FROM friendList
       WHERE uid = "A"OR
              uid IN
                (SELECT friendID
              FROM friendList
              WHERE uid = "A")) AND
     microblog.time BETWEEN TO_DAYS('YYYY-MM-DDHH:MM:SS')
     AND DATE_ADD('YYYY-MM-DD HH:MM:SS',INTERVAL 1HOUR)
```

```
GROUP BY microblog.uid
ORDER BY COUNT(*)DESC
LIMIT 10;
```

10. Find out the top-x (x may be 10, 50, or 100) usersordered by the number of their micro-blogs beingre-tweetedby others.And the time of the retweetmicro-blogs should be in a timeRange from a timestamp.

```
SELECT x.reuid
FROM microblog,
    (SELECT retweet.mid as mid, microblog.uid as reuid
     FROM microblog,retweet
WHERE microblog.mid = retweet.remid) AS x
WHERE microblog.mid = x.mid AND
      microblog.time BETWEEN TO_DAYS('YYYY-MM-DD HH:MM:SS')
      AND DATE_ADD('YYYY-MM-DD HH:MM:SS', INTERVAL 1HOUR)
GROUP BY x.reuid
ORDER BY COUNT(*) DESC
LIMIT 10;
```

11. Find out top-x (x may be 10, 50, or 100) usersordered by the number of their micro-blogsretweeting user A's micro-blogs, and the time of the retweetmicro-blogs should be in a timeRange from a timestamp.

```
SELECT microblog.uid
FROM microblog,
     (SELECT retweet.mid as mid,microblog.uid as reuid
      FROM microblog, retweet
      WHERE microblog.mid = retweet.remid) AS x
WHERE microblog.mid = x.mid AND
      x.reuid = "A" AND
      microblog.time BETWEEN TO_DAYS('YYYY-MM-DDHH:MM:SS')
      AND DATE_ADD('YYYY-MM-DD HH:MM:SS',INTERVAL 1HOUR)
GROUP BY microblog.uid
ORDER BY COUNT(*)DESC
LIMIT 10;
```

12. Find out top-x (x may be 10, 50, or 100)micro-blogs: Thosemicro-blogs from A's followees or followees of them. Get allmicro-blogs from A's followees or followees of them.Order them by the number of being re-tweeted by others.And the time of the retweetmicro-blogs should be in a timeRange from a timestamp.

```
SELECT x.remid
FROM microblog,
```

```
    (SELECT retweet.mid AS mid,retweet.remid AS remid
     FROM microblog,retweet
     WHERE microblog.mid = retweet.remid) AS x
WHERE microblog.mid = x.mid AND
   microblog.uid IN
       (SELECT friendID
       FROM friendList
       WHERE uid = "A" OR
              uid IN
                (SELECT friendID
              FROM friendList
              WHERE uid = "A")) AND
     microblog.time BETWEEN TO_DAYS('YYYY-MM-DDHH:MM:SS')
     AND DATE_ADD('YYYY-MM-DD HH:MM:SS',INTERVAL 1HOUR)
GROUP BY x.remid
ORDER BY COUNT(*)DESC
LIMIT 10;
```

13. Find out top-x (x may be 10, 50, or 100)users: Those users are not user A's followees. Get all users are not A's followees.Order them by the number of tags which be mentioned by A's micro-blogs and their micro-blogs.And the time of the micro-blogs should be in a timeRange from a timestamp.

```
SELECT microblog.uid
FROM microblog,event
WHERE microblog.mid = event.mid AND
      microblog.uid<> "A" AND
      event.tag IN
        (SELECT DISTINCT tag
         FROM event,microblog
        WHERE microblog.mid = event.mid AND
           microblog.uid = "A") AND
     microblog.uid NOT IN
        (SELECT friendID
        FROM friendList
        WHERE uid = "A")AND
     microblog.time BETWEEN TO_DAYS('YYYY-MM-DD HH:MM:SS')
     AND DATE_ADD('YYYY-MM-DD HH:MM:SS', INTERVAL 1HOUR)
GROUP BY microblog.uid
ORDER BY COUNT(event.tag)DESC
LIMIT 10;
```

14. Find out top-x (x may be 10, 50, or 100)tagsordered by the number of being mentioned by themicro-blogs.And the time of the micro-blogs should be in a timeRange from a timestamp.

