NANYANG TECHNOLOGICAL UNIVERSITY

SINGAPORE

Database Schema - Lab3

CZ2007 –INTRODUCTION TO DATABASES

Group 6

Name and Signature	**Individual Contributions** (To justify the percentage contribution)	**Percentage of Contribution (100% in total)**
Cui Chenling [illegible]	Identifying Relational Schema and FD, Checking BCNF/3NF, Report	25%
Luo Wenyu [illegible]	Identifying Relational Schema and FD, Checking BCNF/3NF, Report	25%
Bodipati Kiran [illegible]	Identifying Relational Schema and FD, Checking BCNF/3NF, Report	25%
Cao Shuwen [illegible]	Identifying Relational Schema and FD, Checking BCNF/3NF, Report	25%

List of Tables

PERSON(person_id, person_email, phone_number, name, hometown, birthday, sex, company_id)

ADMIN(person_id)

CONTACT_PERSON(person_id)

CATEGORY(category_name)

LOCATION_CD (locatoin_id, location_coordinates, location_name)

COMPANY_INFO (company_id, company_name, mailing_address, location_id)

DISCUSSION_FORUM(discussion_id, time, description, person_id)

SWAB_TEST(swab_test_id, time, result, clinic location, person_id)

TEMPERATURE(time_stamp,person_id, temperature)

COORDINATES(time_stamp, person_id, current_coordinate)

CHECK_IN_RECORD(checkin_time, person_id, location_id, checkout_time, rating, comment)

RELATE_TO(person_id, family_id, relationship)

CONTAIN(category_name, subcategory_name)

ABOUT(location_id, discussion_id)

ASSOCIATE(category_name, location_id)

COMMENT(person_id, discussion_id, time, text)

Explanation of the Derived Tables

PERSON(person_id, person_email, phone_number, name, hometown, birthday, sex, company_id)
Keys: {person_id}, {person_email}, {phone_number} (We assume that a phone_number can uniquely identify a person)
Primary key: person_id
FDs:
person_id → person_email, phone_number, name, hometown, birthday, sex, company_id
person_email→ person_id, phone_number, name, hometown, birthday, sex, company_id
phone_number→ person_id, phone_number, name, hometown, birthday, sex, company_id

The PERSON Table is in BCNF as the LHS of all the FDs contain a key.

ADMIN(person_id)
Keys: person_id
Primary key: person_id
ADMIN is a subclass of the PERSON entity set, and has the person ID as the only attribute. We are using the ER approach to derive the schema and the ADMIN is used elsewhere. Hence we use a table with a single attribute to denote the list of admins

CONTACT_PERSON(person_id)
Keys: person_id
Primary key: person_id
CONTACT_PERSON is a subclass of the PERSON entity set, and has the person ID as the only attribute. We are using the ER approach to derive the schema and the ADMIN is used elsewhere. Hence we use a table with a single attribute to denote the list of admins

CATEGORY(category_name)
Keys: category_name
Primary key: category_name
CATEGORY is in BCNF as there is no non-trivial FD.

LOCATION(location_id, location_coordinates, location_name, description)
Keys: {location_id},{ coordinates}
Primary key: location_id
FDs: location_id → location_coordinates, location_name, description
location_name → description
LOCATION is not in BCNF as the FD location_name → description is a violation of BCNF.
$\{location\_name\}^+$ = {locatoin_name, description}
LOCATION_INFO (location_name, description)
Keys: location_name
Primary key: location_name
FDs: location_name → description
LOCATION_CD (locatoin_id, location_coordinates, location_name)
Keys: location_id

Primary key: location_id
FDs: location_id → location_coordinates, location_name
LOCATION_INFO and LOCATION_CD are in BCNF as the LHS of all the FDs contains a key.

COMPANY(company_id, company_name, mailing_address, location_id, location coordinates)
Keys: company_id
Primary key: company_id
FDs: company_id → company_name, mailing_address, location_id
location_id → location_coordinates
COMPANY is not in BCNF as the FD location_id → location_coorinates is a violation to BCNF.
{location_id}$^+$ = {location_id, location_coordinates}
COMPANY_LOC (location_id, location_coordinates)
Keys: location_id
Primary key: location_id
FDs: location_id → location_coordinates
COMPANY_INFO (company_id, company_name, mailing_address, location_id)
Keys: company_id
Primary key: company_id
FDs: company_id → company_name, mailing_address, location_id
COMPANY_INFO and COMPANY_LOC are in BCNF as the LHS of all the FDs contains a key.

