CZ2007 Lab 4 Database Schema, Queries and Triggers

SSR1 Group 5

Ang Wan Qi, An Zheyuan, Janssen Tan, Lee Suyeon, Thong Hoi Wei

CREATING TABLES

```
CREATE TABLE UserTb (
        UserID          varchar(200)   NOT NULL,
        name            varchar(MAX),
        sex             varchar(MAX),
        email           varchar(200),
        hometown        varchar(MAX),
        birthday        date,
        PRIMARY KEY (UserID),
        UNIQUE (email)
);

CREATE TABLE Employer (
        employerName        varChar(200)  NOT NULL,
        PRIMARY KEY (employerName),
);

CREATE TABLE WorkedAt (
        UserID              varChar(200)  NOT NULL,
        employerName        varChar(200)  NOT NULL,
        city                varChar(200),
        Position            varChar(MAX),
        startDate           date,
        endDate             date,
        PRIMARY KEY (UserID, employerName),
        FOREIGN KEY (UserId) REFERENCES UserTb(UserId),
        FOREIGN KEY (employerName) REFERENCES Employer(employerName),
        CONSTRAINT          chk_start_end          CHECK (endDate >startDate),
);

CREATE TABLE School (
        schoolName    varChar (200) NOT NULL,
        schoolType    varChar(MAX),
        PRIMARY KEY (schoolName),
);

CREATE TABLE Attended (
        UserID          varChar(200) NOT NULL,
        schoolName      varChar(200) NOT NULL,
        classYear       int,
        PRIMARY KEY (UserID, schoolName),
```

```
        FOREIGN KEY (UserId) REFERENCES UserTb(UserId),
        FOREIGN KEY (schoolName) REFERENCES School(schoolName)
);


CREATE TABLE Message (
        timeInstant     datetime NOT NULL,
        senderID        varChar(200) NOT NULL,
        receiverID      varChar (200),
        content         varChar (max),
        PRIMARY KEY (senderID, timeInstant),
        FOREIGN KEY (senderID) REFERENCES UserTb(UserID),
        FOREIGN KEY (receiverID) REFERENCES UserTb(UserID)
);

CREATE TABLE Comment (
        msg_timeInstant             datetime,
        msg_senderID                varchar(200),
        comment_UserID              varchar(200)   NOT NULL,
        comment_timeInstant         datetime       NOT NULL,
        content                     varchar(max),
        CONSTRAINT chk_date CHECK (comment_timeInstant > msg_timeInstant),
        PRIMARY KEY (comment_UserID, comment_timeInstant),
        FOREIGN KEY (comment_UserID) REFERENCES UserTb(UserID),
        FOREIGN KEY (msg_senderID, msg_timeInstant) REFERENCES
        Message(senderID, timeInstant),
);

CREATE TABLE POI_1 (
        POI_ID varchar(200) not null,
        name varchar(max),
        description_text varchar(max),
        coord_lat float,
        coord_long float,
        category varchar(max),
        contactInfo varchar(max),

        PRIMARY KEY (POI_ID),
