NANYANG
TECHNOLOGICAL
UNIVERSITY
SINGAPORE

CZ2007 - Introduction to Databases
Lab 5 Report

Name and Signature	Individual Contributions (to justify the percentage contribution)	Percentage of contribution
Bachhas Nikita	Creation of Database and Formulation of Query Statements	25%
Delon Lim Long Ting	Creation of Database and Formulation of Query Statements	25%
Pang Kee Yang	Creation of Database and Formulation of Query Statements	25%
Tan Hui Zhan	Creation of Database and Formulation of Query Statements	25%

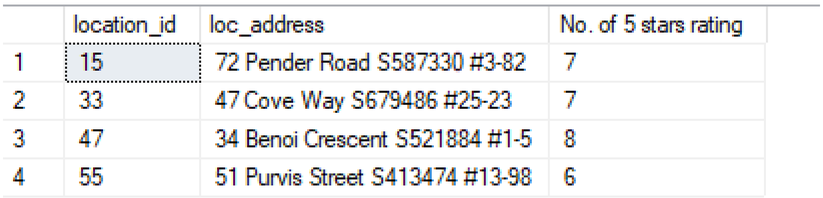
Table of contents

1st Query

1. Find the locations that receive at least 5 ratings of “5” in Dec 2020, and order them by their average ratings.

```
SELECT L.location_id, L.loc_address, COUNT(R1.rate) AS 'No.
of 5 stars rating'
FROM Locations AS L INNER JOIN
(
      SELECT location_id, rate, check_in_time
      FROM Ratings
      WHERE rate = 5 AND check_in_time BETWEEN '2020-12-01
00:00:00.000' AND '2020-12-31 23:59:59.000'
) AS R1
ON L.location_id = R1.location_id
GROUP BY L.location_id, L.loc_address
HAVING COUNT(R1.rate) > 5
order by AVG(CAST(R1.rate AS DECIMAL))
```

Results:



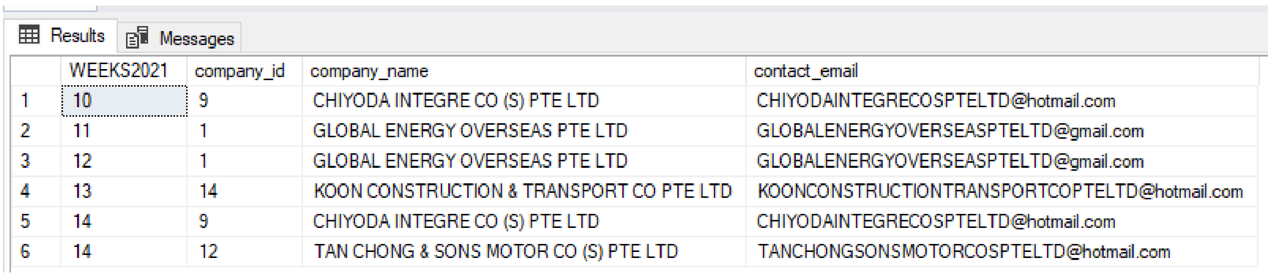
	location_id	loc_address	No. of 5 stars rating
1	15	72 Pender Road S587330 #3-82	7
2	33	47 Cove Way S679486 #25-23	7
3	47	34 Benoi Crescent S521884 #1-5	8
4	55	51 Purvis Street S413474 #13-98	6

2nd Query

2. Find the companies whose posts have received the most number of comments for each week of the past month.

```
SELECT AllCounts.WEEKS2021, COMP.company_id, COMP.company_name,
COMP.contact_email
FROM(
      SELECT DATEPART(WEEK, COMM.time_stamp) AS 'WEEKS2021',
CP.company_id, COUNT(COMM.msg_id) AS 'CommCOUNT'
            FROM Comments COMM JOIN ContactPersonsMessages CPM ON
COMM.msg_id = CPM.msg_id
                  JOIN ContactPersons CP ON CP.contact_person_id =
CPM.contact_person_id
            WHERE COMM.time_stamp BETWEEN (DATEADD(MM, -1,
GETDATE())) AND (GETDATE())
            GROUP BY DATEPART(WEEK, COMM.time_stamp), CP.company_id
      ) AS AllCounts,
(SELECT WEEKS2021, MAX(CommCount) AS 'MaxCount'
      FROM (
            SELECT DATEPART(WEEK, COMM.time_stamp) AS 'WEEKS2021',
CP.company_id, COUNT(COMM.msg_id) AS 'CommCOUNT'
            FROM Comments COMM JOIN ContactPersonsMessages CPM ON
COMM.msg_id = CPM.msg_id
                  JOIN ContactPersons CP ON CP.contact_person_id =
CPM.contact_person_id
            WHERE COMM.time_stamp BETWEEN (DATEADD(MM, -1,
GETDATE())) AND (GETDATE())
            GROUP BY DATEPART(WEEK, COMM.time_stamp), CP.company_id
      ) AS Test
      GROUP BY WEEKS2021
) AS MaxCounts,
Companies AS COMP
WHERE
      CommCOUNT = MaxCount
      AND AllCounts.WEEKS2021 = MaxCounts.WEEKS2021
      AND COMP.company_id = AllCounts.company_id
ORDER BY MaxCounts.WEEKS2021
```

