FINAL PROJECT REPORT

Personal Health Management System

BY

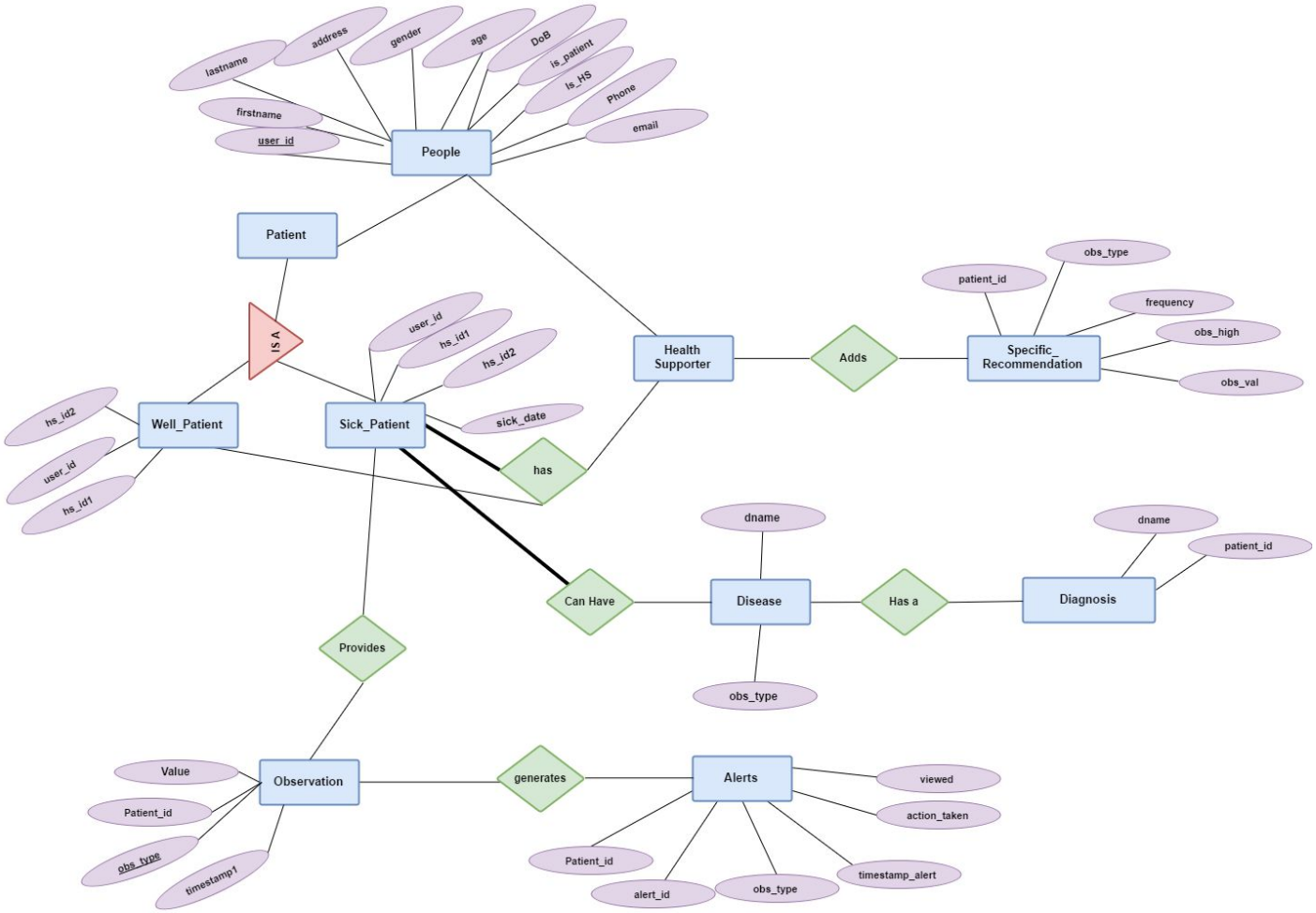
Ankur kataria (akatari2)
Falak ravani (fvravani)
Parin sanghvi (prsangha)
Shalini Sejwani (smsejwan)

Guide: Kemafor Ogan

Computer Science Department
North Carolina State University, Raleigh

Table of Contents

ER Diagram


lastname
address
gender
age
DoB
is_patient
Is_HS
Phone
email
firstname
user_id
People
Patient
IS A
Well_Patient
hs_id2
user_id
hs_id1
Sick_Patient
user_id
hs_id1
hs_id2
sick_date
has
Health Supporter
Adds
Specific_ Recommendation
patient_id
obs_type
frequency
obs_high
obs_val
Can Have
Disease
dname
obs_type
Has a
Diagnosis
dname
patient_id
Provides
Observation
Value
Patient_id
obs_type
timestamp1
generates
Alerts
viewed
action_taken
Patient_id
alert_id
obs_type
timestamp_alert

Constraints:

Application Constraints

1. Sick patients have at least 1 disease.
2. A patient may have up to two health supporters.
3. No access to patient info before auth date.
4. Patients inherit the observation recommendations for that class of patients.(for a particular disease of class)
5. For sick patient's, observation requirements are mandatory.
6. Weight should be numeric type and it should have upper and lower limits.
7. Blood pressure has two components - systolic, distolic and both have numeric types and each component has upper and lower limits.
8. Oxygen saturation of SPO2 level is numeric type and each component has upper and lower limits.
9. Pain is in the range of 1 to 10.
10. Mood is in the range of 1 to 5 where 1 is sad and 5 is happy.
11. Alerts are of two types
    a. Outside the limit.
    b. Low activity
12. Alerts can be cleared by patients and health supporters.
13. A sick patient has at least one health supporter
14. A sick patient should always have one primary health supporter.

