REVIEW-1

INTRODUCTION:

Our project is based on orphan house database management system. The database contains all the information about the kids present in this orphanage, staff and ngo's which are connected to them. It also contains details about various facilities like education and health provided to them. Information about the families which are interested in adoption, and the various events organized to raise awareness and funds, all this information can be found in this database.

DATA REQUIREMENTS:

ENTITIES:

1. **KIDS:** An entity type which has many attributes like Name (composite attribute) which is divided into first, middle and last name. Every child has been given a K-ID which is unique for every child. Other attributes like religion and gender,D.O.B and D.O.J are also present.
2. **HEALTH REPORT(KIDS):** An entity type which has attributes like food allergies, an ID which is different for every child, height, weight and blood group.

3. **ADOPTIVE FAMILIES:** An entity type which has many attributes like Name of the parents, Address, Salary earned by parents, Blood Group, Religion. Every family has a F-ID which is uniquely identified. Other attributes like health problems and contact details are multi valued types. Apart from these attributes D.O.B and D.O.A are also present. There is one more attribute called Age which is a derived attribute (parent attribute- D.O.B).

4. **STAFF:** An entity type which has attributes like Name, Post , Address. Every staff member has a distinct S-ID . There are other attributes called Salary, Phone number and D.O.B.

5. **NGO'S :** An entity type which has attributes like Name and address. Every Ngo which has an association with the orphanage has an unique ID called N-ID. There are two multi valued attributes. First is contact details The other is cause the NGO supports.

6. **TUTORS:** An entity type which has attributes like name, degree and contact details of the teacher. T-ID is another attribute assigned to every tutor which teaches the kids. There is another attribute called subject which is a multi valued attribute

7. **DOCTORS:** An entity type which has attributes like name, degree, contact details and speciality. D-ID is another attribute which is assigned to every doctor which works with the NGO for the orphanage.

8. **EVENTS:** This is a weak entity type. It has attributes like Name, budget and Date of event. There is a multi-valued attribute called cause for the event.

RELATIONSHIPS:

1. Kids are adopted by adopted families (M-1)

One child can be adopted by only one family but a family can adopt more than one child. Children have partial participation while family has total participation.

2. Kids are taken care by staff (M-N)

A staff member can take care of more than one child and a child can be taken care of by more than one staff member.
Both have total participation.

3.Kids are taught by tutors (M-N)

One tutor can teach more than one child. Similarly a child can be taught by multiple tutors. Both have total participation.

4.Doctor makes the health report (1-M)

A doctor can make more than one health report but a child can have only one health report.
Both have total participation.

5.NGO’S hosts events (M-N)

A NGO can organize multiple events and one event can be organized by more than one NGO. Events has total participation while NGO has partial partipation as it is not necessary for the NGO to organize all the events, but it is necessary for all the events to be organized by the NGO.

6. Kids participate in events (M-N)

Children can participate in multiple events and one event can have multiple participants.
Both have total participation.

7. Doctors have tie up with NGO's (M-N)

A NGO can have tie up with more than one doctor. Similarly a doctor can have tie up with more than one NGO.
Both have total participation.

8.Tutors have tie up with NGO's (M-N)

A NGO can have tie up with more than one tutor. Similarly a teacher can have tie up with more than one NGO.
Both have total participation.

Functional Requirements:

USER (staff of orphanage)

The System must allow users to login if they enter the correct login id and password according to the post they hold in the orphanage. The user must be able to access the following:

- Details of the kids in the orphanage
- Details of the families who adopt the kids and also the families who are interested in adopting children.
- The health report for each kid should also be visible accordingly.
- Detail of each staff member.
- Details of the NGO'S
- Details of the events organised/sponsored by them. Also event participation details.
- Details of the doctors, tutors provided by different NGO's.

Basic Analogy

1) View the website
2) Login in the website
3) Select the menu of which details you want to see
4) View the kids details
5) View respective adoptive family detail
6) View the health report of each kid
7) View the NGO's who sponsor
8) View the details of Events taking place
9) View details of the Tutors and Doctors
10) View the Staff details

Example- View all staff details

1) The name
2) The address
3) The contact no.
4) The salary
5) The post
6) The staff ID

ADMINISTRATOR

Administrator of this orphanage based database system is in charge of creating the website for the orphanage which is used to access the database. Administrator has all the privileges of the user but has the authority to add, update and remove data from the database which the user cannot do.

The administrator is responsible for:

- Creating login ids and passwords for different users accounts.
- They have the authority to update the information in the database as well.
- They can add new NGO’S that have tied up with the orphanage
- Modify the contact no. of an adoptive family if needed.
- Modify the health report from time to time basis.
- If a staff member resigns then the administrator can remove the details of the respective staff member from the database as well.

Basic analogy

1) Create website
2) Generate logins
3) Display menus to choose from (eg- kid, staff e.t.c.)
4) Add , update or remove the data according to the requirements

○ different scenarios of removal of old data

♦ If any staff member resigns or is removed, then their data needs to be removed from the database.

♦ If a NGO withdraws their sponsorship from the orphanage, their data needs to be removed.

♦ If a doctor or a tutor resigns or is removed, then their data needs to be removed from the database

♦ If an event that was supposed to be conducted gets cancelled due unforeseen reasons, its details need to be removed

○ different scenarios for modification of existing data

1 After the regular checkups the kids health report data (eg. height, weight) needs to be modified if any changes occur.
2 If a staff member gets a promotion their post and salary should be modified accordingly.

3. If an adoptive family is relocating to a new address or they change their contact details then the following details needs to be modified.
4. If the event budget is changed due to any unforeseen reason, the budget needs to be modified.

- **eight different scenarios of data retrieval.**

- View all the information of the kid

  Retrieved data:
  a) Name
  b) Gender
  c) Religion
  d) K-ID
  e) D.o.b
  f) D.o.j

- View the health problems and id-proofs of all the adoptive families

  We are retrieving the health problems with the id-proof
  a) F-ID
  b) Health problems

- View the name and contact details of all the staff members.

The Retrieved data:

a) Name
b) Contact details

- View all the details of the doctors

The retrieved data:

a) Name
b) D-ID
c) Qualifications
d) Speciality
e) Contact
f) D.O.B

- View all the details of the tutors

The retrieved data:

a) Name
b) T-ID
c) Subject
d) Qualification
e) Contact
f) D.O.B

- View the health report of a particular kid

The retrieved data:

a) K-ID (of the respective kid)

b) Weight
c) Height
d) Blood group
e) Food allergies

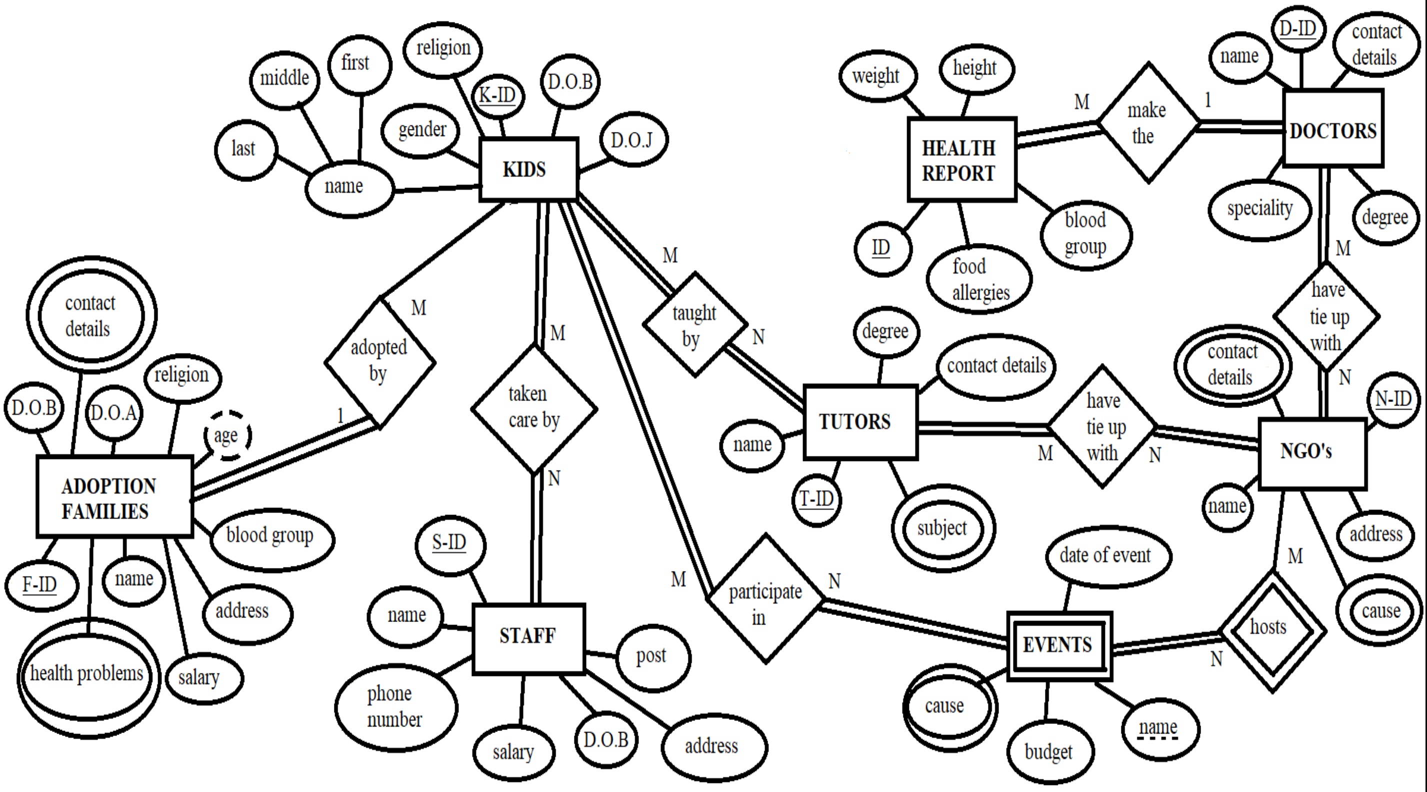
- View the name of kids and their blood groups