Results:



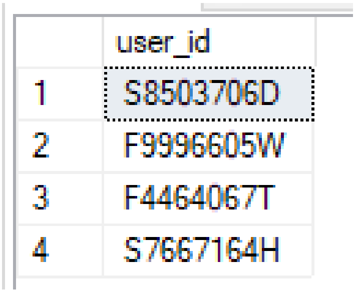
	WEEKS2021	company_id	company_name	contact_email
1	10	9	CHIYODA INTEGRE CO (S) PTE LTD	CHIYODAINTEGRECOSPTELTD@hotmail.com
2	11	1	GLOBAL ENERGY OVERSEAS PTE LTD	GLOBALENERGYOVERSEASPTELTD@gmail.com
3	12	1	GLOBAL ENERGY OVERSEAS PTE LTD	GLOBALENERGYOVERSEASPTELTD@gmail.com
4	13	14	KOON CONSTRUCTION & TRANSPORT CO PTE LTD	KOONCONSTRUCTIONTRANSPORTCOPTELTD@hotmail.com
5	14	9	CHIYODA INTEGRE CO (S) PTE LTD	CHIYODAINTEGRECOSPTELTD@hotmail.com
6	14	12	TAN CHONG & SONS MOTOR CO (S) PTE LTD	TANCHONGSONSMOTORCOSPTELTD@hotmail.com

3rd Query

3. Find the users who have checked in more than 10 locations every day in the last week.

```
SELECT C1.user_id
FROM CheckInOuts C1
join(
      SELECT check_in_time, user_id
      FROM CheckInOuts
      WHERE check_in_time BETWEEN (DATEADD(DD, -7, GETDATE())) and
(GETDATE())
 ) C2 on C2.check_in_time = C1.check_in_time
group by C1.user_id
having count(*) > 10
```

Results:



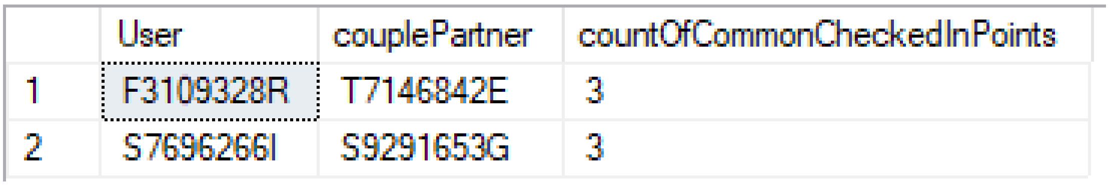
	user_id
1	S8503706D
2	F9996605W
3	F4464067T
4	S7667164H

4th Query

4. Find all the couples such that each couple has checked in at least 2 common locations on 1 Jan 2021.

```
SELECT * FROM (
      SELECT  COUP1.user_id as 'User' , COUP1.family_member_id as 
couplePartner , COUNT(COUP1.location_id) as 
countOfCommonCheckedInPoints
      FROM (SELECT DISTINCT FM.user_id, family_member_id, 
location_id -- All Couples checked in location in 1st jan
            FROM FamilyMembers FM, CheckInOuts CI --
            WHERE CI.user_id = FM.user_id
                  AND FM.rs_id = 14  -- Find all the couples
                  AND CI.check_in_time BETWEEN '01/01/2021 12:00:00 
AM' AND '01/01/2021 23:59:59 PM') as COUP1, -- ALL checkedIn Location
in 1st jan
            (SELECT DISTINCT FM.user_id, family_member_id, 
location_id -- duplicate table
            FROM FamilyMembers FM, CheckInOuts CI
            WHERE CI.user_id = FM.user_id
                  AND FM.rs_id = 14
                  AND CI.check_in_time BETWEEN '01/01/2021 12:00:00 
AM' AND '01/01/2021 23:59:59 PM') as COUP2
      WHERE COUP1.user_id <> COUP2.user_id
      AND COUP1.location_id = COUP2.location_id
      GROUP BY COUP1.user_id, COUP1.family_member_id) AS Z
WHERE countOfCommonCheckedInPoints >= 2
```