Functional Dependencies

1. **People**

   U_id → name, address, DoB, gender, age, contact_info, is_paitent,, Is_HS
2. **Patient**

   U_id → hs_id1, hs_id2, type, is_sick
3. **Health Supporter**

   U_id → auth_date, patient_id
4. **Specific_recommendation**

   U_id, obs_type, freq → obs_high, obs_low
5. **Diease_recommendation**

   D_name, obs_type → freq., upper, lower
6. **Observation**

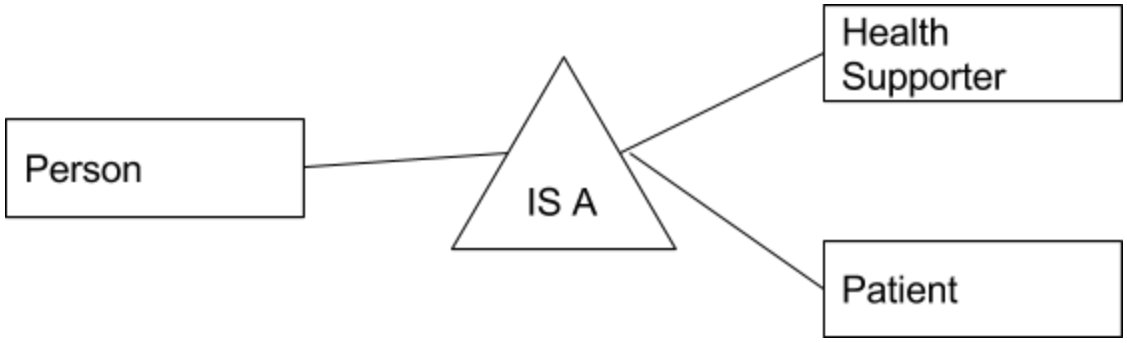
   U_id, obs_type, timestamp → value, obs_time
7. **Alerts**

   Patient_id, alert_id → type, action_taken, viewed

Relational Model:

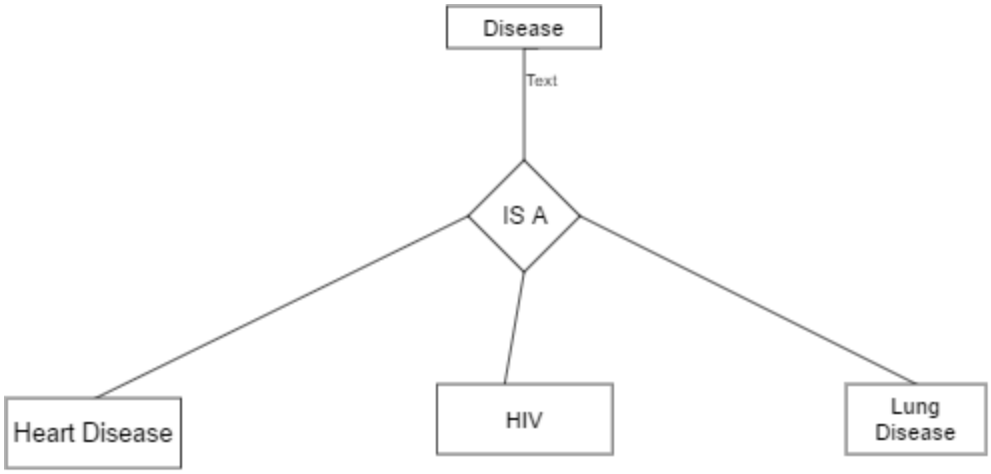
A List of Relationships

Person - IS A - Health Supporter and Patient (Binary)

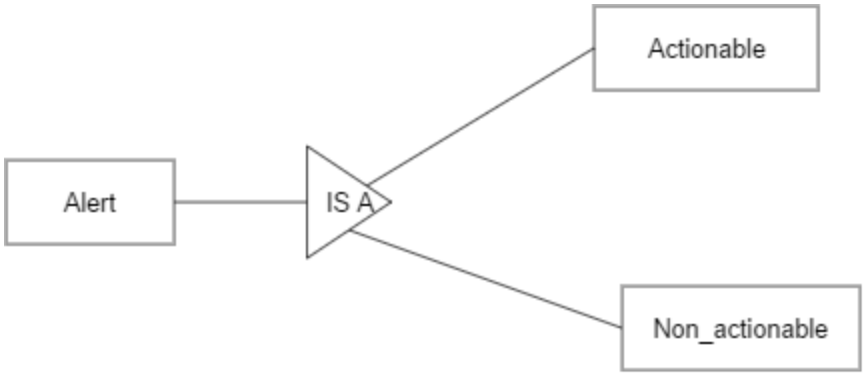


Patient - gets - Alerts (Binary)

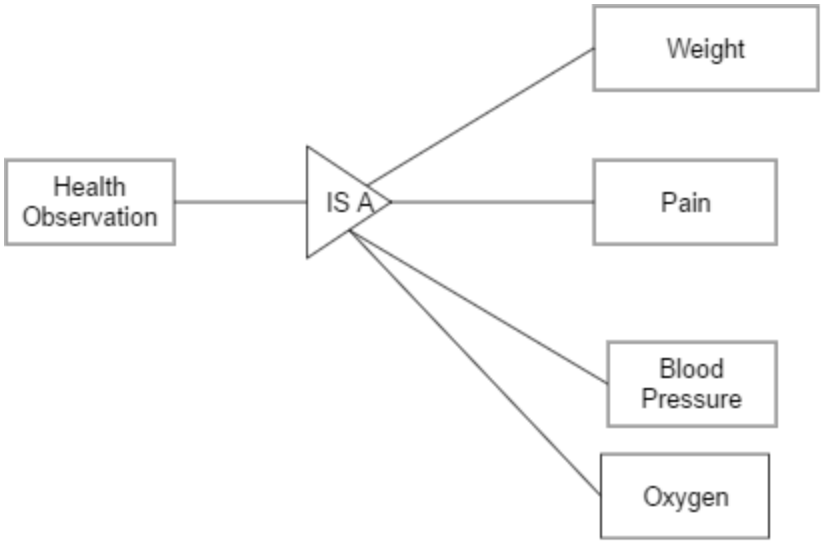
Disease - IS A - Heart_Disease, Lung_Disease and HIV (Quaternary means 4-ary.)