  The retrieved data with the specified blood group:
  a) Name
  b) Blood group

- View the budget and names of all the events conducted

  The retrieved data:
  a) Name
  b) Budget

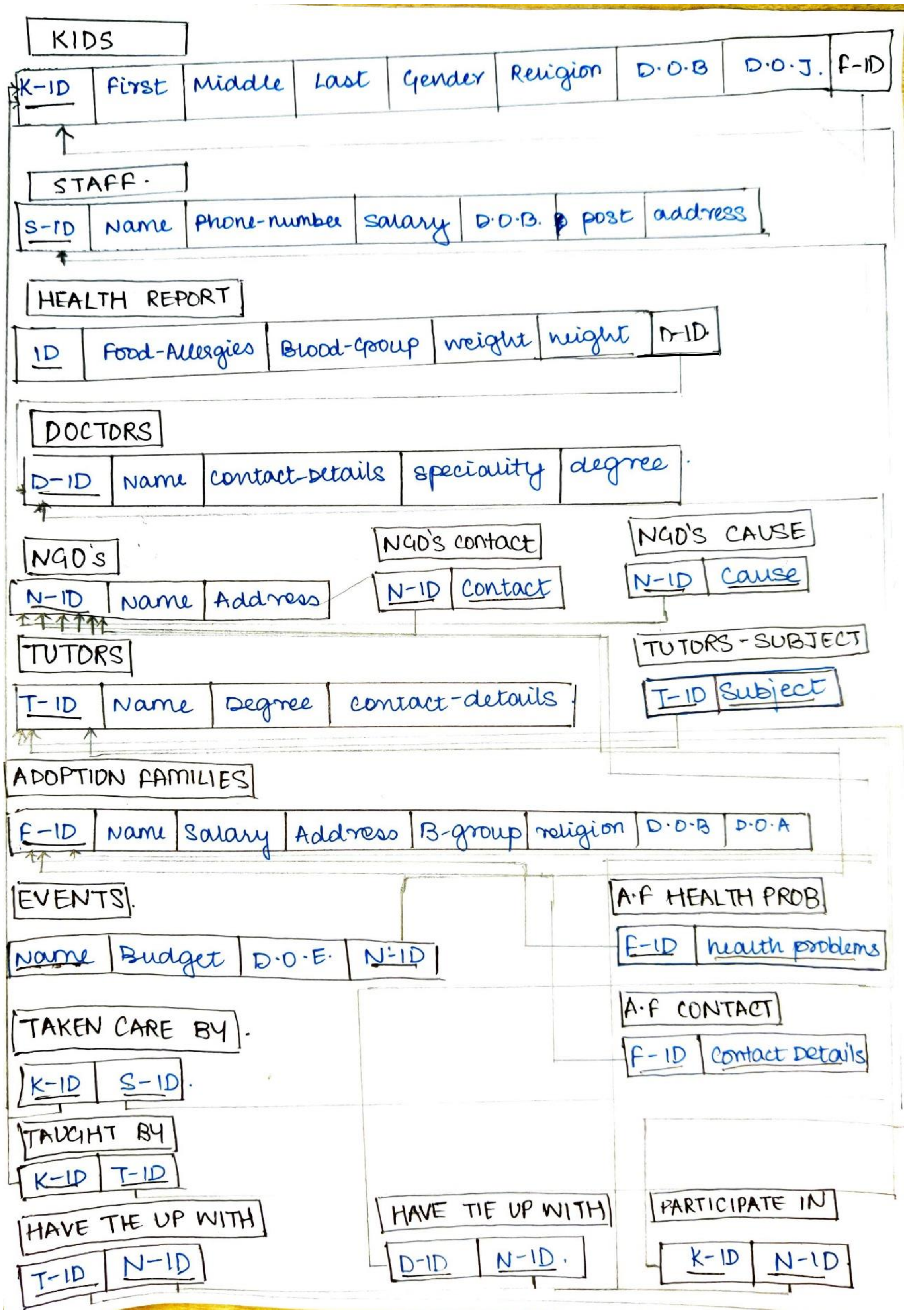
E-R DIAGRAM:



KIDS
name
first
middle
last
religion
K-ID
D.O.B
gender
D.O.J
adopted by
taken care by
taught by
participate in
ADOPTION FAMILIES
contact details
religion
D.O.B
D.O.A
age
blood group
F-ID
name
address
health problems
salary
STAFF
S-ID
name
phone number
post
salary
D.O.B
address
HEALTH REPORT
weight
height
ID
food allergies
blood group
make the
DOCTORS
D-ID
name
contact details
speciality
degree
TUTORS
degree
contact details
name
T-ID
subject
have tie up with
NGO's
contact details
N-ID
name
address
cause
hosts
EVENTS
date of event
cause
budget
name
M
N
1

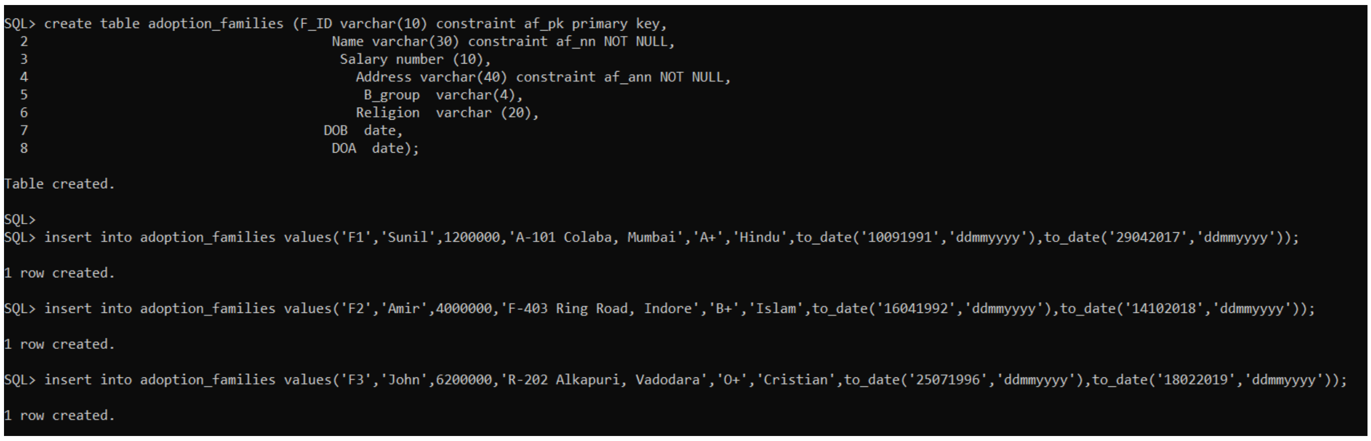
4.

E-R DIAGRAM TO RELATIONAL MODEL

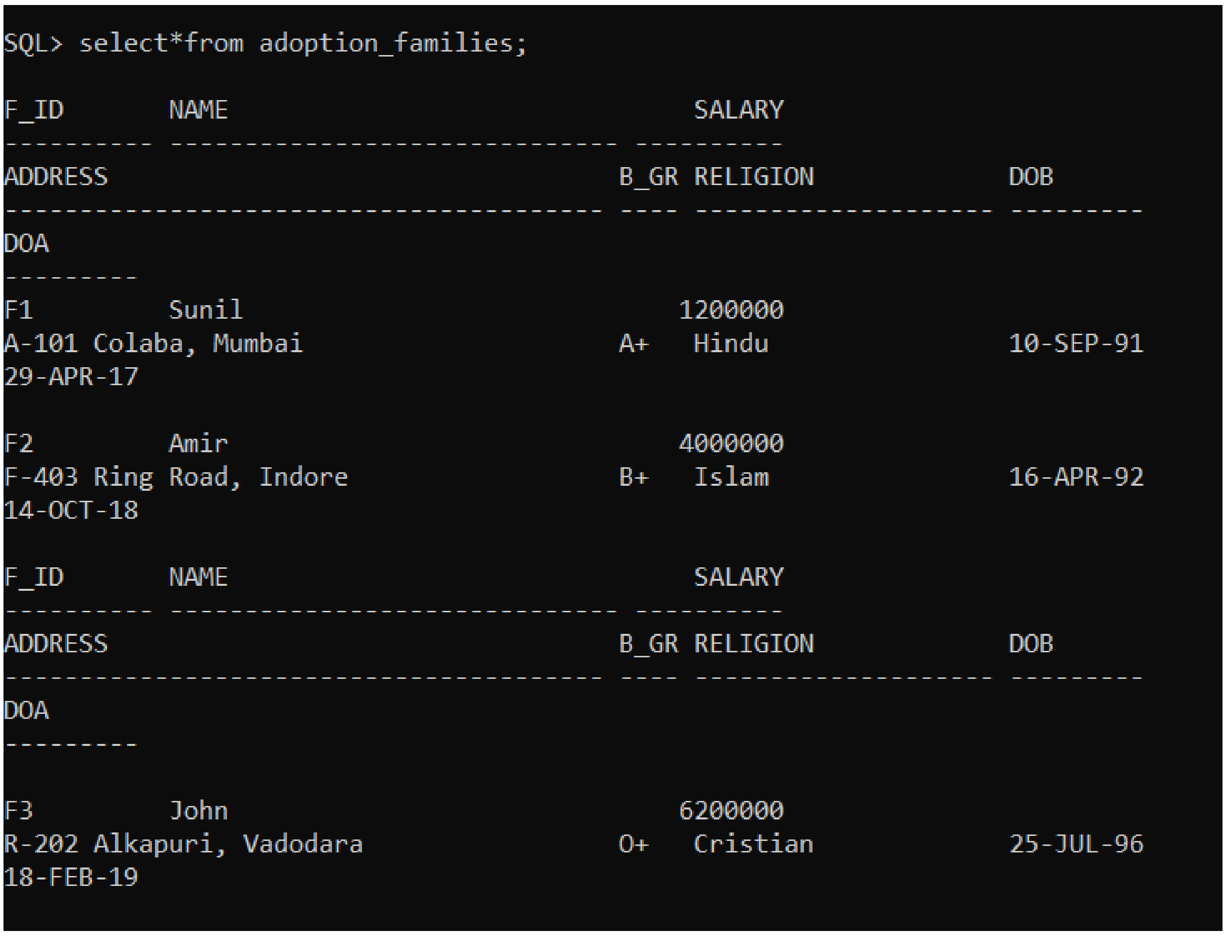

KIDS
K-ID | First | Middle | Last | Gender | Religion | D.O.B | D.O.J. | F-ID
STAFF.
S-ID | Name | Phone-number | Salary | D.O.B. | post | address
HEALTH REPORT
ID | Food-Allergies | Blood-Group | weight | height | D-ID
DOCTORS
D-ID | Name | contact-Details | speciality | degree
NGO's
N-ID | Name | Address
NGO'S contact
N-ID | Contact
NGO'S CAUSE
N-ID | Cause
TUTORS
T-ID | Name | Degree | contact-details
TUTORS-SUBJECT
T-ID | Subject
ADOPTION FAMILIES
F-ID | Name | Salary | Address | B-group | religion | D.O.B | D.O.A
EVENTS
Name | Budget | D.O.E. | N-ID
A.F HEALTH PROB
F-ID | health problems
A.F CONTACT
F-ID | Contact Details
TAKEN CARE BY
K-ID | S-ID
TAUGHT BY
K-ID | T-ID
HAVE TIE UP WITH
T-ID | N-ID
HAVE TIE UP WITH
D-ID | N-ID
PARTICIPATE IN
K-ID | N-ID

5(a) & 5(b).

ADOPTION FAMILIES

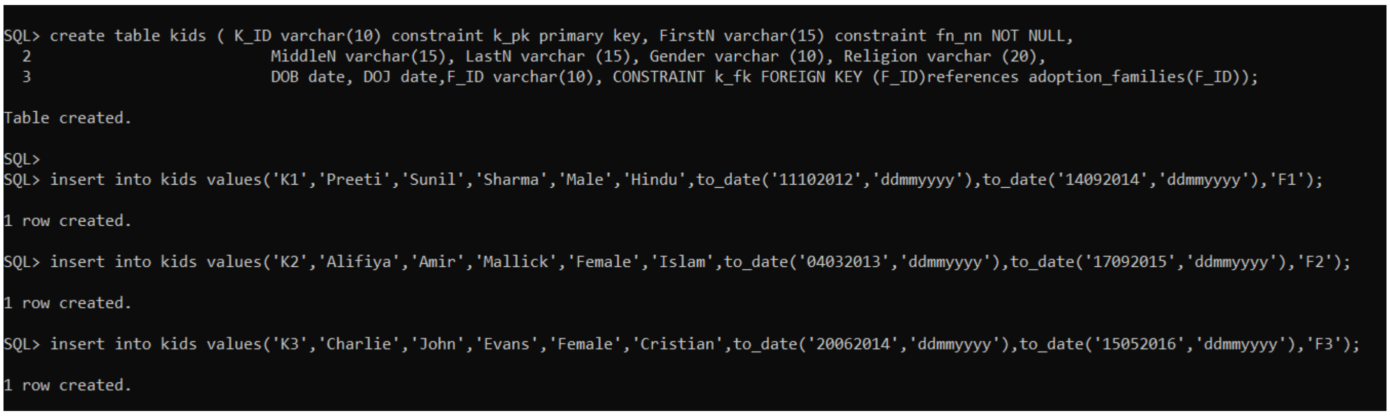


```
SQL> create table adoption_families (F_ID varchar(10) constraint af_pk primary key,
                                        Name varchar(30) constraint af_nn NOT NULL,
                                         Salary number (10),
                                           Address varchar(40) constraint af_ann NOT NULL,
                                            B_group  varchar(4),
                                           Religion  varchar (20),
                                       DOB  date,
                                        DOA  date);

Table created.

SQL>
SQL> insert into adoption_families values('F1','Sunil',1200000,'A-101 Colaba, Mumbai','A+','Hindu',to_date('10091991','ddmmyyyy'),to_date('29042017','ddmmyyyy'));

1 row created.

SQL> insert into adoption_families values('F2','Amir',4000000,'F-403 Ring Road, Indore','B+','Islam',to_date('16041992','ddmmyyyy'),to_date('14102018','ddmmyyyy'));

1 row created.

SQL> insert into adoption_families values('F3','John',6200000,'R-202 Alkapuri, Vadodara','O+','Cristian',to_date('25071996','ddmmyyyy'),to_date('18022019','ddmmyyyy'));

1 row created.
```