Results:



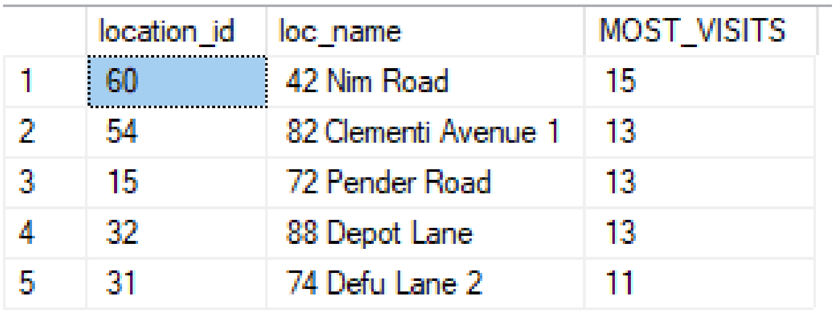
	User	couplePartner	countOfCommonCheckedInPoints
1	F3109328R	T7146842E	3
2	S7696266I	S9291653G	3

5th Query

5. Find 5 locations ids and their names that are checked in by the most number of users in the last 10 days.

```
SELECT TOP 5 L.location_id, L.loc_name, MOST_VISITS
FROM Locations L
join(
    SELECT C.location_id, COUNT(C.location_id) AS MOST_VISITS
    FROM CheckInOuts C
    WHERE check_in_time BETWEEN (DATEADD(DD, -10, GETDATE())) and
(GETDATE())
    GROUP BY C.location_id
)C on L.location_id = C.location_id
ORDER BY MOST_VISITS DESC
```

Results:



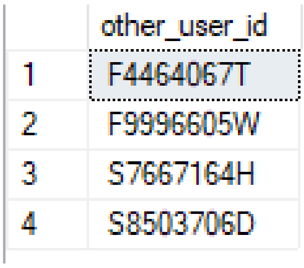
	location_id	loc_name	MOST_VISITS
1	60	42 Nim Road	15
2	54	82 Clementi Avenue 1	13
3	15	72 Pender Road	13
4	32	88 Depot Lane	13
5	31	74 Defu Lane 2	11

6th Query

6. Given a user, find the list of users that checked in the same locations with the user within 1 hour in the last week.

```
SELECT DISTINCT other_user_id
FROM (SELECT CheckInOuts.user_id as other_user_id,
CheckInOuts.check_in_time as
other_check_in_time,CheckInOuts.location_id, CILW.check_in_time
    FROM CheckInOuts JOIN (
             SELECT location_id, check_in_time
             FROM CheckInOuts
             WHERE
             CheckInOuts.check_in_time BETWEEN (DATEADD(DD, -7,
GETDATE())) AND (GETDATE())
             AND
             CheckInOuts.user_id = 'F9996605W' -- GIVEN USER
    ) AS CILW -- GivenUserCheckedInLastWk
    ON CILW.location_id = CheckInOuts.location_id
    WHERE
    CheckInOuts.check_in_time BETWEEN (DATEADD(HH, -1,
CILW.check_in_time))
    AND (DATEADD(HH, +1, CILW.check_in_time))
) AS OtherUserCheckedInSameLocationWithin1hr
```

Results:



	other_user_id
1	F4464067T
2	F9996605W
3	S7667164H
4	S8503706D

Interesting Queries

Design two queries that are not in the above list. They are evaluated based on the usefulness, complexity, and the interestingness. You are encouraged (not compulsory) to design trigger queries, from which you can learn something different and new.