Alerts - IS A - Actionable and Non_Actionable



Health-Observation - IS A - Weight, Blood pressure, Oxygen-Saturation and Pain



Well_Patient - HAS A - Health_supporter (Binary)

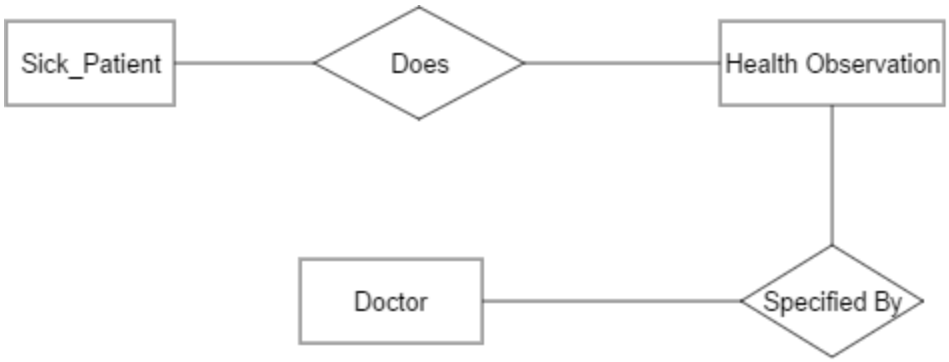
Sick_Patient - HAS A - Health_supporter (binary)

Sick_Patient - IS diagnosed - Disease (binary)

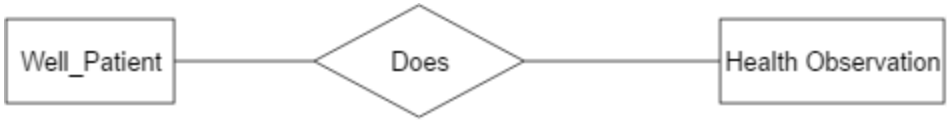
Patient - gets - Alerts (binary)

Health_Supporter --Updates-- Health_Recommendation

Sick_Patient --needs-- Health Observations --specified by-- Doctor (Ternary)



Well_Patient --does-- Health Observations (Binary)



Triggers

1. **Delete Low Activity alert when the observations are entered in the observation table after the alert has been issued.**

```
create or replace TRIGGER OBS_DELETE_LOWACT
BEFORE INSERT OR UPDATE of obs_time, patient_id, obs_type ON
OBSERVATION
REFERENCING OLD AS PREV NEW AS RECENT

FOR EACH ROW
```

```
DECLARE
alerid number;

BEGIN
  alerid := null;
  select alert_id into alerid from alerts where patient_id = :recent.patient_id and
obs_type= :recent.obs_type and ALERT_TYPE='Low_activity';

  if alerid is not null
  then
  update alerts set action_taken='y' where patient_id = :recent.patient_id and
obs_type= :recent.obs_type;
End if;
END;
```

2. Limit the health supporters to two.

```
create or replace TRIGGER HS_LIMIT
BEFORE INSERT OR UPDATE OF PATIENT_ID,USER_ID ON
HEALTH_SUPPORTER
REFERENCING OLD AS PREV NEW AS RECENT
FOR EACH ROW

DECLARE
v_num NUMBER(10);
CURSOR hs_trigger
IS SELECT COUNT(USER_ID) FROM HEALTH_SUPPORTER WHERE
Patient_id = :recent.Patient_ID;

BEGIN
OPEN hs_trigger;
FETCH hs_trigger INTO v_num;
CLOSE hs_trigger;
IF
 :recent.Patient_id = :recent.user_id
then
  Raise_application_error(-20202,'patient can not be his own health supporter');
END IF;
IF
  v_num =2
then
  RAISE_APPLICATION_ERROR(-20020, 'Already has maximum allowed Health
Supporters');
END IF;
```

```
END;
```

3. **If a patient is Sick ensure that he has a entry in the Disease table.**

```
create or replace TRIGGER IF_SICK_ADD_DISEASE
BEFORE INSERT OR UPDATE OF DNAME,PATIENT_ID ON DIAGNOSIS
REFERENCING OLD AS PREV NEW AS RECENT

FOR EACH ROW

DECLARE
sick NUMBER(10);
check_sick NUMBER(10);
hs_id1 varchar2(4);
hs_id2 varchar2(4);
type hsidtab is table of health_supporter.user_id%type;
hsids hsidtab;

CURSOR sick_trigger
IS SELECT COUNT(user_id) FROM HEALTH_SUPPORTER WHERE Patient_id
= :recent.Patient_ID;

CURSOR check_sick_trigger
IS SELECT COUNT(user_id) FROM Sick_Patient WHERE user_id =
:recent.Patient_ID;

CURSOR Hs_trigger
IS SELECT user_id FROM HEALTH_SUPPORTER WHERE Patient_id =
:recent.Patient_ID;

BEGIN
OPEN sick_trigger;
FETCH sick_trigger INTO sick;
CLOSE sick_trigger;

OPEN check_sick_trigger;
FETCH check_sick_trigger INTO check_sick;
CLOSE check_sick_trigger;

OPEN Hs_trigger;
FETCH Hs_trigger BULK COLLECT INTO hsids;
CLOSE Hs_trigger;
```

```
IF
  sick =0
then
  Raise_application_error(-20202,'patient has no health supporter, add health
supporter');
END IF;
IF
  sick =1 OR sick = 2
then
  IF check_sick =0
    then
    DELETE FROM WELL_PATIENT WHERE user_id =:recent.patient_id;
    UPDATE PATIENT set is_sick ='y' WHERE user_id =:recent.patient_id;
    INSERT INTO SICK_PATIENT VALUES(:recent.patient_id, hsids(1), hsids(2),
sysdate );
  End IF;
END IF;
END;
```

4. **Enter a new patient in Well patient table whenever a new entry is added in people table with is_patient = yes**

```
create or replace TRIGGER NEW_PEOPLE_PATIENT
After INSERT OR UPDATE OF IS_PATIENT,USER_ID ON PEOPLE
REFERENCING OLD AS PREV NEW AS RECENT
For each row
BEGIN
if :recent.IS_PATIENT = 'y'
then
insert into Patient values(:recent.user_id,'N');
insert into Well_patient values(:recent.user_id,null,null);
End if;

END;
```