```
SELECT event.tag
FROM microblog,event
WHERE microblog.mid = event.mid AND
      microblog.time BETWEEN TO_DAYS('YYYY-MM-DD HH:MM:SS)
      AND DATE_ADD('YYYY-MM-DD HH:MM:SS', INTERVAL 1HOUR)
GROUP BY event.tag
ORDER BY COUNT(*) DESC
LIMIT 10;
```

15. Find out top-x (x may be 10, 50, or 100) users ordered by the number of their micro-blogs containingtag “?”.And the time of the micro-blogs should bein a timeRange from a timestamp.

```
SELECT microblog.uid
FROM microblog,event
WHERE microblog.mid = event.mid AND
     event.tag="?" AND
     microblog.time BETWEEN TO_DAYS('YYYY-MM-DDHH:MM:SS')
     AND DATE_ADD('YYYY-MM-DD HH:MM:SS',INTERVAL 1HOUR)
GROUP BY microblog.uid
ORDER BY COUNT(*)DESC
LIMIT 10;
```

16. Find out top-x (x may be 10, 50, or 100)users: Thoseusers that would beanyones. Get all users, order them by the number ofthe micro-blogsbeing re-tweeted by themmicro-blogsfrom userA’s followees.And the time of the retweetmicro-blogs should be in a timeRange from a timestamp.

```
SELECT x.reuid
FROM microblog,
    (SELECT retweet.mid as mid,microblog.uid as reuid
     FROM microblog,retweet
     WHERE microblog.mid = retweet.remid) AS x
WHERE microblog.mid = x.mid AND
     microblog.uid IN
       (SELECT friendID
        FROM friendList
        WHERE uid = "A") AND
     microblog.time BETWEEN TO_DAYS('YYYY-MM-DD HH:MM:SS')
     AND DATE_ADD('YYYY-MM-DD HH:MM:SS’,INTERVAL 1HOUR)
GROUP BY x.reuid
ORDER BY COUNT(*)DESC
LIMIT 10;
```

17. Find out top-x (x may be 10, 50, or 100) users: Those users that are A's followees. Get all A's followees.Order them by the number of being mentioned by all the micro-blogs. And the time of the micro-blogs should bein a timeRange from a timestamp.

```
SELECT mention.uid
FROM microblog,mention
WHERE microblog.mid = mention.mid AND
     microblog.uid IN
       (SELECT friendID
        FROM friendList
        WHERE uid = "A")AND
     microblog.time BETWEEN TO_DAYS('YYYY-MM-DDHH:MM:SS')
     AND DATE_ADD('YYYY-MM-DD HH:MM:SS',INTERVAL 1HOUR)
GROUP BY mention.uid
ORDER BY COUNT(*)DESC
LIMIT 10;
```

18. Find out top-x (x may be 10, 50, or 100)users: Those users that are A's followers. Get all A's followers, order them by the number of their micro-blogsmentioning A, and the time of the micro-blogs should bein a timeRange from a timestamp.

```
SELECT mention.uid
FROM microblog,mention
WHERE microblog.mid = mention.mid AND
      mention.uid="A" AND
      microblog.uid IN
        (SELECT uid
         FROM friendList
         WHERE friendID = "A") AND
     microblog.time BETWEEN TO_DAYS('YYYY-MM-DD HH:MM:SS')
     AND DATE_ADD('YYYY-MM-DD HH:MM:SS', INTERVAL 1HOUR)
GROUP BY mention.uid
ORDER BY COUNT(*) DESC
LIMIT 10;
```

19. Find out top-x (x may be 10, 50, or 100)trendingtags: Thosetags that come from the micro-blogs ofA's followees or followees of them. Get all tags fromthe micro-blogs ofA's followees and followees of them.Order them by the number ofbeing mentioned by the micro-blogs of A's followees or followees of them.And the time of the micro-blogs should be in a timeRange from a timestamp.

```
SELECT event.tag
FROM microblog,event
```

```
WHERE microblog.mid = event.mid AND
microblog.uid<> "A" AND
microblog.uid IN
       (SELECT friendID
        FROM friendList
        WHERE uid = "A" OR
              uid IN
                (SELECT friendID
                 FROM friendList
                 WHERE uid = "A")) AND
   microblog.time BETWEEN TO_DAYS('YYYY-MM-DD HH:MM:SS')
   AND DATE_ADD('YYYY-MM-DD HH:MM:SS', INTERVAL 1HOUR)
GROUP BY event.tag
ORDER BY COUNT(*)DESC
LIMIT 10;
```