DISCUSSION_FORUM(discussion_forum_id, time, description, person_id)
Keys: discussion_forum_id, {person_id,time}
Primary key: discussion_forum_id
FDs: discussion_forum_id → time, description, person_id
DISCUSSION_FORUM is in BCNF as LHS of the only FD contains the key.

SWAB_TEST(swab_test_id, time, result, clinic location, person_id)
Keys: swab_test_id,{person_id,time}
Primary key: swab_test_id
FD(s): swab_test_id → time, result, clinic location, person_id
SWAB TEST is in BCNF as the LHS of the only FD contains the key.

TEMPERATURE(time_stamp,person_id, temperature)
Keys: {time_stamp, person_id}
Primary key: {time_stamp, person_id}
FD(s): time_stamp, person_id → temperature
TEMPERATURE is in BCNF as LHS of the only FD contains the key.

COORDINATES(time_stamp, person_id, current_coordinate)
Keys: {time_stamp, person_id}
Primary key: {time_stamp, person_id}
FD(s): time_stamp, person_id → current_coordinate

COORDINATES is in BCNF as LHS of the only FD contains the key.

CHECK_IN_RECORD(checkin_time, person_id, location_id, checkout_time, rating, comment)
Keys: {checkin_time, person_id, location_id},{checkout_time,person_id,location_id}
Primary key: {checkin_time, person_id, location_id}
FD(s): checkin_time, person_id, location_id → checkout_time, rating, comment
CHECK_IN_RECORD is in BCNF as the LHS of the only FD contains the key.

RELATE_TO(person_id, family_id, relationship)
Keys: {person_id, family_id}
Primary key: {person_id, family_id}
FD(s): person_id, family_id → relationship
RELATE_TO is in BCNF as the LHS of the only FD contains the key.

CONTAIN(category_name, subcategory_name)
Keys: {category_name, subcategory_name}
Primary key: {category_name, subcategory_name}
RELATE_TO is in BCNF as there is no non-trivial FD.

ABOUT(location_id, discussion_id)
Keys: {location_id, discussion_id}
Primary key: {location_id, discussion_id}
FD(s): location_id, discussion_id → location_id, discussion_id (Trivial)
ABOUT is in BCNF as there is no non-trivial FD.

ASSOCIATE(category_name, location_id)
Keys: {category_name, location_id}
Primary key: {category_name, discussion_id}
FD(s): category_name, discussion_id → category_name, discussion_id (Trivial)
ASSOCIATE is in BCNF as there is no non-trivial FD.

COMMENT(person_id, discussion_id, time, text)
Keys: {person_id, discussion_id,time}
Primary key: {person_id, discussion_id,time}
FD(s): person_id, discussion_id,time → text
COMMENT is in BCNF as LHS of the FD contains the key.

SQL Queires

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

```
CREATE VIEW loc_rtlist AS
SELECT cr.location_id, location_name
FROM CHECKIN_RECORD AS cr,LOCATION_CD AS lc
WHERE cr.rating=5 AND cr.location_id = lc.location_id
GROUP BY cr.location_id,location_name
HAVING COUNT(*) >= 5

SELECT cr.location_id, location_name, AVG(cr.rating) AS ave_rating
FROM CHECKIN_RECORD AS cr JOIN Loc_rtlist AS ll
ON ll.location_id = cr.location_id
GROUP BY cr.location_id, ll.location_name
ORDER BY ave_rating
```