5. **If the entry in the observation table is less than or greater than the allowed limit than the patient should get an outside the limit alert.**

```
create or replace TRIGGER OUTSIDE_LIMIT_ALERT
BEFORE INSERT OR UPDATE OF obs_value, patient_id, obs_type on
Observation
REFERENCING OLD AS PREV NEW AS RECENT
```

```
FOR EACH ROW

DECLARE
upper_val number(10);
lower_val number(10);


BEGIN
Select obs_high into upper_val from SpecificRec where patient_id
=:recent.patient_id and obs_type = :recent.obs_type;
Select obs_low into lower_val from SpecificRec where patient_id
=:recent.patient_id and obs_type = :recent.obs_type;

IF lower_val is not null
THEN
If :recent.obs_value > upper_val or :recent.obs_value < lower_val
then
insert into alerts values(:recent.patient_id, alert_sequence.nextval ,
'Outside_Limit',:recent.obs_type,sysdate, 'N', 'N');
End if;
ELSE
If :recent.obs_value > upper_val
then
insert into alerts values(:recent.patient_id, alert_sequence.nextval,
'Outside_Limit',:recent.obs_type, sysdate, 'N', 'N');
End if;

END IF;
END;
```

6. **If in the Sick_patient table the is_sick changes from ‘y’ to ‘n’ then we should delete the patient from sick_patient and insert into well_patient table**

```
create or replace TRIGGER SICKTOWELL
After DELETE ON DIAGNOSIS
REFERENCING OLD AS PREV NEW AS RECENT
for each row
Declare
number1 number;
hsid1 varchar2(10);
hsid2 varchar2(10);

BEGIN
select count(*) into number1 from Diagnosis where patient_id = :prev.patient_id;
```

```
if number1 = 0
then
select hs_id1 into hsid1 from sick_patient where USER_ID = :prev.patient_id;
select hs_id2 into hsid2 from sick_patient where USER_ID = :prev.patient_id;
insert into well_patient values(:prev.patient_id, hsid1,hsid2);
update patient set is_sick='n' where USER_ID = :prev.patient_id;
delete from sick_patient where USER_ID = :prev.patient_id;
End if;
END;
```

Sequence:

```
CREATE SEQUENCE  "UNITYID"."ALERT_SEQUENCE"  MINVALUE 1 MAXVALUE
10000 INCREMENT BY 1 START WITH 101 CACHE 20 NOORDER  NOCYCLE ;
```

Procedures:

1. **If a patient has no specific recommendation then the default recommendations of the disease should be updated for him in the recommendation table.**

```
create or replace PROCEDURE CHECK_REC_PATIENT AS
TYPE isick is Table of char(1);
TYPE patient_id1 IS TABLE OF varchar2(4);
value1 patient_id1;
value2 isick;
ck number;
dname1 varchar2(10);
Begin
  execute immediate 'SELECT user_id from patient' bulk collect into value1;
  execute immediate 'SELECT is_sick from patient' bulk collect into value2;

  For indx in 1 .. value1.count
  Loop
  Select count(*) into ck from Specificrec Where PATIENT_id =  value1(indx);
    if ck =0
    then
      if value2(indx) = 'n'
      then
      /* Insert well patient values*/
      Insert into SpecificRec values (value1(indx), 'weight', '7', '200', '120');
```

```
Insert into SpecificRec values (value1(indx), 'BP_s', null, null, null);

Insert into SpecificRec values (value1(indx), 'BP_d', null, null, null);

Insert into SpecificRec values (value1(indx), 'Oxygen', null, null, null);

Insert into SpecificRec values (value1(indx), 'Pain', null, null, null);

Insert into SpecificRec values (value1(indx), 'Mood', null, null, null);

Insert into SpecificRec values (value1(indx), 'Temperature', null, null, null);

DBMS_OUTPUT.put_line('wel patient values insert');
end if;

if value2(indx) = 'y'
then
  select dname into dname1 from diagnosis where patient_id = value1(indx);
  if dname1 = 'Heart'
  then
    Insert into SpecificRec values (value1(indx), 'weight', '7', '200', '120');

    Insert into SpecificRec values (value1(indx), 'BP_s', '1', '159', '140');

    Insert into SpecificRec values (value1(indx), 'BP_d','1', '99', '90');

    Insert into SpecificRec values (value1(indx), 'Oxygen', null, null , null);

    Insert into SpecificRec values (value1(indx), 'Pain', null, null, null);

    Insert into SpecificRec values (value1(indx), 'Mood', '7', 'Happy', null);

    Insert into SpecificRec values (value1(indx), 'Temperature', null, null, null);

  DBMS_OUTPUT.put_line('wel patient values insert');
  /* insert values for heart*/
  End If;
  if dname1='COPD'
  then

    Insert into SpecificRec values (value1(indx), 'weight', null, null, null);
```

```
        Insert into SpecificRec values (value1(indx), 'BP_s', null, null, null);

        Insert into SpecificRec values (value1(indx), 'BP_d', null, null, null);

        Insert into SpecificRec values (value1(indx), 'Oxygen', '1', '99' , '90');

        Insert into SpecificRec values (value1(indx), 'Pain', null, null, null);

        Insert into SpecificRec values (value1(indx), 'Mood', null, null, null);

        Insert into SpecificRec values (value1(indx), 'Temperature', '1' , '100', '99');

      DBMS_OUTPUT.put_line('wel patient values insert');
      /* insert values for COPD*/
      End if;
      if dname1='HIV'
      then

        Insert into SpecificRec values (value1(indx), 'weight', '7', '200', '120');

        Insert into SpecificRec values (value1(indx), 'BP_s', '1', null, null);

        Insert into SpecificRec values (value1(indx), 'BP_d','1', null, null);

        Insert into SpecificRec values (value1(indx), 'Oxygen', null, null, null);

        Insert into SpecificRec values (value1(indx), 'Pain', '1', '5', null);

        Insert into SpecificRec values (value1(indx), 'Mood', null, null, null);

        Insert into SpecificRec values (value1(indx), 'Temperature', null, null, null);

        DBMS_OUTPUT.put_line('wel patient values insert');
      /* insert values for HIV*/
      End if;

    End if;
  End if;


 END Loop;

END CHECK_REC_PATIENT;
```