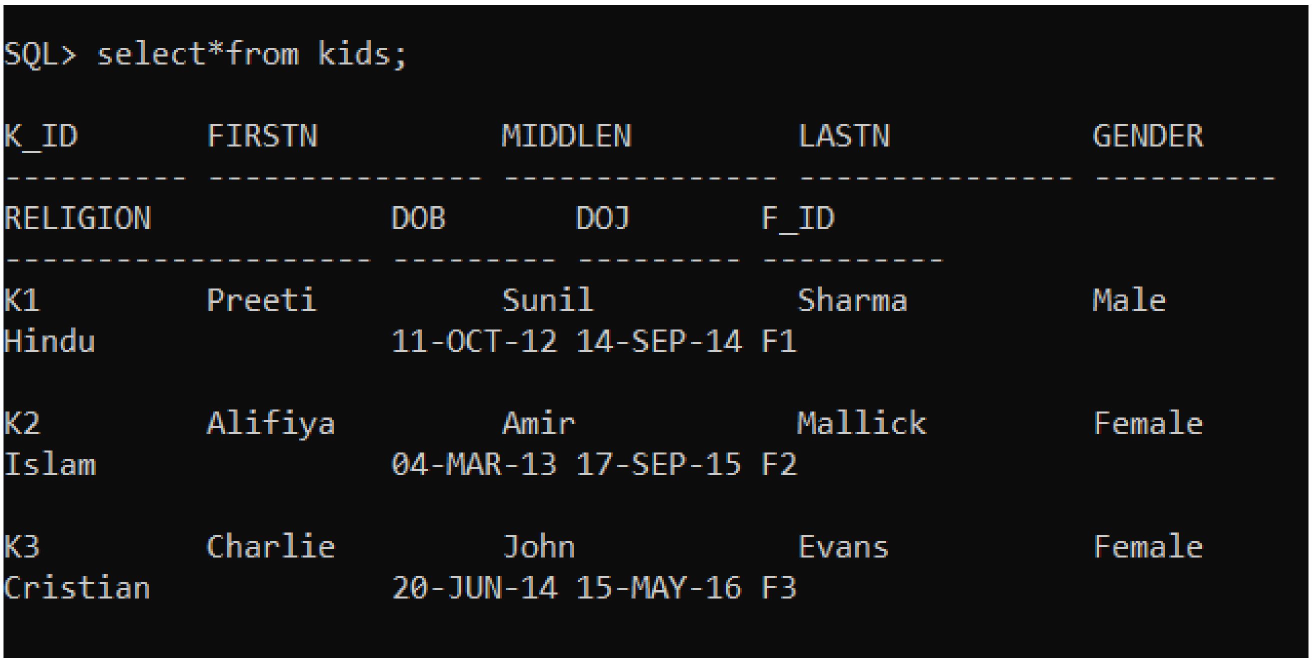


```
SQL> select*from adoption_families;

F_ID       NAME                               SALARY
---------- ------------------------------ ----------
ADDRESS                                  B_GR RELIGION             DOB
---------------------------------------- ---- -------------------- ---------
DOA
---------
F1         Sunil                             1200000
A-101 Colaba, Mumbai                     A+   Hindu                10-SEP-91
29-APR-17

F2         Amir                              4000000
F-403 Ring Road, Indore                  B+   Islam                16-APR-92
14-OCT-18

F_ID       NAME                               SALARY
---------- ------------------------------ ----------
ADDRESS                                  B_GR RELIGION             DOB
---------------------------------------- ---- -------------------- ---------
DOA
---------

F3         John                              6200000
R-202 Alkapuri, Vadodara                 O+   Cristian             25-JUL-96
18-FEB-19
```

KIDS

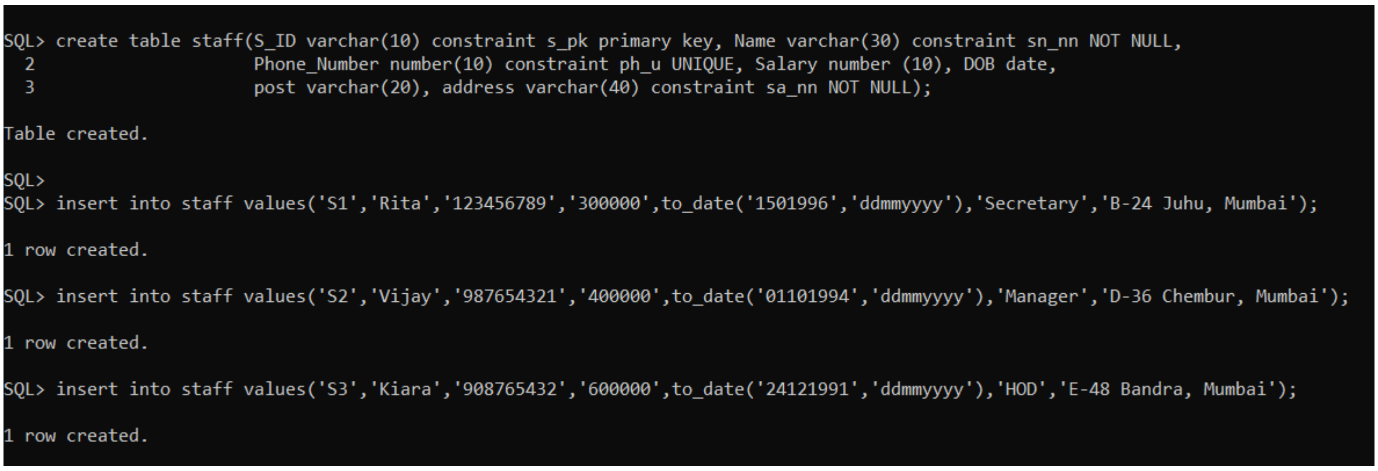


```
SQL> create table kids ( K_ID varchar(10) constraint k_pk primary key, FirstN varchar(15) constraint fn_nn NOT NULL,
  2                           MiddleN varchar(15), LastN varchar (15), Gender varchar (10), Religion varchar (20),
  3                           DOB date, DOJ date,F_ID varchar(10), CONSTRAINT k_fk FOREIGN KEY (F_ID)references adoption_families(F_ID));

Table created.

SQL>
SQL> insert into kids values('K1','Preeti','Sunil','Sharma','Male','Hindu',to_date('11102012','ddmmyyyy'),to_date('14092014','ddmmyyyy'),'F1');

1 row created.

SQL> insert into kids values('K2','Alifiya','Amir','Mallick','Female','Islam',to_date('04032013','ddmmyyyy'),to_date('17092015','ddmmyyyy'),'F2');

1 row created.

SQL> insert into kids values('K3','Charlie','John','Evans','Female','Cristian',to_date('20062014','ddmmyyyy'),to_date('15052016','ddmmyyyy'),'F3');

1 row created.
```

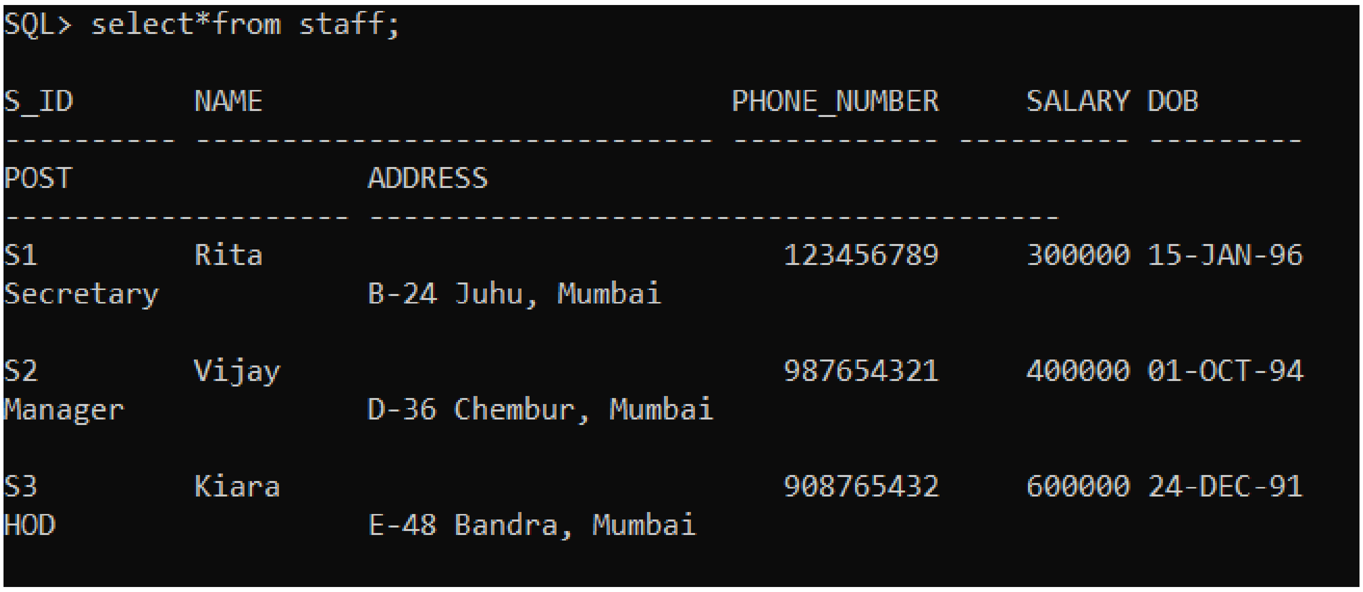


```
SQL> select*from kids;

K_ID       FIRSTN          MIDDLEN         LASTN           GENDER
---------- --------------- --------------- --------------- ----------
RELIGION             DOB       DOJ       F_ID
-------------------- --------- --------- ----------
K1         Preeti          Sunil           Sharma          Male
Hindu                11-OCT-12 14-SEP-14 F1

K2         Alifiya         Amir            Mallick         Female
Islam                04-MAR-13 17-SEP-15 F2

K3         Charlie         John            Evans           Female
Cristian             20-JUN-14 15-MAY-16 F3
```