        UNIQUE(coord_lat, coord_long)
);

CREATE TABLE POI_2 (
coord_lat float  NOT NULL,
coord_long float not null,
addressText    varchar(max),
PRIMARY KEY (coord_lat, coord_long),
```

```
FOREIGN KEY (coord_lat, coord_long) REFERENCES POI_1 (coord_lat, coord_long)
);

CREATE TABLE Photo (
        photoID         varChar (200) NOT NULL,
        UserID          varChar (200),
        POI_ID          varChar (200),
        caption         varChar (MAX),
        PRIMARY KEY (photoID),
        FOREIGN KEY (UserID) REFERENCES UserTb(UserID),
        FOREIGN KEY (POI_ID) REFERENCES POI_1(POI_ID)
);

CREATE TABLE CheckIn (
        UserID          varChar(200)  NOT NULL,
        chkIn_date      date    NOT NULL,
        chkIn_time      time    NOT NULL,
        POI_ID          varChar(200)  NOT NULL,
        shout           varChar(MAX),
        PRIMARY KEY (UserID, chkIn_date, chkIn_time),
        FOREIGN KEY (UserID) REFERENCES UserTb(UserID),
        FOREIGN KEY (POI_ID) REFERENCES POI_1(POI_ID),
);

CREATE TABLE Tip (
        UserID          varChar(200)  NOT NULL,
        timeInstant     datetime        NOT NULL,
        POI_ID          varChar(200)  NOT NULL,
        content         varChar(Max),
        PRIMARY KEY (UserID, timeInstan`t),
        FOREIGN KEY (UserID) REFERENCES UserTb(UserID),
        FOREIGN KEY (POI_ID) REFERENCES POI_1(POI_ID),
);

CREATE TABLE Category (
        name   varchar(200) NOT NULL,
        PRIMARY KEY (name)
);

CREATE TABLE Has (
        POI_ID                  varchar(200)   NOT NULL,
        categoryName            varchar(200)   NOT NULL,
        PRIMARY KEY (POI_ID, categoryName),
        FOREIGN KEY (POI_ID) REFERENCES POI_1(POI_ID),
        FOREIGN KEY (categoryName) REFERENCES Category(name)
);
```

```
CREATE TABLE Friends(
        ownerID         varchar(200)   NOT NULL,
        friendID        varchar(200)   NOT NULL,
        PRIMARY KEY (ownerID, friendID),
        CONSTRAINT chk_user CHECK (ownerID <> friendID),
        FOREIGN KEY (ownerID) REFERENCES UserTb(UserID),
        FOREIGN KEY (friendID) REFERENCES UserTb (UserID),
);

CREATE TABLE FriendList(
        ownerID         varchar(200)   NOT NULL,
        listName        varchar(200)   NOT NULL,
        PRIMARY KEY (ownerID, listName),
        FOREIGN KEY (ownerID) REFERENCES UserTb(UserId)
);

CREATE TABLE Contain(
        ownerID         varchar(200)   NOT NULL,
        listName        varchar(200)   NOT NULL,
        UserID          varchar(200)   NOT NULL,
        PRIMARY KEY (ownerID, listName, UserID),
        CONSTRAINT chk_id CHECK (ownerID <> UserID),
        FOREIGN KEY (UserID) REFERENCES UserTb(UserID),
        FOREIGN KEY (ownerID) REFERENCES FriendList(ownerID),
        FOREIGN KEY (listName) REFERENCES FriendList(listName)
);
```