Interesting Query 1 (include Trigger)

Steps for Interesting Query Idea

1. On update of positive swab test. (Trigger)
2. Find all the close contact of the userid
3. Insert all the new close contacts into the view table and display it out

Creating the Trigger Query:

1) Trigger for when a swab test result becomes positive (Trigger)

```
CREATE TRIGGER OnPositiveSwabResult
ON SCHEDULES
AFTER UPDATE
AS
BEGIN
      DECLARE @user_id VARCHAR(15), @test_result TINYINT

      SELECT @test_result = test_result from inserted
      SELECT @user_id = user_id from inserted

      IF @test_result = 1
      BEGIN
            SELECT family_member_id AS close_contact_user_id FROM
CloseFamilyMembers
            UNION
            SELECT user_id FROM CompanyColleagues
            UNION
            SELECT user_id FROM CloseContacts
      END
END
```

2) Find family members of the positive swab test user
3) Find positive swab test people who are checked in on the same place,same date for the last 14 days, and people come in close contact within the entire time of checked in and checked out time.

```
-- Positive user as view
CREATE VIEW PositiveUser AS
SELECT s.user_id,
FROM Schedules S
WHERE test_result = 1;

-- Family member of positive user
CREATE VIEW CloseFamilyMembers AS
SELECT family_member_id
FROM PositiveUser AS U, FamilyMembers AS F
WHERE F.user_id = U.user_id;
-- People who also work in same company
CREATE VIEW CompanyColleagues AS
SELECT DISTINCT U2.user_id
```

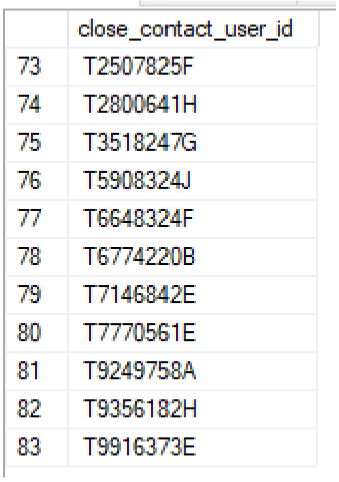
```
FROM (SELECT PositiveUser.user_id,Users.company_id FROM PositiveUser JOIN
Users ON PositiveUser.user_id = Users.user_id ) AS  U1,
Users AS U2
WHERE U1.user_id <> U2.user_id
   AND U1.company_id = U2.company_id

-- Other people who come in close contact based on location
-- R1: PositiveUser
-- R2: Places where positive user checked in for last 14 days
CREATE VIEW PositivePlaces AS
SELECT U.user_id,C.location_id, C.check_in_time, C.check_out_time
FROM PositiveUser AS U JOIN schedules AS S
ON U.user_id = S.user_id
JOIN CheckInOuts AS C
ON U.user_id = C.user_id
--WHERE S.scheduled_time >= DATEADD(day,-14, GETDATE());

-- R3:Other user who also checked in those places while positive user are
still checked in
CREATE VIEW CloseContacts AS
SELECT DISTINCT Users.user_id,CheckInOuts.location_id
FROM (Users JOIN CheckInOuts ON Users.user_id = CheckInOuts.user_id)
JOIN PositivePlaces ON CheckInOuts.location_id = PositivePlaces.location_id
WHERE CheckInOuts.check_in_time BETWEEN  PositivePlaces.check_in_time AND
PositivePlaces.check_out_time

--R4: Union of close contact person
SELECT family_member_id AS close_contact_user_id FROM CloseFamilyMembers
UNION
SELECT user_id FROM CompanyColleagues
UNION
SELECT user_id FROM CloseContacts
```

Results:



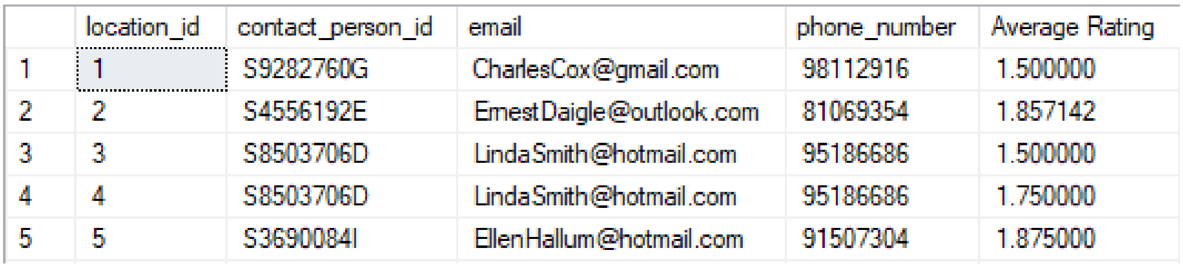
	close_contact_user_id
73	T2507825F
74	T2800641H
75	T3518247G
76	T5908324J
77	T6648324F
78	T6774220B
79	T7146842E
80	T7770561E
81	T9249758A
82	T9356182H
83	T9916373E