2. **Generate low activity alerts on login if the patient has not entered the observations during the specified time.**

```
create or replace PROCEDURE LOW_ACTIVITY AS
TYPE patientid IS TABLE OF varchar2(4);
value1 patientid;

TYPE obstype IS TABLE OF varchar2(30);
value2 obstype;

obsv1 date;
fre number;

BEGIN
  obsv1 :=null;
  fre :=null;
  value1 :=null;
  value2 :=null;

  execute immediate 'SELECT user_id from patient' bulk collect into value1;
  execute immediate 'SELECT distinct obs_type from specificrec' bulk collect into
value2;


  For indx in 1 .. value1.count
  Loop
    For ind in 1 .. value2.count
    Loop
    select frequency into fre from Specificrec where PATIENT_id =  value1(indx)
and OBS_TYPE=value2(ind);
    dbms_output.put_line(fre);

    Select max(obs_time) into obsv1 from Observation Where PATIENT_id =
value1(indx) and OBS_TYPE=value2(ind) ORDER BY obs_time DESC;
    dbms_output.put_line(obsv1);

    dbms_output.put_line(sysdate-obsv1);


    dbms_output.put_line(fre);
```

```
    if sysdate-obsv1 > fre
    then
    insert into alerts values (value1(indx), alert_sequence.nextval,
'Low_activity',value2(ind), sysdate, 'N', 'N');
    End if;
    end loop;
  end loop;
END LOW_ACTIVITY;
```

SQL Queries

Set -A

1. **List the number of health supporters that were authorized in the month of September 2016 by patients suffering from heart disease.**

```
Select Count(user_id)As No_HS from Health_supporter where auth_date between '01-SEP-16' AND '30-SEP-16' AND patient_id in
(SELECT patient_id from Diagnosis where dname='Heart');
```

Query Result

All Rows Fetched: 1 in 0.036 seconds

	NO_HS
1	0

2. **Give the number of patients who were not complying with the recommended frequency of recording observations.**

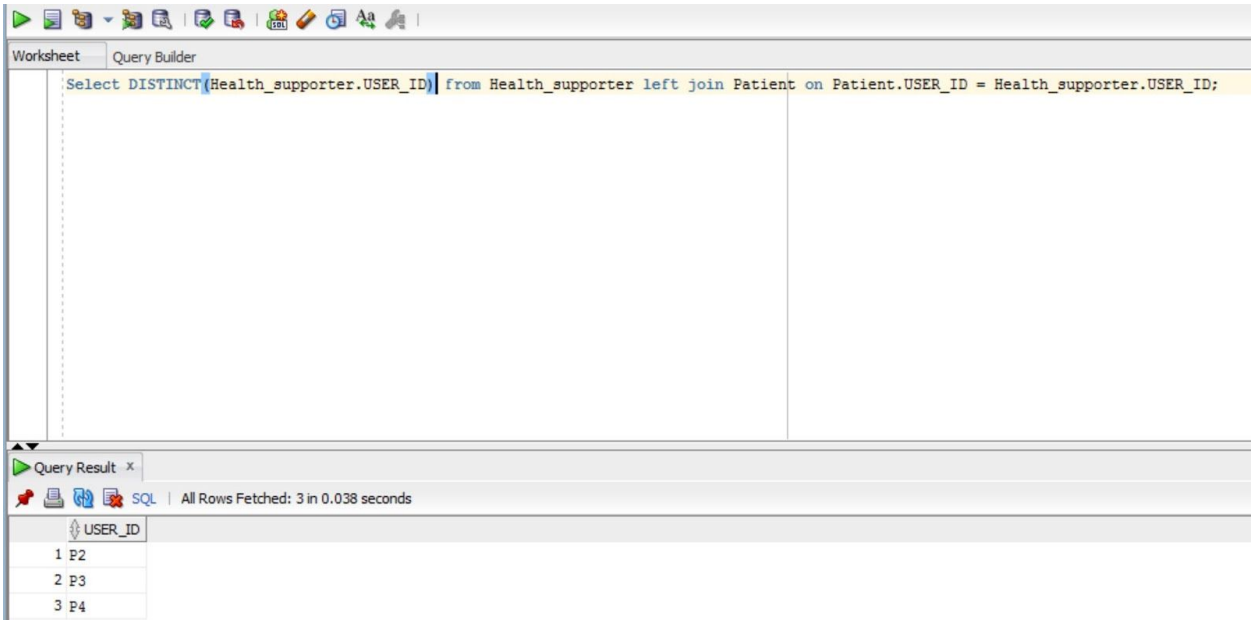
```
Select Count(patient_id) from alerts where ALERT_TYPE='low_activity';
```

Query Result

All Rows Fetched: 1 in 0.053 seconds

	COUNT(PATIENT_ID)
1	0

3. **List the health supporters who themselves are patients.**



4. **List the patients who are not 'sick'.**

Worksheet | Query Builder

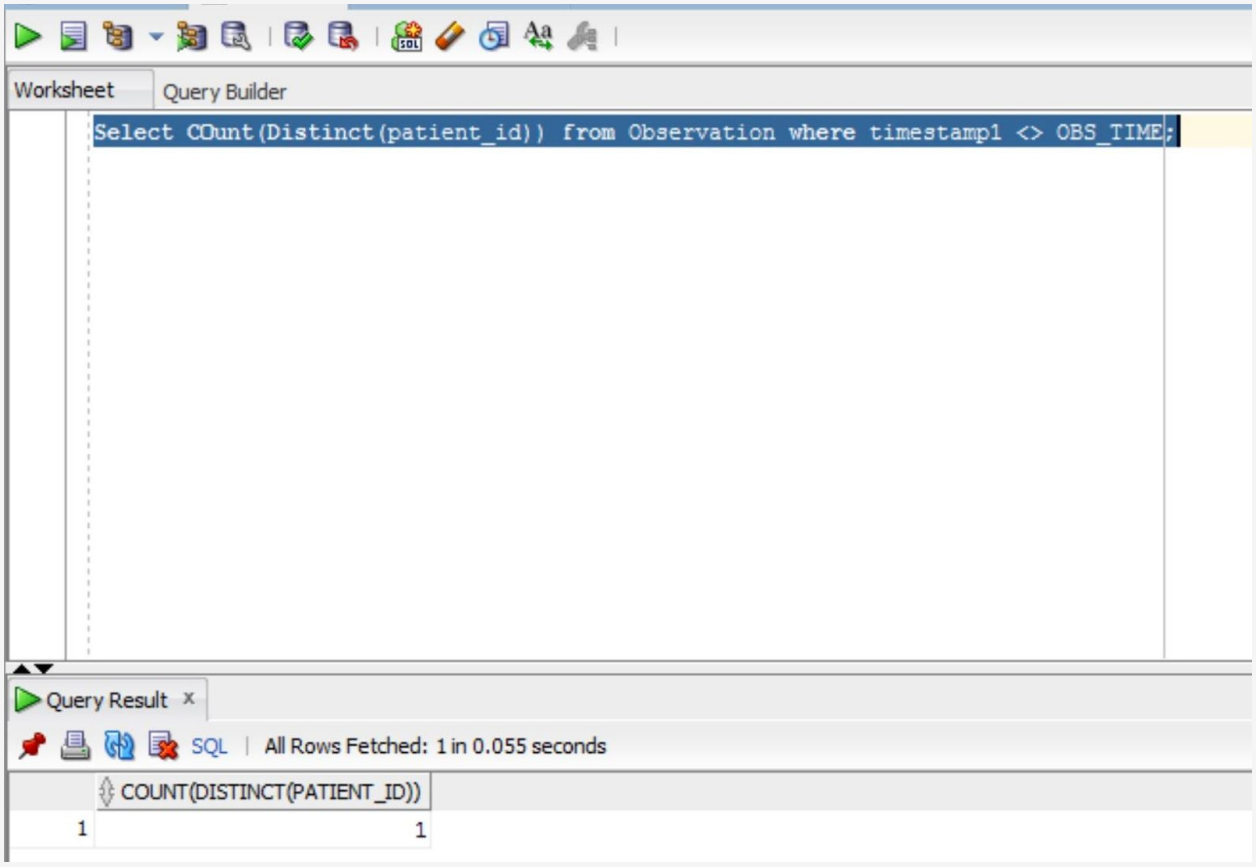
```
Select user_id from Patient where is_sick ='Y';
```