INSERTING TUPLES INTO Datebase

```
INSERT INTO dbo.UserTb
(UserID, name, sex, email, hometown, birthday)
VALUES
('1',  'James',   'M',  'james@gmail.com',   'Singapore',  '1990-01-01'),
('2',  'John',    'M',  'john@gmail.com',    'Singapore',  '1990-01-02'),
('3',  'Robert',  'M',  'robert@gmail.com',  'Singapore',  '1990-01-03'),
('4',  'David',   'M',  'david@gmail.com',   'Singapore',  '1990-01-04'),
('5',  'George',  'M',  'george@gmail.com',  'Singapore',  '1990-01-05'),
('6',  'Mary',    'F',  'mary@gmail.com',    'Singapore',  '1990-01-06'),
('7',  'Lisa',    'F',  'lisa@gmail.com',    'Singapore',  '1990-01-07'),
('8',  'Helen',   'F',  'helen@gmail.com',   'Singapore',  '1990-01-08'),
('9',  'Karen',   'F',  'karen@gmail.com',   'Singapore',  '1990-01-09'),
('10', 'Ruth',    'F',  'ruth@gmail.com',    'Singapore',  '1990-01-10');
```

```
INSERT INTO dbo.Employer
(employerName)
VALUES
('Company A'),
('Company B'),
('Company C'),
('Company D'),
('Company E'),
('Company F'),
('Company G'),
('Company H'),
('Company I')
('Company J');
```

```
INSERT INTO dbo.WorkedAt
(UserId, employerName, city, position, startDate, endDate)
VALUES
('1',  'Company A',  'Singapore',  'Manager',    '2018-01-01',  NULL),
('2',  'Company A',  'Singapore',  'Engineer',   '2017-01-01',  '2018-12-31'),
('3',  'Company B',  'Singapore',  'Manager',    '2018-01-01',  NULL),
('4',  'Company B',  'Singapore',  'Engineer',   '2019-01-01',  NULL),
('5',  'Company C',  'Singapore',  'Associate',  '2017-01-01',  '2018-12-31'),
('6',  'Company C',  'Singapore',  'Engineer',   '2019-01-01',  NULL),
('7',  'Company C',  'Singapore',  'Designer',   '2019-01-01',  NULL),
('8',  'Company D',  'Singapore',  'Engineer',   '2019-01-01',  NULL),
('9',  'Company D',  'Singapore',  'Manager',    '2019-01-01',  NULL),
('10', 'Company D',  'Singapore',  'Engineer',   '2019-01-01',  NULL);
```

```
INSERT INTO dbo.School
(schoolName, schoolType)
VALUES
('School A',     'Graduate School'),
('School B',     'College'),
('School C',     'High School'),
('School D',     'Middle School'),
('School E',     'Elementary School');


INSERT INTO dbo.Attended
(UserId, schoolName, classYear)
VALUES
(‘1’,     'School A',     2019),
(‘1’,     'School B',     2016),
(‘1’,     'School C',     2014),
(‘2’,     'School B',     2019),
('2',     'School A',     2018),
('2',     'School C',     2014),
(‘3’,     'School B',     2019),
(‘4’,     'School C',     2019),
(‘5’,     'School C',     2019),
(‘6’,     'School C',     2019),
(‘7’,     'School C',     2019),
(‘8’,     'School C',     2019),
('8',     'School B',     2016),
(‘9’,     'School C',     2019),
('9',     'School B',     2016),
(‘10’,    'School C',     2019),
(‘10’,    ‘School B’,     2016);

INSERT INTO dbo.Message
(timeInstant, senderID, receiverID, content)
VALUES
('2019-04-06 12:00:00.000',   1,      1,      'Loving the weather today!'),
('2019-04-06 13:00:00.000',   2,      1,      'It was a fun day with you!'),
('2019-04-06 14:00:00.000',   3,      1,      'Hi User 1!’),
('2019-04-06 15:00:00.000',   4,      1,      'Thanks for today!'),