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

```
CREATE VIEW company_List_wk1 AS
SELECT ci.company_id, ci.company_name, COUNT(c.text) as commentCount
FROM COMPANY_INFO AS ci, COMMENT AS c, ABOUT AS a
WHERE c.discussion_id = a.discussion_id AND a.location_id = ci.location_id AND
c.time< ='2021-03-07 23:59' AND c.time>= '2021-03-01 00:00'
GROUP BY ci.company_id, ci.company_name

CREATE VIEW result_wk1 AS
SELECT company_id, company_name, commentCount
FROM company_List_wk1
WHERE commentCount in (SELECT MAX(commentCount)
                                             FROM company_List_wk1)

CREATE VIEW company_List_wk2 AS
SELECT ci.company_id, ci.company_name, COUNT(c.text) as commentCount
FROM COMPANY_INFO AS ci, COMMENT AS c, ABOUT AS a
WHERE c.discussion_id = a.discussion_id AND a.location_id = ci.location_id AND
c.time< ='2021-03-014 23:59' AND c.time>= '2021-03-08 00:00'
GROUP BY ci.company_id, ci.company_name

CREATE VIEW result_wk2 AS
SELECT company_id, company_name, commentCount
FROM company_List_wk2
WHERE commentCount in (SELECT MAX(commentCount)
                                             FROM company_List_wk2)

CREATE VIEW company_List_wk3 AS
```

```
SELECT ci.company_id, ci.company_name, COUNT(c.text) as commentCount
FROM COMPANY_INFO AS ci, COMMENT AS c, ABOUT AS a
WHERE c.discussion_id = a.discussion_id AND a.location_id = ci.location_id AND
c.time< ='2021-03-21 23:59' AND c.time>= '2021-03-15 00:00'
GROUP BY ci.company_id, ci.company_name

CREATE VIEW result_wk3 AS
SELECT company_id, company_name, commentCount
FROM company_List_wk3
WHERE commentCount in (SELECT MAX(commentCount)
                                        FROM company_List_wk3)

CREATE VIEW company_List_wk4 AS
SELECT ci.company_id, ci.company_name, COUNT(c.text) as commentCount
FROM COMPANY_INFO AS ci, COMMENT AS c, ABOUT AS a
WHERE c.discussion_id = a.discussion_id AND a.location_id = ci.location_id AND
c.time< ='2021-03-28 23:59' AND c.time>= '2021-03-22 00:00'
GROUP BY ci.company_id, ci.company_name

CREATE VIEW result_wk4 AS
SELECT company_id, company_name, commentCount
FROM company_List_wk4
WHERE commentCount in (SELECT MAX(commentCount)
                                        FROM company_List_wk4)

CREATE VIEW company_List_wk5 AS
SELECT ci.company_id, ci.company_name, COUNT(c.text) as commentCount
FROM COMPANY_INFO AS ci, COMMENT AS c, ABOUT AS a
WHERE c.discussion_id = a.discussion_id AND a.location_id = ci.location_id AND
c.time< ='2021-03-31 23:59' AND c.time>= '2021-03-29 00:00'
GROUP BY ci.company_id, ci.company_name

CREATE VIEW result_wk5 AS
SELECT company_id, company_name, commentCount
FROM company_List_wk5
WHERE commentCount in (SELECT MAX(commentCount)
                                        FROM company_List_wk5)

SELECT company_id, company_name, commentCount FROM result_wk1
SELECT company_id, company_name, commentCount FROM result_wk2
SELECT company_id, company_name, commentCount FROM result_wk3
SELECT company_id, company_name, commentCount FROM result_wk4
SELECT company_id, company_name, commentCount FROM result_wk5
```

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

```
CREATE VIEW LASTWeek AS (SELECT CAST(checkin_time AS DATE) as DayCheckin,
location_id, person_id, count(person_id) as NUM_TIMES FROM CHECKIN_RECORD
WHERE checkin_time< ='2021-03-31 23:59:59' AND checkin_time>= '2021-03-25 00:00:00'
GROUP BY CAST(checkin_time AS DATE), location_id, person_id)

create view unique_Loc_Each_Day as(select DayCheckin, person_id, count(person_id) as
NumLocations from LastWeek
group by DayCheckin,person_id)

select name, person_id from person
where person_id in (select u.person_id as ID from unique_Loc_Each_Day u
where NumLocations>=10
Group by u.person_id
having count(*)=7)
```

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

```
CREATE VIEW relate_info AS(
SELECT person_id AS person1, location_id
FROM CHECKIN_RECORD as R1
WHERE R1.checkin_time<='2021-01-01 23:59:00' AND R1.checkin_time>='2021-01-01
00:00:00')


CREATE VIEW coupleINfo AS(SELECT R2.person1, R2.person2, R.location_id
FROM relate_info as R, relate_info as R1,RELATE_TO as R2
WHERE R.person1 = R2.person1 AND R1.person1 = R2.person2 AND R2.relationship = 'spouse'
AND R.location_id = R1.location_id
)

SELECT person1,person2,COUNT(location_id) AS location_CNT
FROM coupleINfo
WHERE person1<person2
GROUP BY person1,person2
HAVING COUNT(location_id) >= 2
```