Query Result

SQL | All Rows Fetched: 2 in 0.037 seconds

	USER_ID
1	P1
2	P2

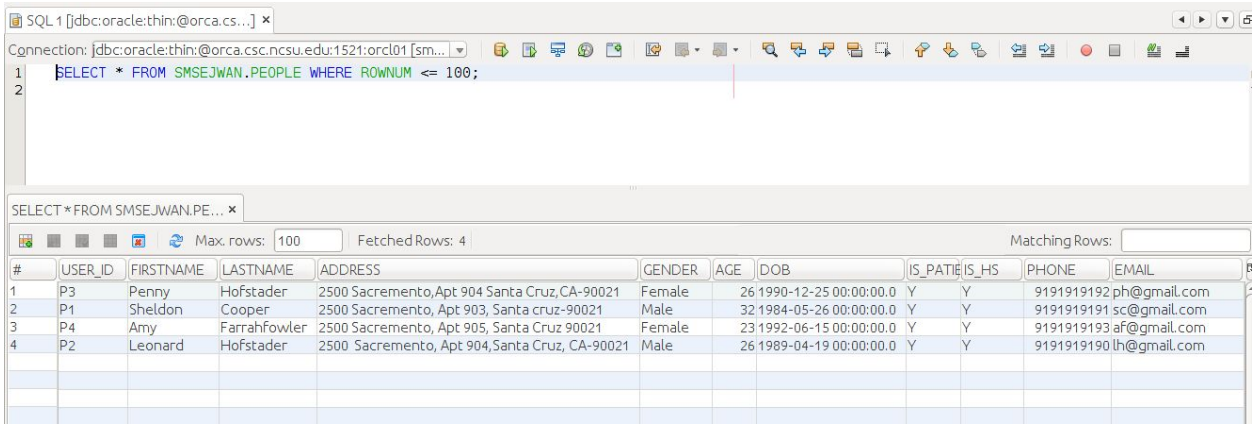
5. **How many patients have different observation time and recording time (of the observation).**



Set -B

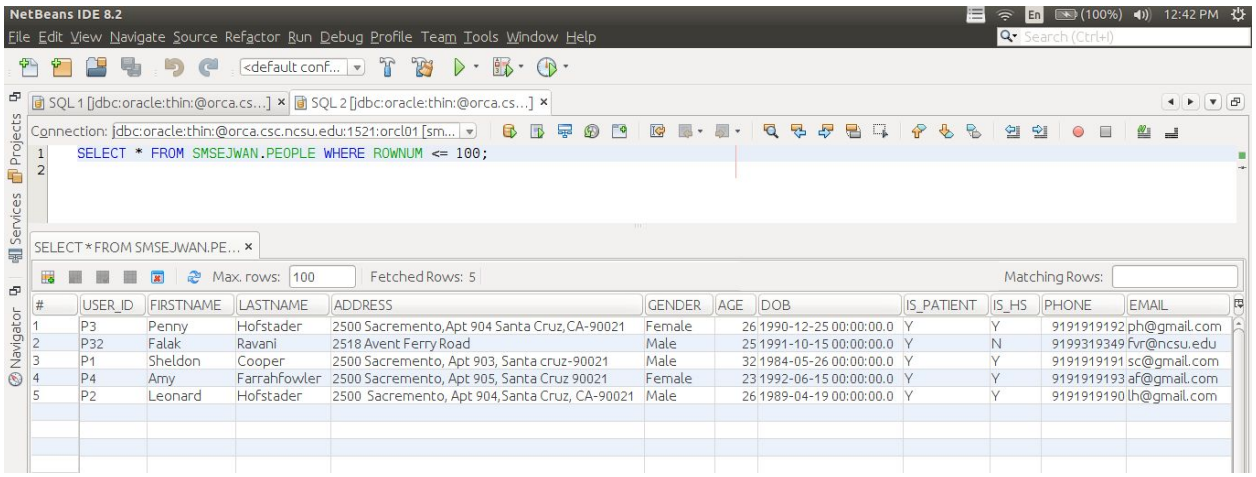
Insert/Update/Delete Queries

1. Add new patient or health supporter account.



#	USER_ID	FIRSTNAME	LASTNAME	ADDRESS	GENDER	AGE	DOB	IS_PATIE	IS_HS	PHONE	EMAIL
1	P3	Penny	Hofstader	2500 Sacremento,Apt 904 Santa Cruz,CA-90021	Female	26	1990-12-25 00:00:00.0	Y	Y	9191919192	ph@gmail.com
2	P1	Sheldon	Cooper	2500 Sacremento, Apt 903, Santa cruz-90021	Male	32	1984-05-26 00:00:00.0	Y	Y	9191919191	sc@gmail.com
3	P4	Amy	Farrahfowler	2500 Sacremento, Apt 905, Santa Cruz 90021	Female	23	1992-06-15 00:00:00.0	Y	Y	9191919193	af@gmail.com
4	P2	Leonard	Hofstader	2500 Sacremento, Apt 904,Santa Cruz, CA-90021	Male	26	1989-04-19 00:00:00.0	Y	Y	9191919190	lh@gmail.com