('2019-04-06 16:00:00.000',   5,      1,      'Congratulations'),
('2019-04-06 17:00:00.000',   2,      1,      'It’s me again!'),
('2019-04-06 18:00:00.000',   1,      1,      'This is my wall!'),
('2019-04-07 00:00:00.000',   1,      2,      'Hi User 2!'),
('2019-04-08 00:00:00.000',   1,      3,      'Hi User 3!'),
('2019-04-09 00:00:00.000',   1,      4,      'Hi User 4!'),
('2019-04-10 00:00:00.000',   1,      5,      'Hi User 5!'),
('2019-04-11 00:00:00.000',   1,      6,      'Hi User 6!');
```

```
INSERT INTO dbo.Photo
(photoID, userId, POI_Id, caption)
VALUES
(1,     1,      1,      'Amusement Park 1!'),
(2,     1,      2,      'Museum 1!'),
(3,     2,      3,      'Shopping 2!'),
(4,     2,      4,      'Resort 2!'),
(5,     3,      5,      'Airport 3!'),
(6,     3,      1,      'Amusement Park 3!'),
(7,     4,      2,      'Museum 4!'),
(8,     4,      3,      'Shopping 4!'),
(9,     5,      4,      'Resort 5!'),
(10,    5,      5,      'Airport 5!');


INSERT INTO dbo.CheckIn
(UserID, chkIn_date, chkIn_time, POI_ID, shout)
VALUES
(2,     '2019-04-06', '12:00:00.000',  1,      'Having fun at POI 1!'),
(2,     '2019-04-07', '12:01:00.000',  1,      'Back at POI 1!'),
(2,     '2019-04-06', '12:02:00.000',  4,      'Check In to POI 4'),
(2,     '2019-04-06', '12:03:00.000',  5,      'Having fun at POI 5!'),
(3,     '2019-04-06', '12:00:00.000',  2,      'Having fun at POI 2!'),
(3,     '2019-04-06', '12:10:00.000',  3,      'Love POI 3!');
//
(4,     '2019-04-06', '12:00:00.000',  3,      'Having fun at POI 3!'),
(4,     '2019-04-07', '12:01:00.000',  3,      'Back at POI 3!'),
(6,     '2019-04-07', '12:10:00.000',  3,      'Love POI 3!'),
(7,     '2019-04-06', '12:20:00.000',  3,      'Check In to POI 3'),
(8,     '2019-04-06', '12:30:00.000',  5,      'Having fun at POI 5!'),
(9,     '2019-04-06', '12:40:00.000',  2,      'Having fun at POI 2!'),
(10,    '2019-04-06', '12:50:00.000',  2,      'Love POI 2!');



INSERT INTO dbo.Tip
(UserId, timeInstant, POI_ID, content)
VALUES
(1,     '2019-04-06 13:01:00.000',    1,      'Put lots of suncreen! Tiring but excellent
place'),
(1,     '2019-04-06 13:02:00.000',    2,      'Take photos at the photozone!'),
(1,     '2019-04-06 13:03:00.000',    3,      'Check out Store A for exclusive products!'),
(2,     '2019-04-06 13:04:00.000',    4,      'Go to the swimming pool!'),
(2,     '2019-04-06 13:05:00.000',    5,      'Be there early!');
```

```
INSERT INTO dbo.Has
(POI_ID, categoryName)
VALUES
(1,       'Amusement Park'),
(2,       'Museum'),
(3,       'Shopping Centre'),
(4,       'Resort'),
(5,       'Airport'),
('6',     'Hawker Centre'),
('7',     'Hawker Centre'),
('8',     'Shopping Centre'),
('9',     'Shopping Centre'),
('10',    'Museum');

INSERT INTO dbo.Category
(name)
VALUES
('Amusement Park'),
('Museum'),
('Shopping Centre'),
('Resort'),
('Airport')
(‘Hawker Centre’);

INSERT INTO dbo.Friends
(ownerID, friendID)
VALUES
('1',     '2'),
('1',     '3'),
('2',   '4'),
('2',     '6'),
('2',     '7'),
('2',     '8'),
('2',     '9'),
('2',     '10'),