STAFF

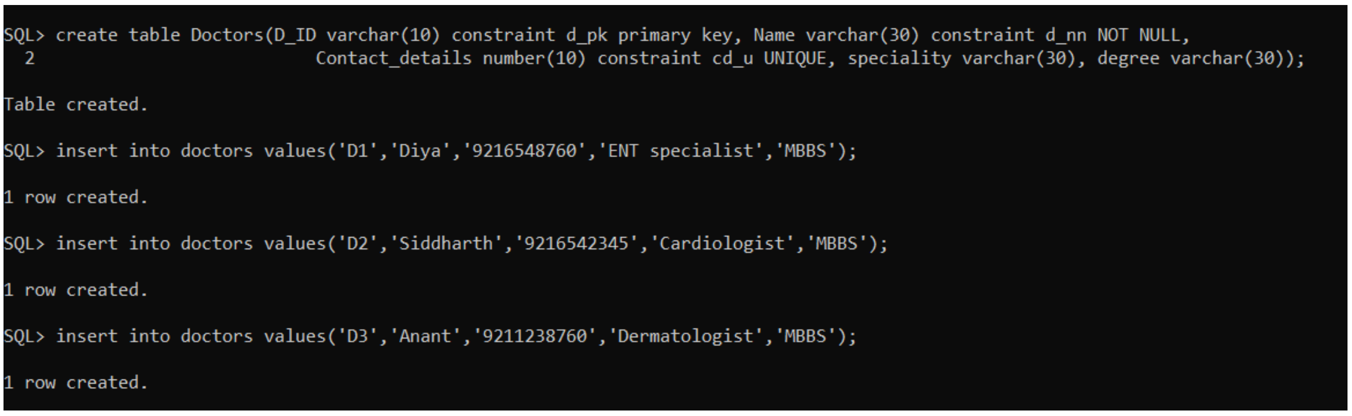


```
SQL> create table staff(S_ID varchar(10) constraint s_pk primary key, Name varchar(30) constraint sn_nn NOT NULL,
  2                    Phone_Number number(10) constraint ph_u UNIQUE, Salary number (10), DOB date,
  3                    post varchar(20), address varchar(40) constraint sa_nn NOT NULL);

Table created.

SQL>
SQL> insert into staff values('S1','Rita','123456789','300000',to_date('1501996','ddmmyyyy'),'Secretary','B-24 Juhu, Mumbai');

1 row created.

SQL> insert into staff values('S2','Vijay','987654321','400000',to_date('01101994','ddmmyyyy'),'Manager','D-36 Chembur, Mumbai');

1 row created.

SQL> insert into staff values('S3','Kiara','908765432','600000',to_date('24121991','ddmmyyyy'),'HOD','E-48 Bandra, Mumbai');

1 row created.
```

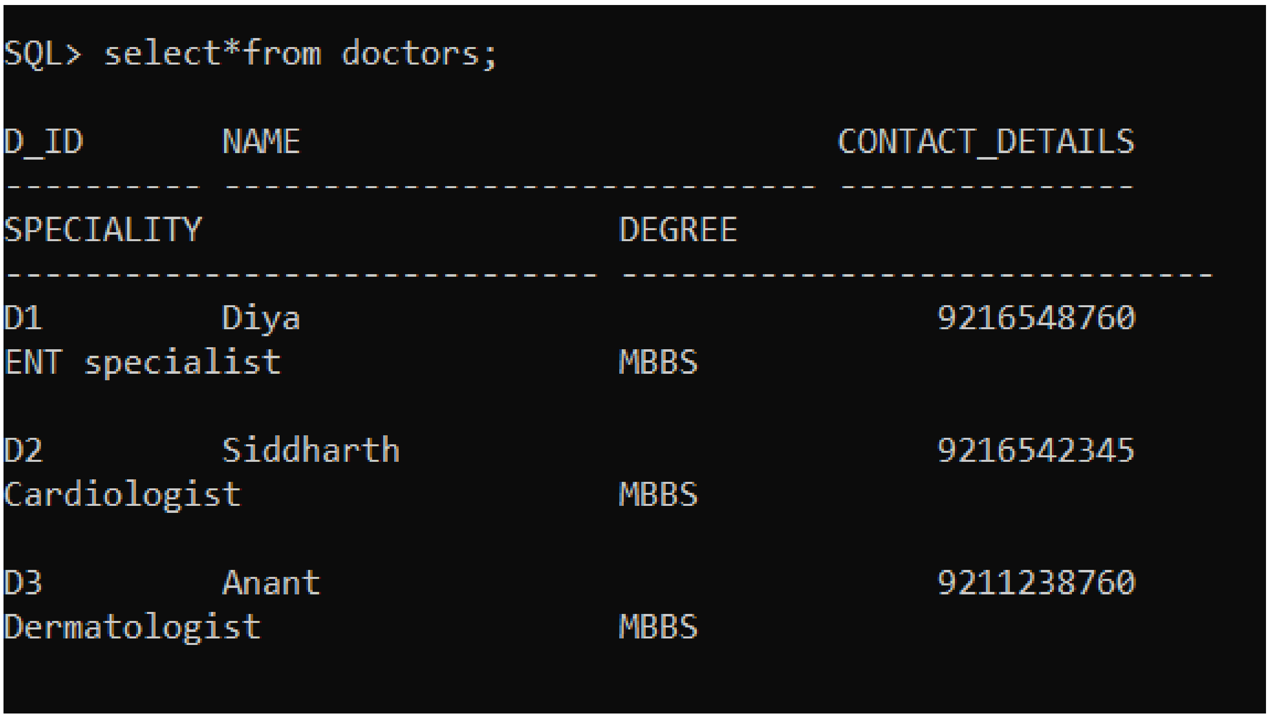


```
SQL> select*from staff;

S_ID       NAME                           PHONE_NUMBER     SALARY DOB
---------- ------------------------------ ------------ ---------- ---------
POST                 ADDRESS
-------------------- ----------------------------------------
S1         Rita                              123456789     300000 15-JAN-96
Secretary            B-24 Juhu, Mumbai

S2         Vijay                             987654321     400000 01-OCT-94
Manager              D-36 Chembur, Mumbai

S3         Kiara                             908765432     600000 24-DEC-91
HOD                  E-48 Bandra, Mumbai
```

DOCTORS

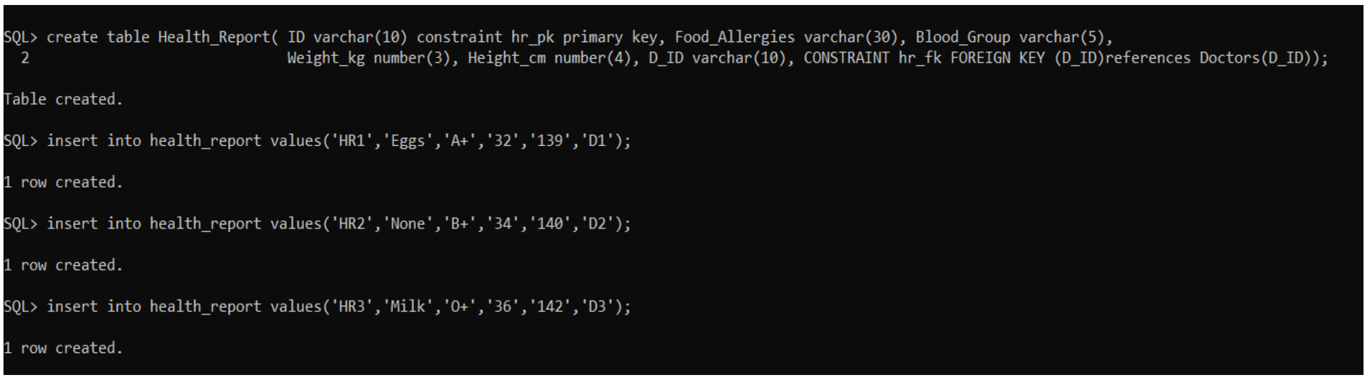


```
SQL> create table Doctors(D_ID varchar(10) constraint d_pk primary key, Name varchar(30) constraint d_nn NOT NULL,
  2                       Contact_details number(10) constraint cd_u UNIQUE, speciality varchar(30), degree varchar(30));

Table created.

SQL> insert into doctors values('D1','Diya','9216548760','ENT specialist','MBBS');

1 row created.

SQL> insert into doctors values('D2','Siddharth','9216542345','Cardiologist','MBBS');

1 row created.

SQL> insert into doctors values('D3','Anant','9211238760','Dermatologist','MBBS');

1 row created.
```

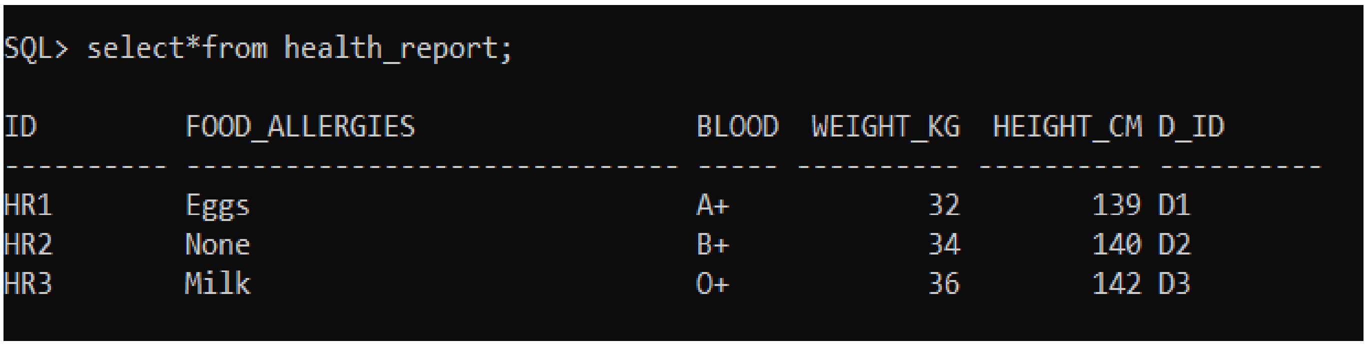


```
SQL> select*from doctors;

D_ID       NAME                           CONTACT_DETAILS
---------- ------------------------------ ---------------
SPECIALITY                     DEGREE
------------------------------ ------------------------------
D1         Diya                                9216548760
ENT specialist                 MBBS

D2         Siddharth                           9216542345
Cardiologist                   MBBS

D3         Anant                               9211238760
Dermatologist                  MBBS
```

HEALTH REPORT

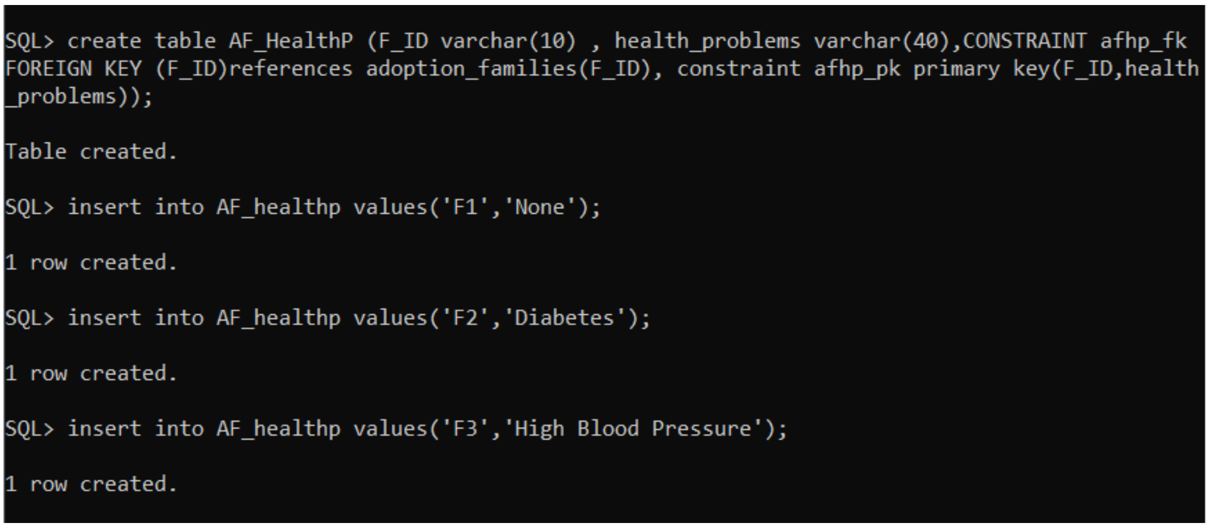


```
SQL> create table Health_Report( ID varchar(10) constraint hr_pk primary key, Food_Allergies varchar(30), Blood_Group varchar(5),
  2                           Weight_kg number(3), Height_cm number(4), D_ID varchar(10), CONSTRAINT hr_fk FOREIGN KEY (D_ID)references Doctors(D_ID));

Table created.

SQL> insert into health_report values('HR1','Eggs','A+','32','139','D1');

1 row created.

SQL> insert into health_report values('HR2','None','B+','34','140','D2');

1 row created.

SQL> insert into health_report values('HR3','Milk','O+','36','142','D3');

1 row created.
```