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

```
CREATE VIEW NUmofvisitors AS(
SELECT location_id, COUNT(person_id) AS user_Count
FROM dbo.CHECKIN_RECORD
WHERE checkin_time< ='2021-03-31 23:59:59' AND checkin_time>= '2021-03-22 00:00:00'
GROUP BY location_id)

SELECT TOP 5 Loc.location_id, Loc.location_name
```

```
FROM dbo.NUmofvisitors AS Num, dbo.LOCATION_CD as Loc
WHERE Num.location_id = Loc.location_id
ORDER BY user_Count DESC
```

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.

```
CREATE VIEW Loc_list AS
SELECT DISTINCT p.person_id, cir.checkin_time, cir.location_id
FROM PERSON AS p, LOCATION_CD AS lcd, CHECKIN_RECORD AS cir
WHERE cir.checkin_time< ='2021-03-31 23:59:59' AND cir.checkin_time>= '2021-03-25 00:00:00'

CREATE VIEW Result_List AS
SELECT ll.person_id AS given_user_id, cir2.person_id AS other_users_id, ll.location_id
FROM CHECKIN_RECORD AS cir2 JOIN Loc_list AS ll
        on ll.location_id = cir2.location_id AND ll.person_id<>cir2.person_id
WHERE datediff(mi, cir2.checkin_time, ll.checkin_time) <= 60 AND datediff(mi, cir2.checkin_time,
ll.checkin_time) >= -60

SELECT DISTINCT given_user_id, other_users_id
FROM Result_List
ORDER BY GIVEN_USER_ID
```

7.Find the persons who have been tested positive in the past 1 week in swab test whose family members having temperature >37.6 degree at least once in the past 2 weeks

```
DROP VIEW posi_person
CREATE VIEW posi_person AS(
SELECT DISTINCT person_id
FROM SWAB_TEST
WHERE time>='2021-03-25 00:00'AND time<='2021-03-31 23:59' AND result= 'Positive')

drop view family_name

CREATE VIEW family_name AS(
SELECT P.person_id,  R.person2 AS family_member_id
FROM posi_person AS P, RELATE_TO AS R
WHERE P.person_id = R.person1 )


SELECT F.person_id, F.family_member_id, T.temperature
FROM TEMPERATURE AS T, family_name AS F
```

```
WHERE T.person_id = F.family_member_id AND T.time_stamp>='2021-03-18 00:00' AND
T.time_stamp<='2021-03-31 23:59' AND T.temperature>=37.6
```

LOCATION_CD (locatoin_id, location_coordinates, location_name)
CHECK_IN_RECORD(checkin_time, person_id, location_id, checkout_time, rating, comment)
COORDINATES(time_stamp, person_id, current_coordinate)

8.Write a Trigger such that if a person checks into a new location, he checks out of his most recent old location