('3',     '6'),
('3',     '7'),
('3',     '8'),
('3',     '9'),
('3',     '10'),
('6',     '7'),
('6',     '8')
;
```

```
INSERT INTO dbo.FriendList
(ownerID, listName)
VALUES
(1,     'Friends'),
(1,     'News Feed'),
(2,     'Friends'),
(2,     'News Feed'),
(3,     'Friends'),
(3,     'News Feed'),
(4,     'Friends'),
(4,     'News Feed'),
(5,     'Friends'),
(5,     'News Feed'),
(6,     'Friends'),
(6,     'News Feed'),
(7,     'Friends'),
(7,     'News Feed'),
(8,     'Friends'),
(8,     'News Feed'),
(9,     'Friends'),
(9,     'News Feed'),
(10,    'Friends'),
(10,    'News Feed');

INSERT INTO dbo.Contain
(ownerID, listName, UserID)
VALUES
(1,     'Friends',      2),
(2,     'Friends',      1),
(1,     'News Feed',    2),
(2,     'News Feed',    1),
(1,     'Friends',      3),
(3,     'Friends',      1),
(1,     'News Feed',    3),
(3,     'News Feed',    1),
(1,     'Friends',      4),
(4,     'Friends',      1),
(1,     'News Feed',    4),
(4,     'News Feed',    1),
(1,     'Friends',      5),
(5,     'Friends',      1),
(1,     'News Feed',    5),
(5,     'News Feed',    1);


INSERT INTO dbo.POI_1
```

```
(POI_ID, name, description_text, coord_lat, coord_long, category, contactInfo)
VALUES
(1, 'POI A', 'Big Amusement Park that uses ipad', 1.346815, 103.683388, 'Amusement Park',
'61234567'),

(2, 'POI B', 'Historic Museum', 1.344749, 103.681773, 'Museum', '62345678'),

(3, 'POI C', 'Largest Shopping Centre',1.385584,103.736900,'Shopping Centre', '63456789'),

(4, 'POI D', 'One of a Kind Resort', 1.362794,103.818281, 'Resort', '61234568'),

(5, 'POI E', 'First Class Airport', 1.431438,103.841785, 'Airport',      '61234569'),

('6','POI F', 'Trendiest Hawker Centre in Singapore', 1.355655,103.936294, 'Hawker Centre',
'63236796'),

('7', 'POI G', 'Fusion food Hawker Centre', 1.323529,103.677956, 'Hawker Centre',
'65647555'),

('8', 'POI H', 'Luxury Shopping Experience', 1.309800,103.768896, 'Shopping Centre',
'63237236'),

('9', 'POI I', 'Fashion Forward Complex', 1.372199,103.894358, 'Shopping Centre',
'63257896'),

('10', 'POI J', 'Modern State of the Art', 1.417298,103.756816, 'Museum', '63236696');


INSERT INTO dbo.POI_2
(coord_lat, coord_long, addressText)
VALUES

(1.346815,      103.683388,   '1 Sentosa Drive'),
(1.344749,      103.681773,   '2 Sentosa Drive'),
(1.385584,      103.736900,   '3 Sentosa Drive'),
(1.362794,      103.818281,   '4 Sentosa Drive'),
(1.431438,      103.841785,   '5 Sentosa Drive'),
(1.355655,      103.936294,   '6 Sentosa Drive'),
(1.323529,      103.677956,   '7 Sentosa Drive'),
(1.309800,      103.768896,   '8 Sentosa Drive'),
(1.372199,      103.894358,   '9 Sentosa Drive'),
(1.417298,      103.756816,   '10 Sentosa Drive');


INSERT INTO dbo.Comment
(msg_timeInstant, msg_senderID, comment_UserID, comment_timeInstant, content)
```

```
VALUES
('2019-04-06 12:00:00.000',  1,  1,  '2019-04-06 12:00:10.000',  'Cool'),
('2019-04-07 00:00:00.000',  1,  2,  '2019-04-07 00:00:10.000',  'Yeah!'),
('2019-04-08 00:00:00.000',  1,  3,  '2019-04-08 00:00:10.000'  ,  'Looks
fun!');
```