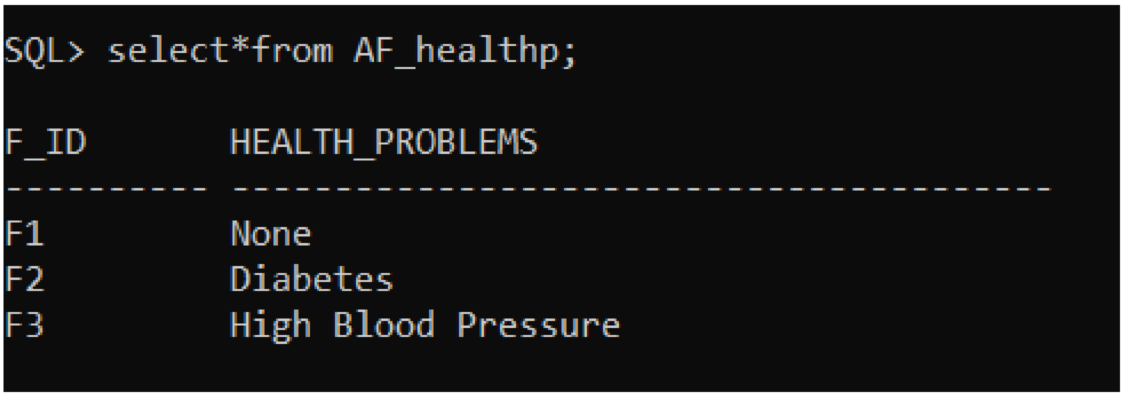


```
SQL> select*from health_report;

ID         FOOD_ALLERGIES                 BLOOD  WEIGHT_KG  HEIGHT_CM D_ID
---------- ------------------------------ ----- ---------- ---------- ----------
HR1        Eggs                           A+            32        139 D1
HR2        None                           B+            34        140 D2
HR3        Milk                           O+            36        142 D3
```

AF_HEALTHP

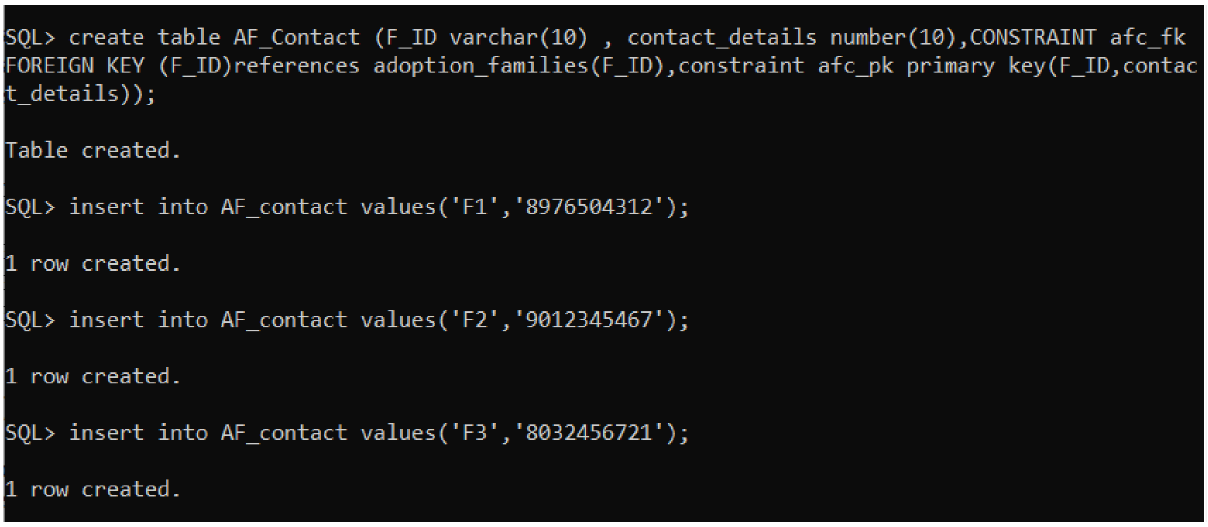


```
SQL> create table AF_HealthP (F_ID varchar(10) , health_problems varchar(40),CONSTRAINT afhp_fk
FOREIGN KEY (F_ID)references adoption_families(F_ID), constraint afhp_pk primary key(F_ID,health
_problems));

Table created.

SQL> insert into AF_healthp values('F1','None');

1 row created.

SQL> insert into AF_healthp values('F2','Diabetes');

1 row created.

SQL> insert into AF_healthp values('F3','High Blood Pressure');

1 row created.
```

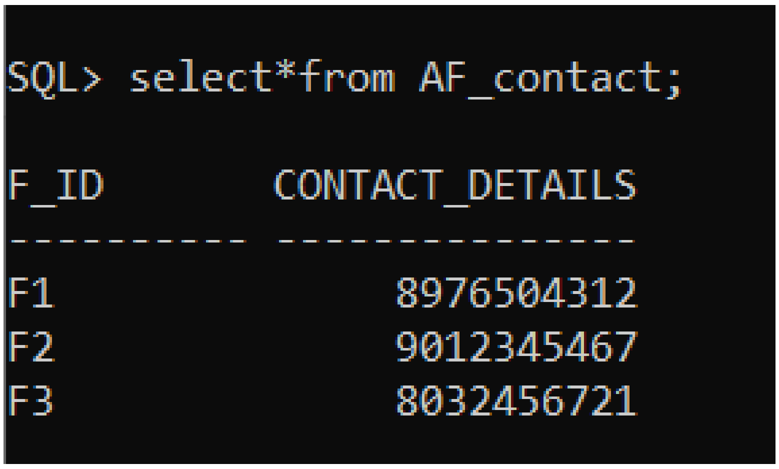


```
SQL> select*from AF_healthp;

F_ID       HEALTH_PROBLEMS
---------- ----------------------------------------
F1         None
F2         Diabetes
F3         High Blood Pressure
```

AF_CONTACT

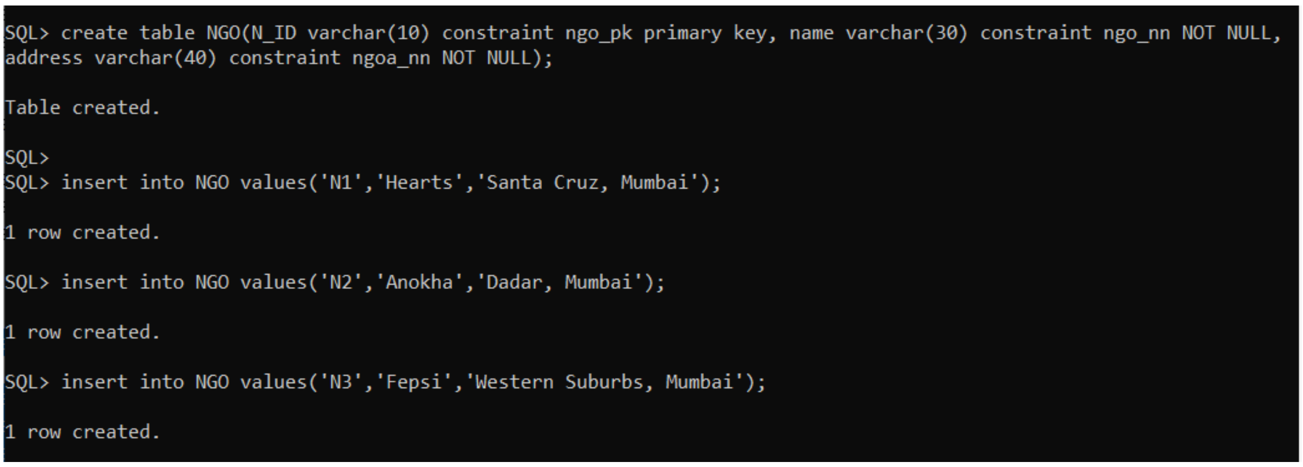


```
SQL> create table AF_Contact (F_ID varchar(10) , contact_details number(10),CONSTRAINT afc_fk
FOREIGN KEY (F_ID)references adoption_families(F_ID),constraint afc_pk primary key(F_ID,contac
t_details));

Table created.

SQL> insert into AF_contact values('F1','8976504312');

1 row created.

SQL> insert into AF_contact values('F2','9012345467');

1 row created.

SQL> insert into AF_contact values('F3','8032456721');

1 row created.
```

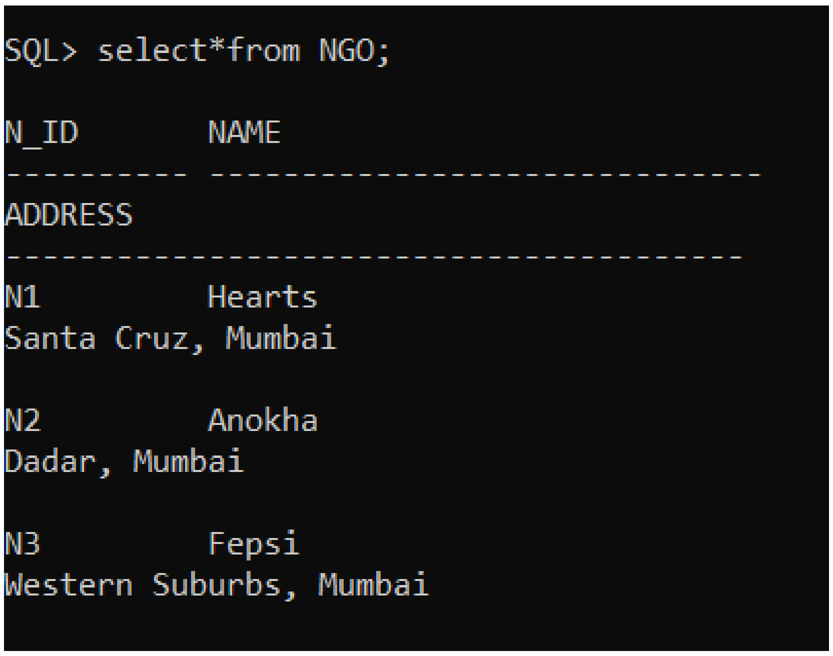


```
SQL> select*from AF_contact;

F_ID       CONTACT_DETAILS
---------- ---------------
F1              8976504312
F2              9012345467
F3              8032456721
```