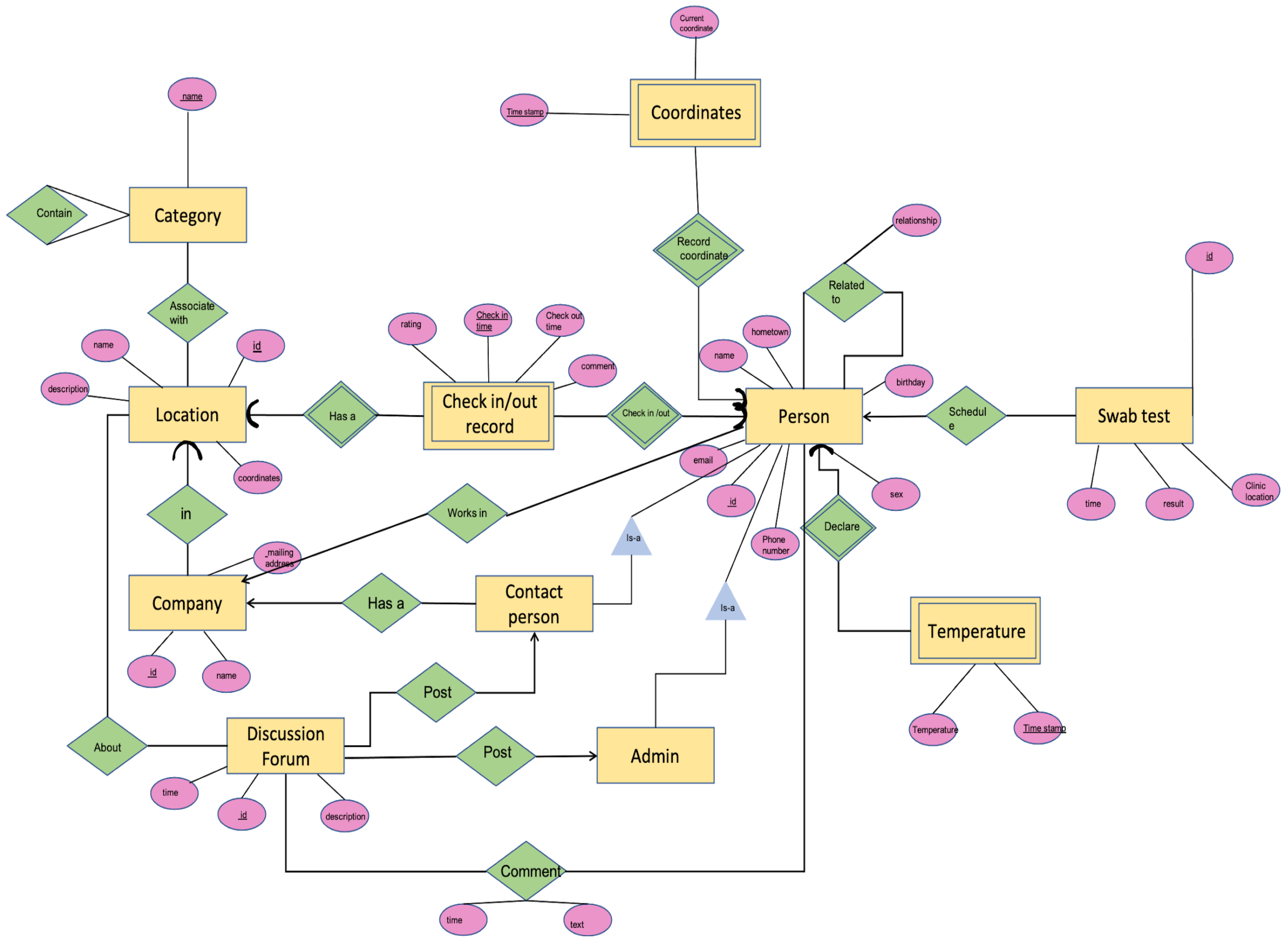
```
CREATE TRIGGER outTrig
ON CHECKIN_RECORD
AFTER INSERT
AS
BEGIN

UPDATE CHECKIN_RECORD
SET CHECKIN_RECORD.checkout_time = inserted.checkin_time
from inserted
WHERE CHECKIN_RECORD.person_id = inserted.person_id
and CHECKIN_RECORD.checkin_time = (SELECT TOP 1 C.checkin_time
                                   FROM CHECKIN_RECORD C, inserted
                                   WHERE C.person_id = inserted.person_id AND
C.checkin_time< inserted.checkin_time
                                   ORDER BY C.checkin_time desc)

END
```