SQL Queries (From Appendix B)

1. *// Assume that we are executing the query for user with userID = 1*

```
SELECT      TOP 5 *
FROM        Message
WHERE       (senderID=1 AND receiverID=1) OR
            (senderID=1 AND receiverID IN (
                  SELECT UserID
                  FROM Contain
                  WHERE ownerID=1 AND listName='News Feed')) OR
            (receiverID=1 AND senderID IN (
                  SELECT UserID
                  FROM Contain
                  WHERE ownerID=1 AND listName='News Feed'))
ORDER BY    timeInstant DESC
```

2. *// Assume that we are executing the query for user with userID = 6*

```
SELECT      TOP 5 f2.ownerID
FROM        Friends f1, friends f2
WHERE       f1.friendID = f2.friendID AND f1.ownerID <> f2.ownerID AND
            f1.ownerID = 6 AND
            f2.ownerID NOT IN (SELECT friendID AS ownerID
                               FROM Friends
                               WHERE ownerID = 6)
GROUP BY f2.ownerID
ORDER BY count(*) desc
```

3.

```
SELECT      TOP 5 ownerID
FROM        friends
GROUP BY    ownerID
ORDER BY    count(friendID) desc
```

4. *// Assume that we are executing the query for user with userID = 2*

```
SELECT      TOP 5 name
FROM        (SELECT POI_ID, COUNT(CheckIn.userID) AS CountCheckIn
             FROM       (SELECT     friendID
                         FROM       Friends
                         WHERE ownerID=2)T1, CheckIn
             WHERE T1.friendID = CheckIn.UserID
             GROUP BY CheckIn.POI_ID)Temp, POI_1
WHERE       Temp.POI_ID = POI_1.POI_ID
ORDER BY    CountCheckIn DESC
```

5. *// Assume that we are executing the query with POI_ID = 2*

```
SELECT      POI_ID
FROM        (SELECT     coord_lat, coord_long
             FROM       POI_1
             WHERE      POI_ID =2) given,

            (SELECT     P1.POI_ID, coord_lat, coord_long
             FROM       dbo.POI_1 P1, Tip T1
             WHERE      CHARINDEX('ipad',P1.description_text) > 0 AND
                        CHARINDEX('excellent',T1.content) > 0 AND
                        P1.POI_ID = T1.POI_ID ) wordy

WHERE       SQRT(POWER((given.coord_lat-wordy.coord_lat) * 110.574, 2)
 + POWER(given.coord_long*111.32*COS(given.coord_lat) -
wordy.coord_long*111.32*COS(wordy.coord_lat), 2)) < 10
```

6.
```
SELECT      TOP 5 ID.UserID, coalesce(msg_cnt, 0)+coalesce(cmt_cnt,
0)+coalesce(chkIn_cnt, 0)+coalesce(tip_cnt, 0) AS activity

FROM        (SELECT     UserID
             FROM       UserTb) AS ID LEFT OUTER JOIN

            (SELECT     senderID, COUNT(*) AS msg_cnt
             FROM       Message
             GROUP BY   senderID) AS tb_1
             ON tb_1.senderID = ID.UserID       LEFT OUTER JOIN

            (SELECT     comment_UserID, count(*) AS cmt_cnt
             FROM       Comment
             GROUP BY   comment_UserID) AS tb_2
             ON ID.UserID = tb_2.comment_UserID LEFT OUTER JOIN
```

```
                (SELECT     UserID, COUNT(*) AS chkIn_cnt
                 FROM       CheckIn
                 GROUP BY   UserID) AS tb_3
                 ON ID.UserID = tb_3.UserID          LEFT OUTER JOIN

                (SELECT     UserID, COUNT(*) AS tip_cnt
                 FROM       Tip
                 GROUP BY   UserID) AS tb_4
                ON ID.UserID = tb_4.UserID

ORDER BY    activity        DESC
```

Additional Queries

1. **Suggest Friends to User based on Education: Given a user, suggest new friends to the user if they had gone to the same school at the same period (aka same classYear)**

```
SELECT DISTINCT    A2.UserID
FROM        Attended A1, Attended A2, Friends F
WHERE       A1.UserID = 1 AND
            A1.UserID <> A2.UserID AND
            A1.schoolName = A2.schoolName AND
            A1.classYear = A2.classYear AND
            A2.UserID NOT IN (  SELECT      friendID
                                FROM        Friends
                                WHERE       ownerID = 1    )
```

2. **Find all the friends of user k who have birthday in a particular month (e.g. month of January and user 1)**

```
SELECT DISTINCT   U.UserID, U.birthday
FROM              UserTb U, Friends F
WHERE             month(birthday) = 1 AND
                  friendID = UserID      AND
                  ownerID = 1;
```

Constraints and Triggers (From Appendix C)

1. CREATE TRIGGER ProtectDefaultList ON FriendList

```
AFTER DELETE
AS
IF EXISTS (SELECT  *
           FROM    deleted
           WHERE   deleted.listName = 'Friends' OR
                   deleted.listName=’News Feed’
)
BEGIN
ROLLBACK TRANSACTION
END
```

2.
```
CREATE TRIGGER autoAddMutualFriend ON Friends
AFTER INSERT
AS
BEGIN
INSERT INTO Friends
SELECT friendID, ownerID
FROM INSERTED
END

CREATE TRIGGER autoDeleteMutualFriend ON Friends
AFTER DELETE
AS
BEGIN
DELETE FROM Friends
WHERE EXISTS    (SELECT     *
                 FROM       deleted
                 WHERE      deleted.friendID = Friends.ownerID AND
                            deleted.ownerID = Friends.friendID)
END
```