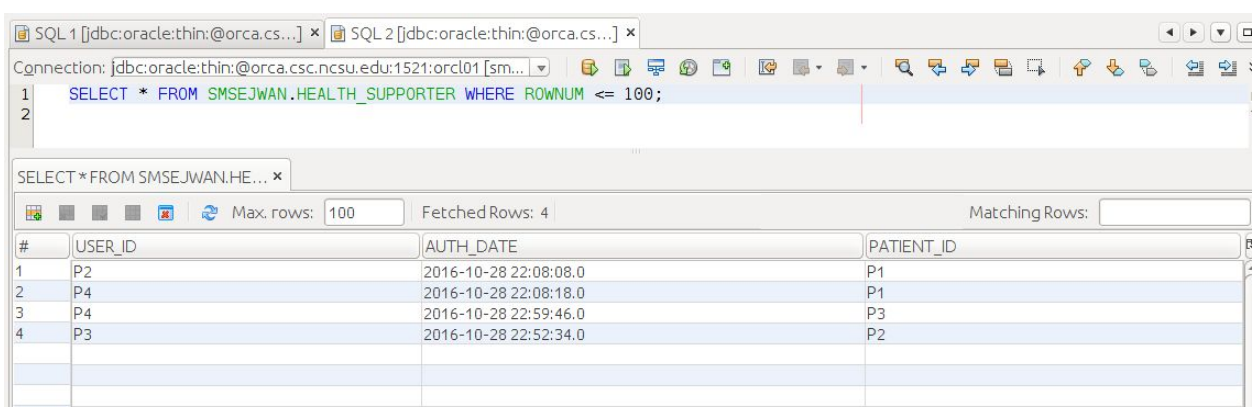
Before query 1



#	USER_ID	FIRSTNAME	LASTNAME	ADDRESS	GENDER	AGE	DOB	IS_PATIENT	IS_HS	PHONE	EMAIL
1	P3	Penny	Hofstader	2500 Sacremento,Apt 904 Santa Cruz,CA-90021	Female	26	1990-12-25 00:00:00.0	Y	Y	9191919192	ph@gmail.com
2	P32	Falak	Ravani	2518 Avent Ferry Road	Male	25	1991-10-15 00:00:00.0	Y	N	9199319349	fvr@ncsu.edu
3	P1	Sheldon	Cooper	2500 Sacremento, Apt 903, Santa cruz-90021	Male	32	1984-05-26 00:00:00.0	Y	Y	9191919191	sc@gmail.com
4	P4	Amy	Farrahfowler	2500 Sacremento, Apt 905, Santa Cruz 90021	Female	23	1992-06-15 00:00:00.0	Y	Y	9191919193	af@gmail.com
5	P2	Leonard	Hofstader	2500 Sacremento, Apt 904,Santa Cruz, CA-90021	Male	26	1989-04-19 00:00:00.0	Y	Y	9191919190	lh@gmail.com

After query 1

2. Add a new diagnosis or insert a health supporter for an existing patient.



#	USER_ID	AUTH_DATE	PATIENT_ID
1	P2	2016-10-28 22:08:08.0	P1
2	P4	2016-10-28 22:08:18.0	P1
3	P4	2016-10-28 22:59:46.0	P3
4	P3	2016-10-28 22:52:34.0	P2

Before query 2a

SQL 1 [jdbc:oracle:thin:@orca.cs...]

Connection: jdbc:oracle:thin:@orca.csc.ncsu.edu:1521:orcl01 [sm...

```
SELECT * FROM SMSEJWAN.HEALTH_SUPPORTER WHERE ROWNUM <= 100;
```

SELECT * FROM SMSEJWAN.HE...

Max. rows: 100 Fetched Rows: 5 Matching Rows:

#	USER_ID	AUTH_DATE	PATIENT_ID
1	P4	2016-10-30 12:53:36.0	P32
2	P2	2016-10-28 22:08:08.0	P1
3	P4	2016-10-28 22:08:18.0	P1
4	P4	2016-10-28 22:59:46.0	P3
5	P3	2016-10-28 22:52:34.0	P2

After query 2a

SQL 1 [jdbc:oracle:thin:@orca.cs...] SQL 2 [jdbc:oracle:thin:@orca.cs...]

Connection: jdbc:oracle:thin:@orca.csc.ncsu.edu:1521:orcl01 [sm...

```
SELECT * FROM SMSEJWAN.DIAGNOSIS WHERE ROWNUM <= 100;
```

SELECT * FROM SMSEJWAN.DI...

Max. rows: 100 Fetched Rows: 2 Matching Rows:

#	DNAME	PATIENT_ID
1	HIV	P2
2	Heart	P1

Before query 2b

SQL 1 [jdbc:oracle:thin:@orca.cs...] SQL 2 [jdbc:oracle:thin:@orca.cs...] SQL 3 [jdbc:oracle:thin:@orca.cs...]

Connection: jdbc:oracle:thin:@orca.csc.ncsu.edu:1521:orcl01 [sm...

```
SELECT * FROM SMSEJWAN.DIAGNOSIS WHERE ROWNUM <= 100;
```

SELECT * FROM SMSEJWAN.DI...

Max. rows: 100 Fetched Rows: 3 Matching Rows:

#	DNAME	PATIENT_ID
1	COPD	P32
2	HIV	P2
3	Heart	P1

After query 2b

3. Update the information for current patients.

SQL 1 [jdbc:oracle:thin:@orca.cs...]

Connection: jdbc:oracle:thin:@orca.csc.ncsu.edu:1521:orcl01 [sm...

```
SELECT * FROM SMSEJWAN.PEOPLE WHERE ROWNUM <= 100;
```

SELECT * FROM SMSEJWAN.PE...