NGO

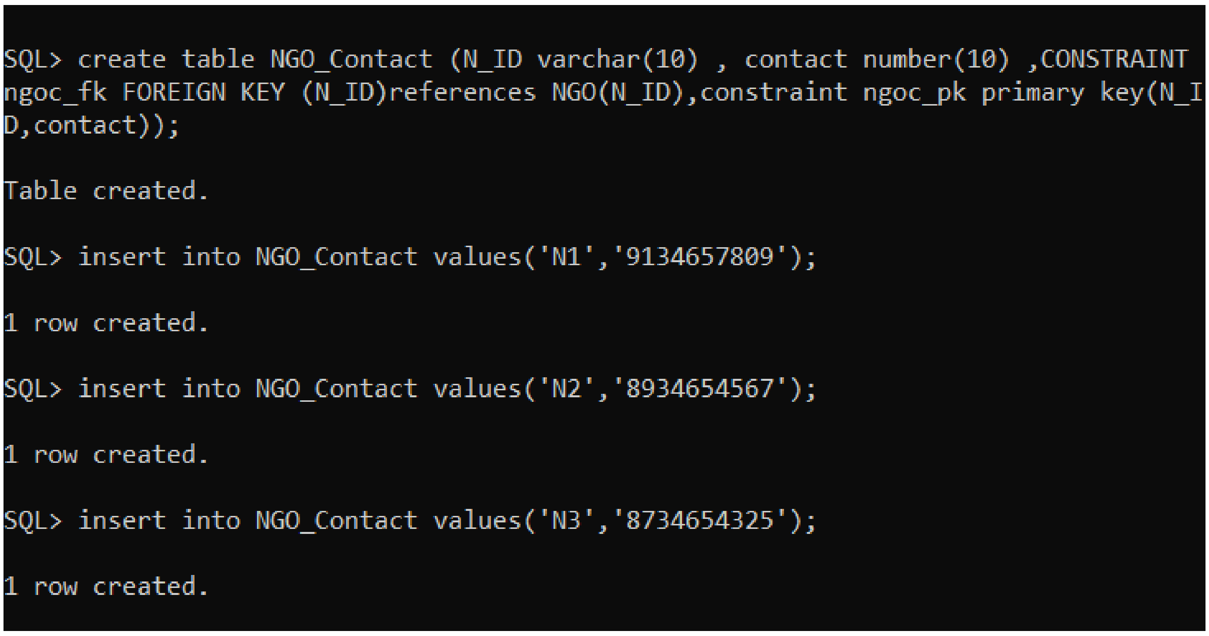


```
SQL> create table NGO(N_ID varchar(10) constraint ngo_pk primary key, name varchar(30) constraint ngo_nn NOT NULL,
address varchar(40) constraint ngoa_nn NOT NULL);

Table created.

SQL>
SQL> insert into NGO values('N1','Hearts','Santa Cruz, Mumbai');

1 row created.

SQL> insert into NGO values('N2','Anokha','Dadar, Mumbai');

1 row created.

SQL> insert into NGO values('N3','Fepsi','Western Suburbs, Mumbai');

1 row created.
```

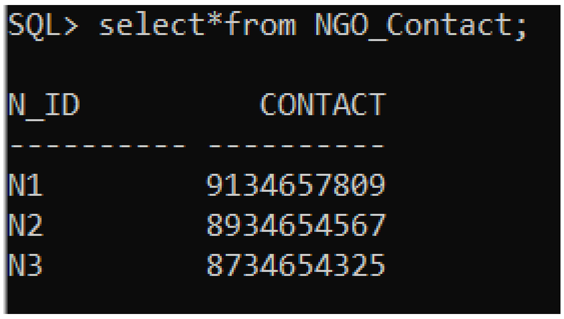


```
SQL> select*from NGO;

N_ID       NAME
---------- ------------------------------
ADDRESS
----------------------------------------
N1         Hearts
Santa Cruz, Mumbai

N2         Anokha
Dadar, Mumbai

N3         Fepsi
Western Suburbs, Mumbai
```

NGO_CONTACT

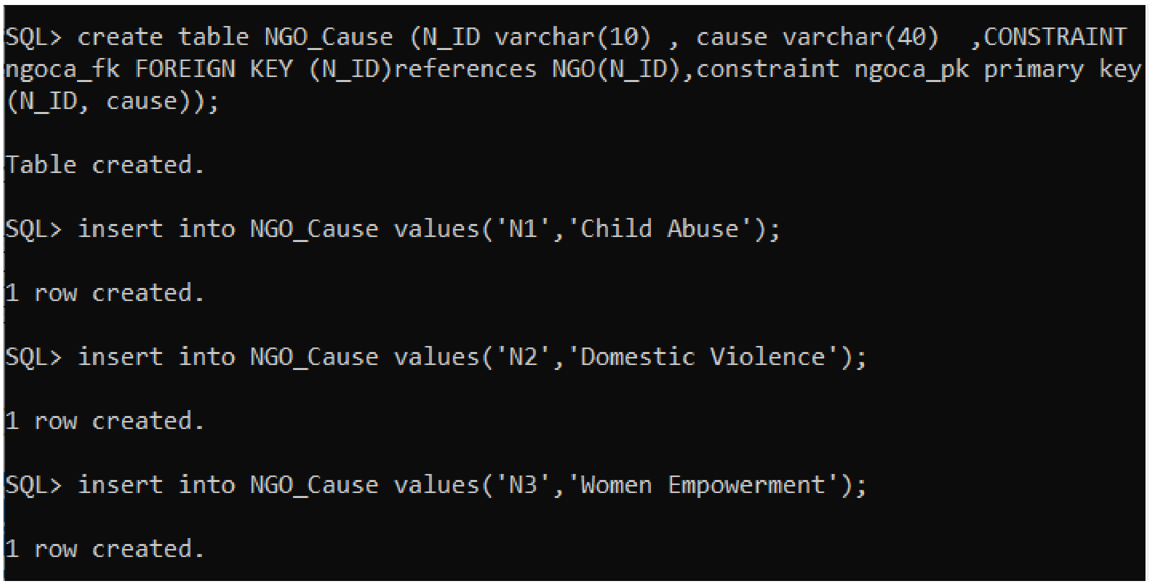


```
SQL> create table NGO_Contact (N_ID varchar(10) , contact number(10) ,CONSTRAINT
ngoc_fk FOREIGN KEY (N_ID)references NGO(N_ID),constraint ngoc_pk primary key(N_I
D,contact));

Table created.

SQL> insert into NGO_Contact values('N1','9134657809');

1 row created.

SQL> insert into NGO_Contact values('N2','8934654567');

1 row created.

SQL> insert into NGO_Contact values('N3','8734654325');

1 row created.
```

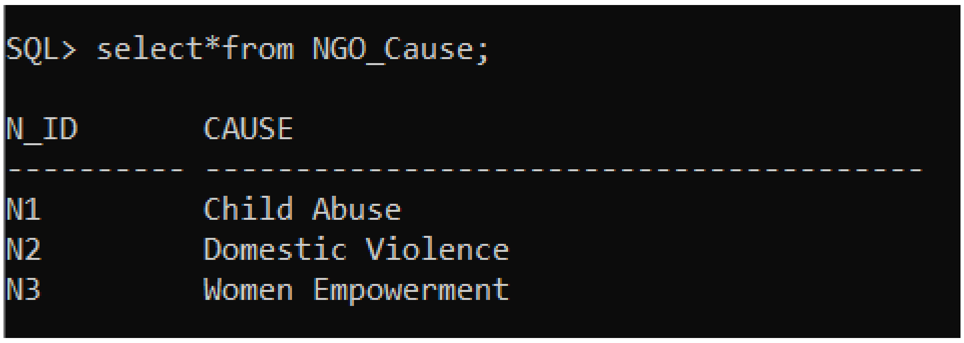


```
SQL> select*from NGO_Contact;

N_ID          CONTACT
---------- ----------
N1         9134657809
N2         8934654567
N3         8734654325
```

NGO_CAUSE

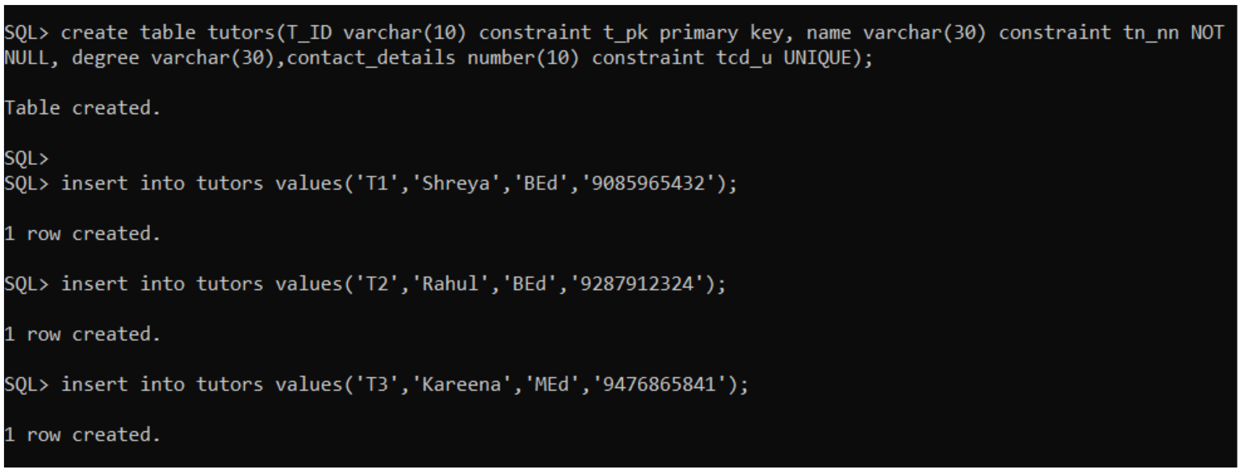


```
SQL> create table NGO_Cause (N_ID varchar(10) , cause varchar(40)  ,CONSTRAINT
ngoca_fk FOREIGN KEY (N_ID)references NGO(N_ID),constraint ngoca_pk primary key
(N_ID, cause));

Table created.

SQL> insert into NGO_Cause values('N1','Child Abuse');

1 row created.

SQL> insert into NGO_Cause values('N2','Domestic Violence');

1 row created.

SQL> insert into NGO_Cause values('N3','Women Empowerment');

1 row created.
```

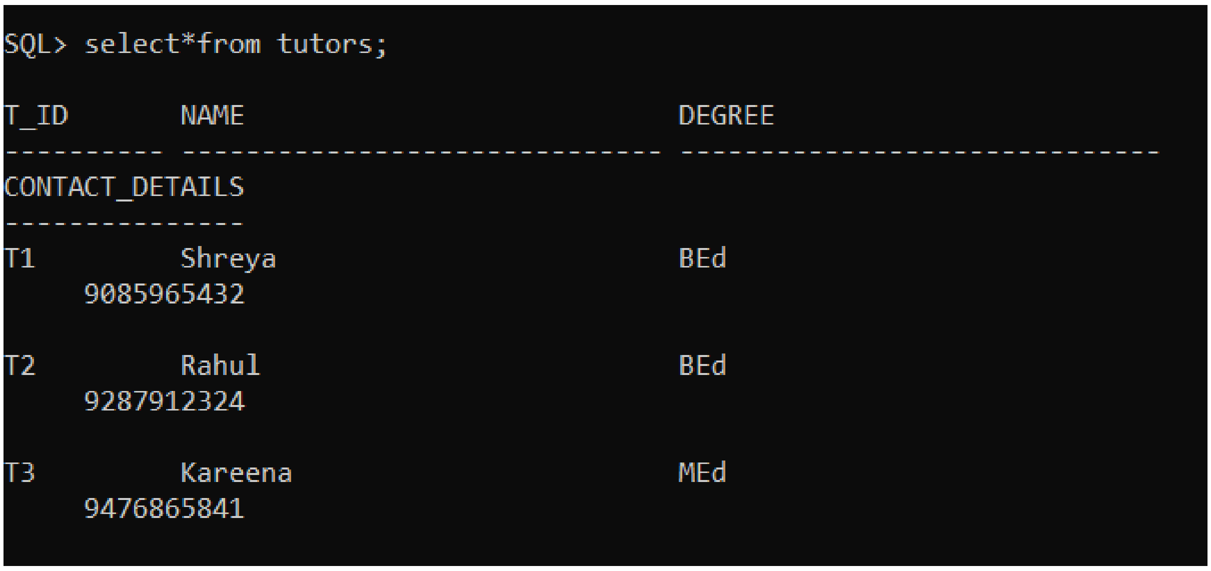


```
SQL> select*from NGO_Cause;

N_ID       CAUSE
---------- ----------------------------------------
N1         Child Abuse
N2         Domestic Violence
N3         Women Empowerment
```