3. **Addition of a New Friend**

```
CREATE TRIGGER addToFriendLists ON Friends
AFTER INSERT
AS
BEGIN
INSERT INTO Contain(ownerID, listName, UserID)
SELECT      ownerID, 'News Feed', friendID
FROM        inserted

INSERT INTO Contain(ownerID, listName, UserID)
SELECT      ownerID, 'Friends', friendID
FROM        inserted;
```

```
END
```

Deletion of a Friend

```
CREATE TRIGGER removeFromFriendLists ON Friends
AFTER DELETE
AS
BEGIN

DELETE FROM Contain
WHERE EXISTS (SELECT    *
                    FROM      deleted
                    WHERE deleted.ownerID=Contain.ownerID AND
                              deleted.friendID = Contain.UserID)
END
```

4.
```
CREATE TRIGGER trackTopUsers ON Message
AFTER UPDATE, DELETE, INSERT
AS

BEGIN
SELECT TOP 10 senderID, msg_cnt+COALESCE(cmt_cnt,0) AS cnt
FROM          (SELECT       senderID, COUNT(*) as msg_cnt
               FROM         Message
               GROUP        BY senderID) AS tb_1 LEFT OUTER JOIN

              (SELECT       comment_UserID, COUNT(*) AS cmt_cnt
               FROM         Comment
               GROUP BY     comment_UserID) AS tb_2

               ON           tb_1.senderID = tb_2.comment_UserID
ORDER BY      cnt      DESC
END

CREATE TRIGGER trackTopUsersFromComment ON Comment
AFTER UPDATE, DELETE, INSERT
AS
BEGIN

SELECT        TOP 10 senderID, msg_cnt+coalesce(cmt_cnt,0) AS cnt
FROM          (SELECT       senderID, count(*) AS msg_cnt
               FROM         Message
               GROUP        BYsenderID) as tb_1 LEFT OUTER JOIN
```

```
                (SELECT     comment_UserID,COUNT(*) AS cmt_cnt
                 FROM       Comment
                 GROUP BY   comment_UserID) AS tb_2

                 ON         tb_1.senderID = tb_2.comment_UserID
ORDER BY    cnt     DESC

END
```

5.
```
CREATE TRIGGER maxDailyCheckIn ON CheckIn
AFTER INSERT
As
IF EXISTS     (SELECT     CheckIn.UserID, CheckIn.chkIn_date, COUNT(*)
               FROM       CheckIn, inserted
               WHERE      CheckIn.UserID = inserted.UserID AND
                          CheckIn.chkIn_date = inserted.chkIn_date
               GROUP BY   CheckIn.UserID, CheckIn.chkIn_date
               HAVING     COUNT(*) >10)

BEGIN
ROLLBACK TRANSACTION
END
```

<u>Additional Trigger</u>

1. **A user must have checked in at a POI before he or she writes a tip on the POI.**

```
CREATE TRIGGER havChkedIn ON Tip
AFTER INSERT
AS
IF NOT EXISTS (SELECT     *
                 FROM     CheckIn,inserted
                 WHERE    checkIn.UserID=inserted.userID AND
                          CheckIn.POI_ID=inserted.UserID)
BEGIN
ROLLBACK TRANSACTION
END
```

2. **The database should automatically keep track of the top 5 most popular point of interest measured by number of check-ins**

```
CREATE TRIGGER trackTopPOI ON CheckIn
```

```
AFTER UPDATE, DELETE, INSERT
AS
BEGIN
SELECT id.POI_ID, POI_1.name
FROM            (SELECT TOP 5 POI_ID
                 FROM CheckIn
                 GROUP BY POI_ID
                 ORDER BY COUNT(*) DESC) AS id, POI_1

WHERE id.POI_ID = POI_1.POI_ID
END
```