Interesting Query 2

Find the locations that are on average 2.5 and below, and the corresponding owner Companies' contact person and mailing address - discounting users who are biased reviewers - users who have left 80% of reviews that are less than 2 stars but they have to already leave at least 10 reviews.

```
SELECT
      L.location_id, CP.contact_person_id,
      U.email, U.phone_number, AVG(CAST(R.rate AS DECIMAL)) AS 'Average
Rating'
FROM Locations L, Ratings R, CompanyLocations CL, ContactPersons CP,
Companies C, Users U
WHERE L.location_id = R.location_id
      AND L.location_id = CL.location_id
      AND CP.company_id = CL.company_id
      AND C.company_id = CL.company_id
      AND U.user_id = CP.contact_person_id
GROUP BY L.location_id, CP.contact_person_id, U.email, U.phone_number
HAVING 2.5 > ( -- subquery #1: select locations with an average rating of <
2.5
      SELECT CAST(SUM(R1.rate) AS Decimal) / COUNT(R1.rate)
      FROM Ratings R1
      WHERE R1.location_id = L.location_id AND
            R1.user_id NOT IN
            ( -- Don't select ratings from troll reviewers
                  SELECT R2.user_id
                  FROM Ratings AS R2
                  WHERE R2.rate <= 2
                        AND (SELECT COUNT(*) FROM Ratings R1 WHERE
R1.user_id = R2.user_id) > 10
                  GROUP BY R2.user_id
                  HAVING (CAST(COUNT(R2.rate) AS DECIMAL)/(SELECT COUNT(*)
FROM Ratings R3 WHERE R3.user_id = R2.user_id) * 100) >= 80
            )
)
ORDER BY L.location_id
```

Results:



	location_id	contact_person_id	email	phone_number	Average Rating
1	1	S9282760G	CharlesCox@gmail.com	98112916	1.500000
2	2	S4556192E	ErnestDaigle@outlook.com	81069354	1.857142
3	3	S8503706D	LindaSmith@hotmail.com	95186686	1.500000
4	4	S8503706D	LindaSmith@hotmail.com	95186686	1.750000
5	5	S3690084I	EllenHallum@hotmail.com	91507304	1.875000

Interesting Trigger 1

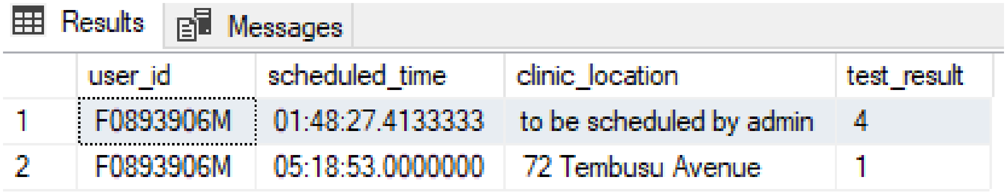
Triggering event: If a new tuple in TemperatureDeclaration is > 37.5
Action: schedule a new swab test for him/her - insert new tuple in Schedule

```
CREATE TRIGGER OnHighTempDeclaration
ON TemperatureDeclarations
AFTER INSERT
AS
BEGIN
      DECLARE @user_id VARCHAR(15), @temperature DECIMAL(3,1)
      SELECT @user_id = user_ID from inserted
      SELECT @temperature = temperature from inserted

      IF @temperature > 37.5
      BEGIN
            IF NOT EXISTS (SELECT * FROM Schedules S, INSERTED I WHERE
S.user_id = I.user_id AND S.test_result = 4)
            BEGIN
                  INSERT INTO Schedules VALUES(@user_id, GETDATE() ,'to be
scheduled by admin', 4)
            END
      END

      -- for DEBUG:
      SELECT * FROM Schedules WHERE user_id = @user_id
END
```

Results:



	user_id	scheduled_time	clinic_location	test_result
1	F0893906M	01:48:27.4133333	to be scheduled by admin	4
2	F0893906M	05:18:53.0000000	72 Tembusu Avenue	1