TUTORS

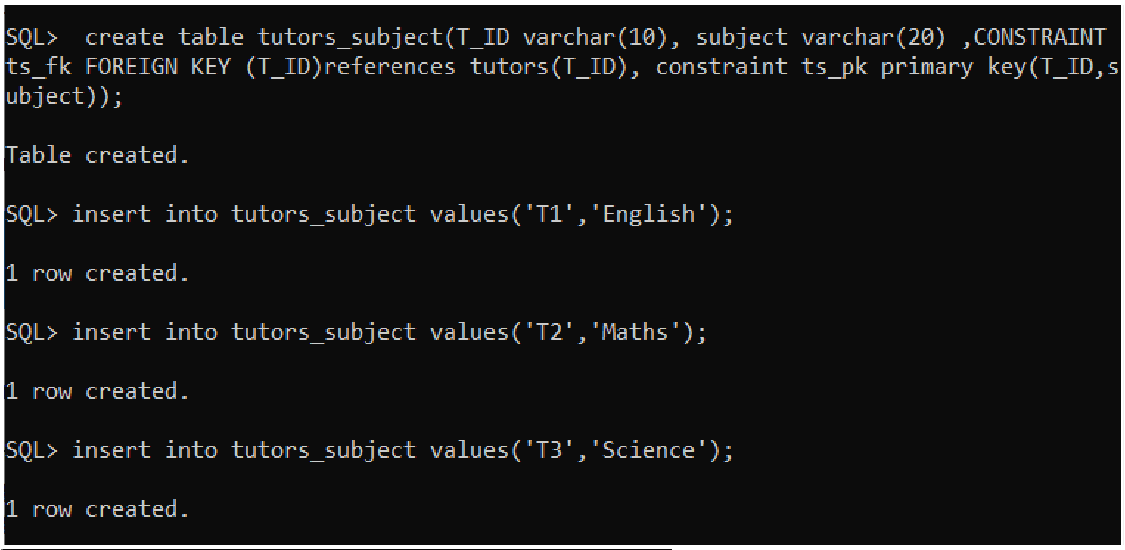


```
SQL> create table tutors(T_ID varchar(10) constraint t_pk primary key, name varchar(30) constraint tn_nn NOT
NULL, degree varchar(30),contact_details number(10) constraint tcd_u UNIQUE);

Table created.

SQL>
SQL> insert into tutors values('T1','Shreya','BEd','9085965432');

1 row created.

SQL> insert into tutors values('T2','Rahul','BEd','9287912324');

1 row created.

SQL> insert into tutors values('T3','Kareena','MEd','9476865841');

1 row created.
```

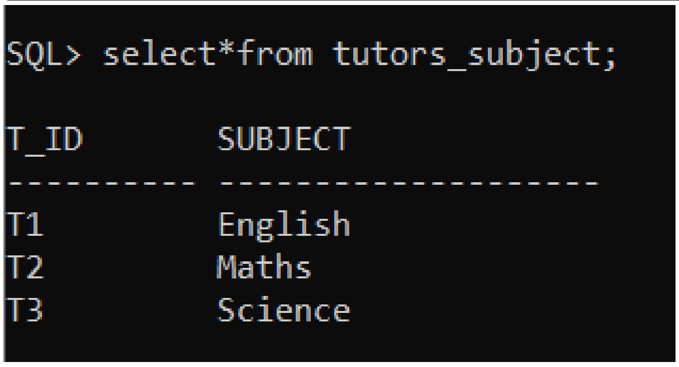


```
SQL> select*from tutors;

T_ID       NAME                           DEGREE
---------- ------------------------------ ------------------------------
CONTACT_DETAILS
---------------
T1         Shreya                         BEd
     9085965432

T2         Rahul                          BEd
     9287912324

T3         Kareena                        MEd
     9476865841
```

TUTORS_SUBJECT

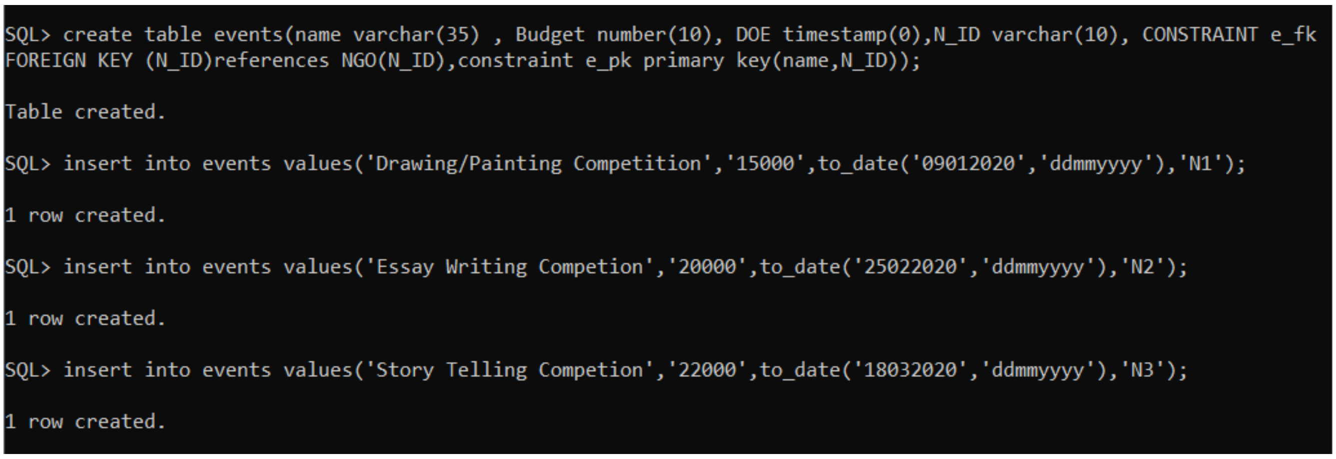


```
SQL>  create table tutors_subject(T_ID varchar(10), subject varchar(20) ,CONSTRAINT
ts_fk FOREIGN KEY (T_ID)references tutors(T_ID), constraint ts_pk primary key(T_ID,s
ubject));

Table created.

SQL> insert into tutors_subject values('T1','English');

1 row created.

SQL> insert into tutors_subject values('T2','Maths');

1 row created.

SQL> insert into tutors_subject values('T3','Science');

1 row created.
```

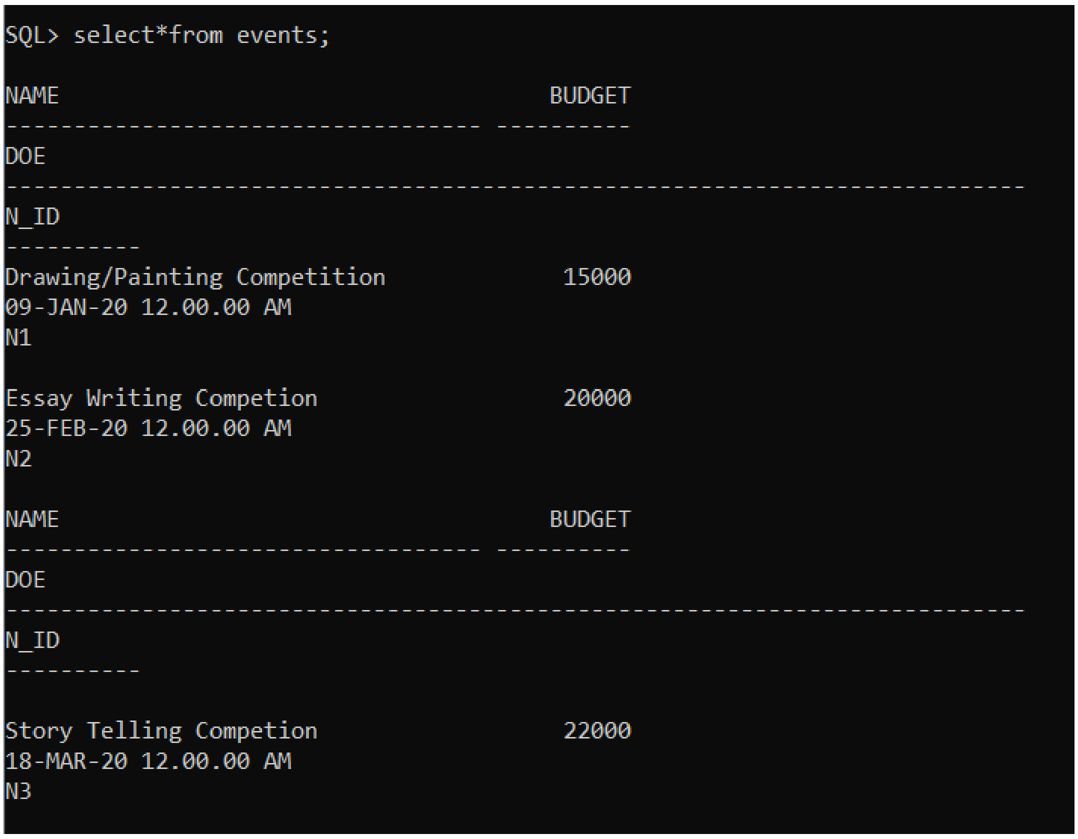


```
SQL> select*from tutors_subject;

T_ID       SUBJECT
---------- --------------------
T1         English
T2         Maths
T3         Science
```

EVENTS

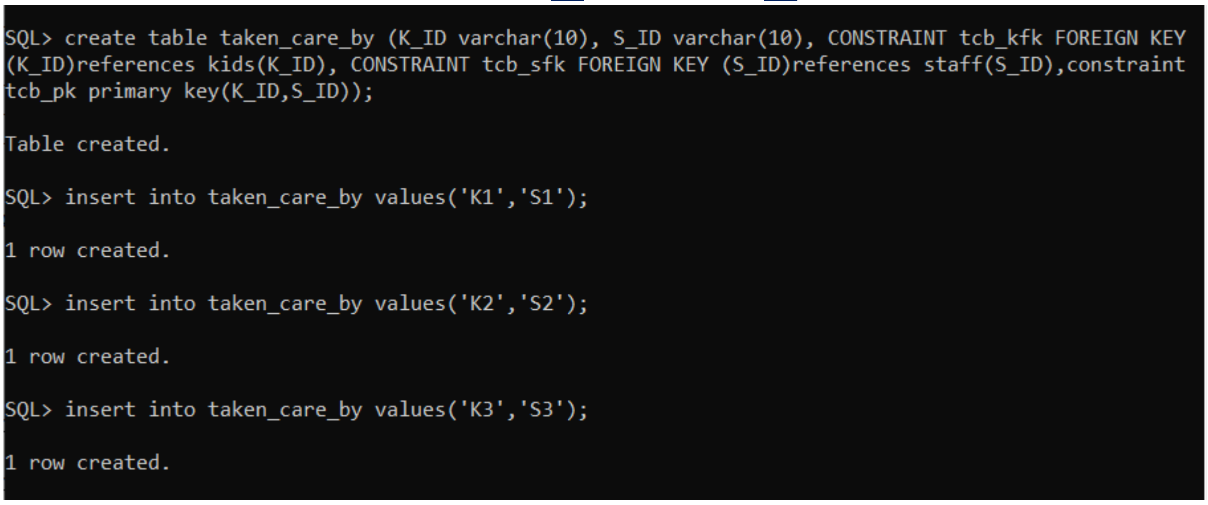


```
SQL> create table events(name varchar(35) , Budget number(10), DOE timestamp(0),N_ID varchar(10), CONSTRAINT e_fk
FOREIGN KEY (N_ID)references NGO(N_ID),constraint e_pk primary key(name,N_ID));

Table created.

SQL> insert into events values('Drawing/Painting Competition','15000',to_date('09012020','ddmmyyyy'),'N1');

1 row created.

SQL> insert into events values('Essay Writing Competion','20000',to_date('25022020','ddmmyyyy'),'N2');

1 row created.

SQL> insert into events values('Story Telling Competion','22000',to_date('18032020','ddmmyyyy'),'N3');

1 row created.
```