Max. rows: 100 Fetched Rows: 5 Matching Rows:

#	USER_ID	FIRSTNAME	LASTNAME	ADDRESS	GENDER	AGE	DOB	IS_PATIENT	IS_HS	PHONE	EMAIL
1	P3	Penny	Hofstader	2500 Sacremento,Apt 904 Santa Cruz,CA-90021	Female	26	1990-12-25 00:00:00.0	Y	Y	9191919192	ph@gmail.com
2	P32	Falak	Ravani	2518 Avent Ferry Road	Male	25	1991-10-15 00:00:00.0	Y	N	9199319349	fvr@ncsu.edu
3	P1	Sheldon	Cooper	2500 Sacremento, Apt 903, Santa cruz-90021	Male	32	1984-05-26 00:00:00.0	Y	Y	9191919191	sc@gmail.com
4	P4	Amy	Farrahfowler	2500 Sacremento, Apt 905, Santa Cruz 90021	Female	23	1992-06-15 00:00:00.0	Y	Y	9191919193	af@gmail.com
5	P2	Leonard	Hofstader	2500 Sacremento, Apt 904,Santa Cruz, CA-90021	Male	26	1989-04-19 00:00:00.0	Y	Y	9191919190	lh@gmail.com

Before query 3

SQL 1 [jdbc:oracle:thin:@orca.cs...]

Connection: jdbc:oracle:thin:@orca.csc.ncsu.edu:1521:orcl01 [sm...

```
SELECT * FROM SMSEJWAN.PEOPLE WHERE ROWNUM <= 100;
```

SELECT * FROM SMSEJWAN.PE...

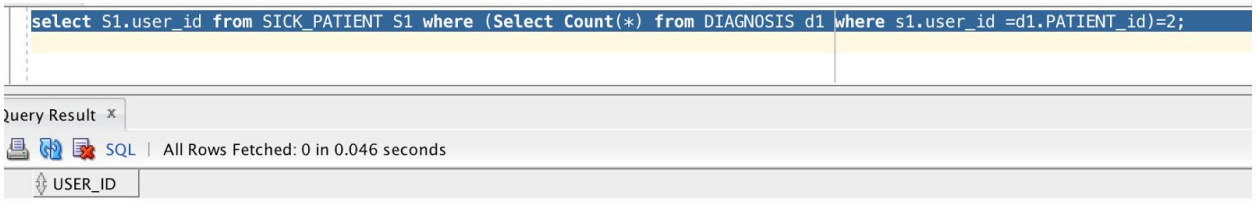
Max. rows: 100 Fetched Rows: 5 Matching Rows:

#	USER_ID	FIRSTNAME	LASTNAME	ADDRESS	GENDER	AGE	DOB	IS_PATIENT	IS_HS	PHONE	EMAIL
1	P3	Penny	Hofstader	2500 Sacremento,Apt 904 Santa Cruz,CA-90021	Female	26	1990-12-25 00:00:00.0	Y	Y	9191919192	ph@gmail.com
2	P32	Falak	Ravani	2582 Gorman Street	Male	25	1991-10-15 00:00:00.0	Y	N	3449311234	fvr@ncsu.edu
3	P1	Sheldon	Cooper	2500 Sacremento, Apt 903, Santa cruz-90021	Male	32	1984-05-26 00:00:00.0	Y	Y	9191919191	sc@gmail.com
4	P4	Amy	Farrahfowler	2500 Sacremento, Apt 905, Santa Cruz 90021	Female	23	1992-06-15 00:00:00.0	Y	Y	9191919193	af@gmail.com
5	P2	Leonard	Hofstader	2500 Sacremento, Apt 904,Santa Cruz, CA-90021	Male	26	1989-04-19 00:00:00.0	Y	Y	9191919190	lh@gmail.com

After query 4

4. Find patients who belong to more than one Sick Patient class

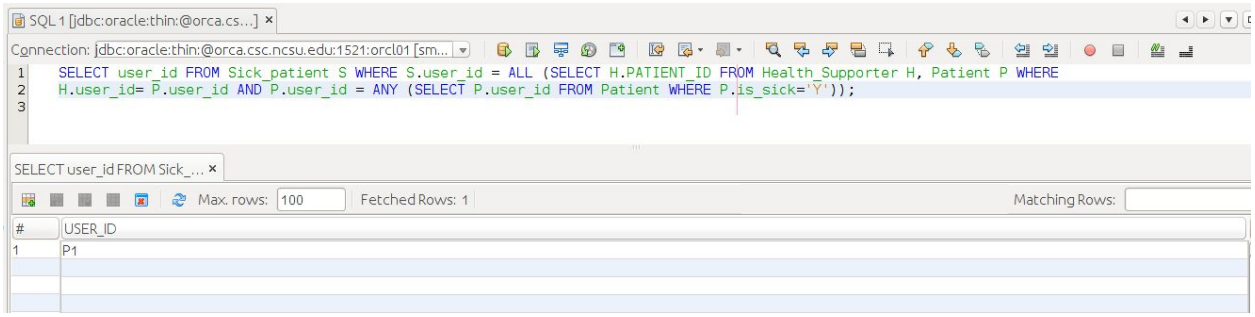
select S1.user_id from SICK_PATIENT S1 where (Select Count(*) from DIAGNOSIS d1 where s1.user_id =d1.PATIENT_id)=2;



5. Find all Sick patients whose Health Supporters are also Sick patients

Query ->

SELECT user_id FROM Sick_patient S WHERE S.user_id = ALL (SELECT H.PATIENT_ID FROM Health_Supporter H, Patient P WHERE H.user_id= P.user_id AND P.user_id = ANY (SELECT P.user_id FROM Patient WHERE P.is_sick='Y'));



6. Find patients who have two Health Supporters

select P1.user_id from Patient P1 where (select count(*) from HEALTH_SUPPORTER h1 where h1.patient_id = p1.user_id)=2;

```sql
select P1.user_id from Patient P1 where (select count(*) from HEALTH_SUPPORTER h1 where h1.patient_id = p1.user_id)=2;
```

Query Result

SQL | All Rows Fetched: 1 in 0.047 seconds

	USER_ID
1	P1