Schema

```
CREATE TABLE PERSON (
        person_id INT NOT NULL,
        person_email VARCHAR(30),
        phone_number VARCHAR(30),
        name VARCHAR(30),
        hometown VARCHAR(30),
        birthday DATETIME,
        sex VARCHAR(30),
        company_id INT,
        PRIMARY KEY (person_id));

CREATE TABLE ADMIN (
        person_id INT NOT NULL,
        PRIMARY KEY (person_id),
        FOREIGN KEY (person_id) REFERENCES PERSON (person_id));

CREATE TABLE CONTACT_PERSON(
        person_id INT NOT NULL,
        PRIMARY KEY (person_id),
        FOREIGN KEY (person_id) REFERENCES PERSON(person_id));

CREATE TABLE CATEGORY (
        category_name VARCHAR(30)
        PRIMARY KEY (category_name));

CREATE LOCATION_CD(
        location_id INT NOT NULL,
        location_name NVARCHAR(32),
        coordinates_X DECIMAL(18,3),
        coordinates Y DECIMAL(18,3).
        PRIMARY KEY(location_id));


CREATE TABLE COMPANY_INFO (
        company_id INT NOT NULL,
        company_name VARCHAR(30),
        mailing_address VARCHAR(100),
        location_id INT NOT NULL,
        PRIMARY KEY (company_id),
        FOREIGN KEY (location_id) REFERENCES LOCATION_CD (location_id));

CREATE TABLE DISCUSSION_FORUM (
        discussion_id INT NOT NULL,
        time DATETIME ,
        description VARCHAR(1000),
        person_id int NOT NULL,
        PRIMARY KEY(discussion_id, time,person_id)
        FOREIGN KEY (person_id) REFERENCES PERSON (person_id));

CREATE TABLE SWAB_TEST (
        swab_test_id INT NOT NULL,
        time DATETIME,
        result VARCHAR(30),
        clinic_location VARCHAR(100),
        person_id INT NOT NULL,
        PRIMARY KEY (swab_test_id),
        FOREIGN KEY (person_id) REFERENCES PERSON (person_id));

CREATE TABLE TEMPERATURE (
        time_stamp DATETIME,
        person_id INT NOT NULL,
        temperature DECIMAL (5,3),
        PRIMARY KEY (time_stamp, person_id),
        FOREIGN KEY (person_id) REFERENCES PERSON (person_id));

CREATE TABLE COORDINATES (
        time_stamp DATETIME,
        person_id INT NOT NULL,
        x DECIMAL (5,3),
        y DECIMAL (5,3),
        PRIMARY KEY (time_stamp, person_id),
        FOREIGN KEY (person_id) REFERENCES PERSON (person_id));

CREATE TABLE CHECK_IN_RECORD (
        checkin_time DATETIME,
        checkout_time DATETIME,
        person_id INT NOT NULL,
        location_id INT NOT NULL,
        rating INT,
        comment VARCHAR(1000),
        PRIMARY KEY (checkin_time, person_id, location_id),
        FOREIGN KEY (person_id) REFERENCES PERSON (person_id),
        FOREIGN KEY (location_id) REFERENCES LOCATION_CD (location_id));

CREATE TABLE RELATE_TO(
        person_id INT NOT NULL,
        family_id INT NOT NULL,
        relationship NVARCHAR(8) NOT NULL,
        PRIMARY KEY(person_id, family_id),
        FOREIGN KEY(person_id) REFERENCES PERSON(person_id));

CREATE TABLE CONTAIN(
        category_name VARCHAR(30) NOT NULL,
        subcategory_name VARCHAR(30) NOT NULL,
        PRIMARY KEY(category_name, subcategory_name));


CREATE TABLE ABOUT (
        location_id INT NOT NULL,
        discussion_id INT NOT NULL,
        PRIMARY KEY (location_id, discussion_id),
        FOREIGN KEY(location_id) REFERENCES LOCATION_CD(location_id)
```

```
	FOREIGN KEY(discussion_id) REFERENCES DISCUSSION_FORUM(discussion_id));

CREATE TABLE ASSOCIATE (
	category_name VARCHAR(30),
	location_id INT NOT NULL,
	PRIMARY KEY (category_name, location_id),
	FOREIGN KEY(location_id) REFERENCES LOCATION_CD(location_id),
	FOREIGN KEY(category_name) REFERENCES CATEGORY(category_name)

CREATE TABLE COMMENT(
	person_id INT NOT NULL,
	discussion_id INT NOT NULL,
	time DATETIME NOT NULL,
	text NVARCHAR(512) NOT NULL,
	PRIMARY KEY(person_id, discussion_id, time)
	FOREIGN KEY (person_id) REFERENCES PERSON(person_id),
	FOREIGN KEY (discussion_id) REFERENCES DISCUSSION_FORUM(discussion_id));
```

Appendix


Current coordinate
name
Coordinates
Time stamp
Category
Contain
Record coordinate
relationship
id
Related to
Associate with
rating
Check in time
Check out time
name
id
hometown
comment
name
birthday
description
Location
Has a
Check in/out record
Check in /out
Person
Schedule
Swab test
email
coordinates
id
sex
time
result
Clinic location
in
Works in
Declare
Is-a
Phone number
mailing address
Is-a
Company
Has a
Contact person
Temperature
id
name
Post
Temperature
Time stamp
About
Discussion Forum
Post
Admin
time
id
description
Comment
time
text