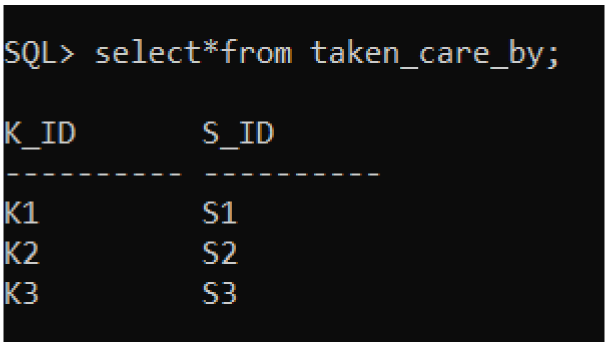


```
SQL> select*from events;

NAME                                    BUDGET
----------------------------------- ----------
DOE
---------------------------------------------------------------------------
N_ID
----------
Drawing/Painting Competition             15000
09-JAN-20 12.00.00 AM
N1

Essay Writing Competion                  20000
25-FEB-20 12.00.00 AM
N2

NAME                                    BUDGET
----------------------------------- ----------
DOE
---------------------------------------------------------------------------
N_ID
----------

Story Telling Competion                  22000
18-MAR-20 12.00.00 AM
N3
```

TAKEN_CARE_BY

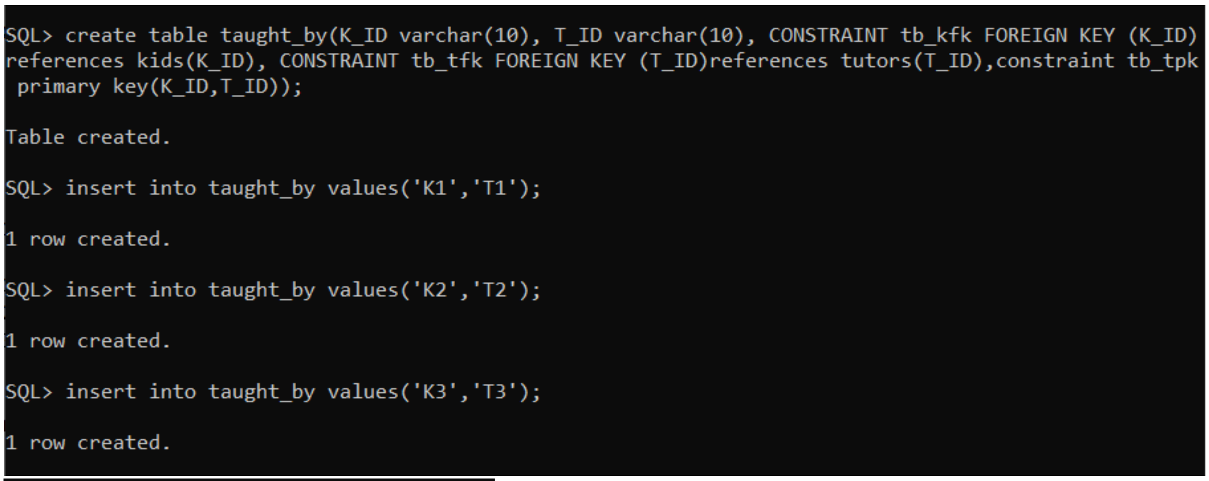


```
SQL> create table taken_care_by (K_ID varchar(10), S_ID varchar(10), CONSTRAINT tcb_kfk FOREIGN KEY
(K_ID)references kids(K_ID), CONSTRAINT tcb_sfk FOREIGN KEY (S_ID)references staff(S_ID),constraint
tcb_pk primary key(K_ID,S_ID));

Table created.

SQL> insert into taken_care_by values('K1','S1');

1 row created.

SQL> insert into taken_care_by values('K2','S2');

1 row created.

SQL> insert into taken_care_by values('K3','S3');

1 row created.
```

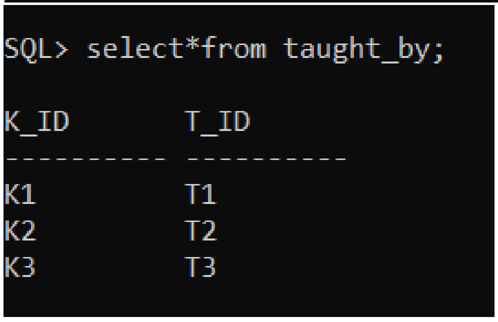


```
SQL> select*from taken_care_by;

K_ID       S_ID
---------- ----------
K1         S1
K2         S2
K3         S3
```

TAUGHT_BY

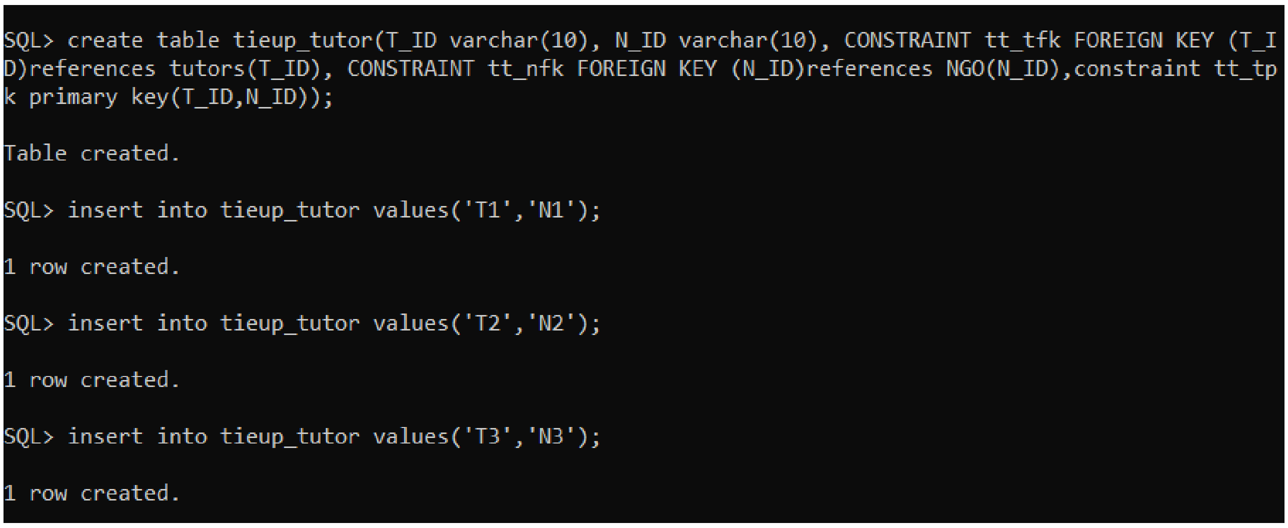


```
SQL> create table taught_by(K_ID varchar(10), T_ID varchar(10), CONSTRAINT tb_kfk FOREIGN KEY (K_ID)
references kids(K_ID), CONSTRAINT tb_tfk FOREIGN KEY (T_ID)references tutors(T_ID),constraint tb_tpk
 primary key(K_ID,T_ID));

Table created.

SQL> insert into taught_by values('K1','T1');

1 row created.

SQL> insert into taught_by values('K2','T2');

1 row created.

SQL> insert into taught_by values('K3','T3');

1 row created.
```

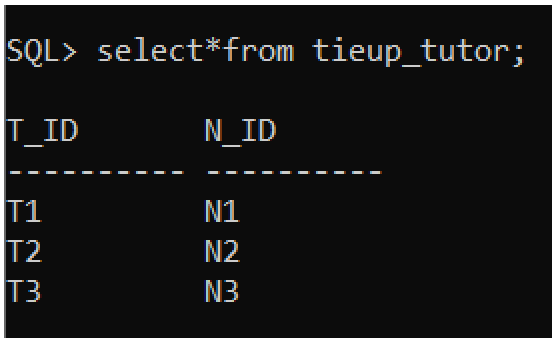


```
SQL> select*from taught_by;

K_ID       T_ID
---------- ----------
K1         T1
K2         T2
K3         T3
```

TIEUP_TUTOR

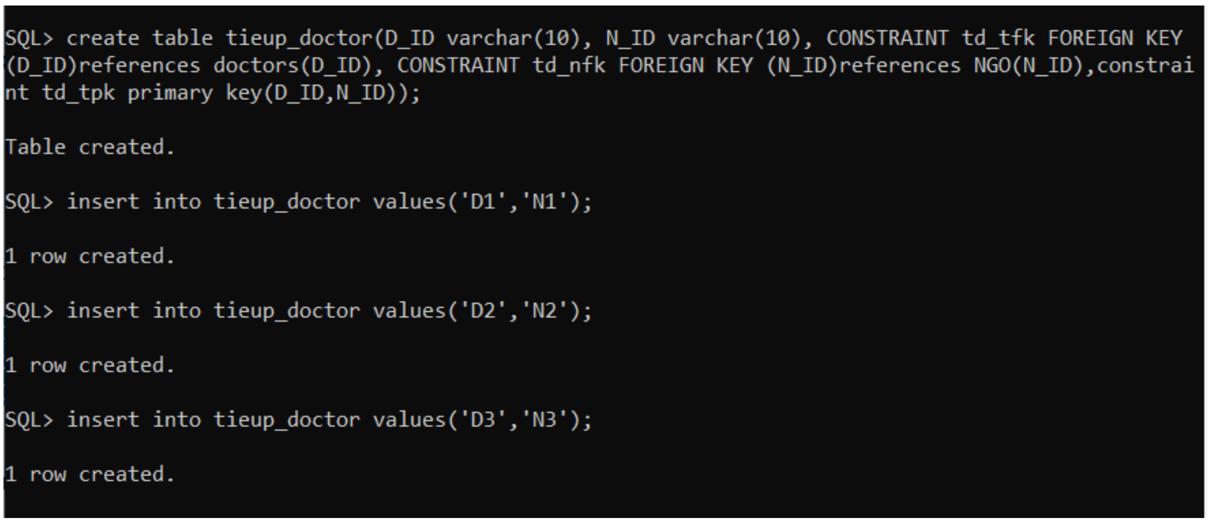


```
SQL> create table tieup_tutor(T_ID varchar(10), N_ID varchar(10), CONSTRAINT tt_tfk FOREIGN KEY (T_I
D)references tutors(T_ID), CONSTRAINT tt_nfk FOREIGN KEY (N_ID)references NGO(N_ID),constraint tt_tp
k primary key(T_ID,N_ID));

Table created.

SQL> insert into tieup_tutor values('T1','N1');

1 row created.

SQL> insert into tieup_tutor values('T2','N2');

1 row created.

SQL> insert into tieup_tutor values('T3','N3');

1 row created.
```

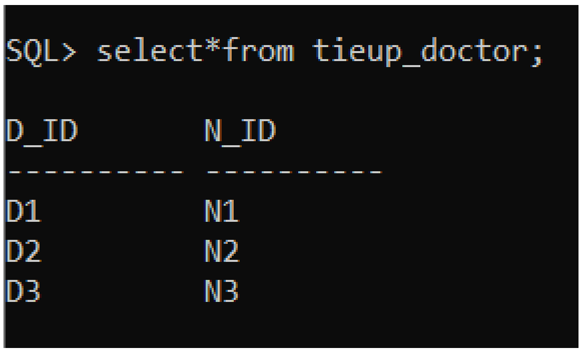


```
SQL> select*from tieup_tutor;

T_ID       N_ID
---------- ----------
T1         N1
T2         N2
T3         N3
```

TIEUP_DOCTOR

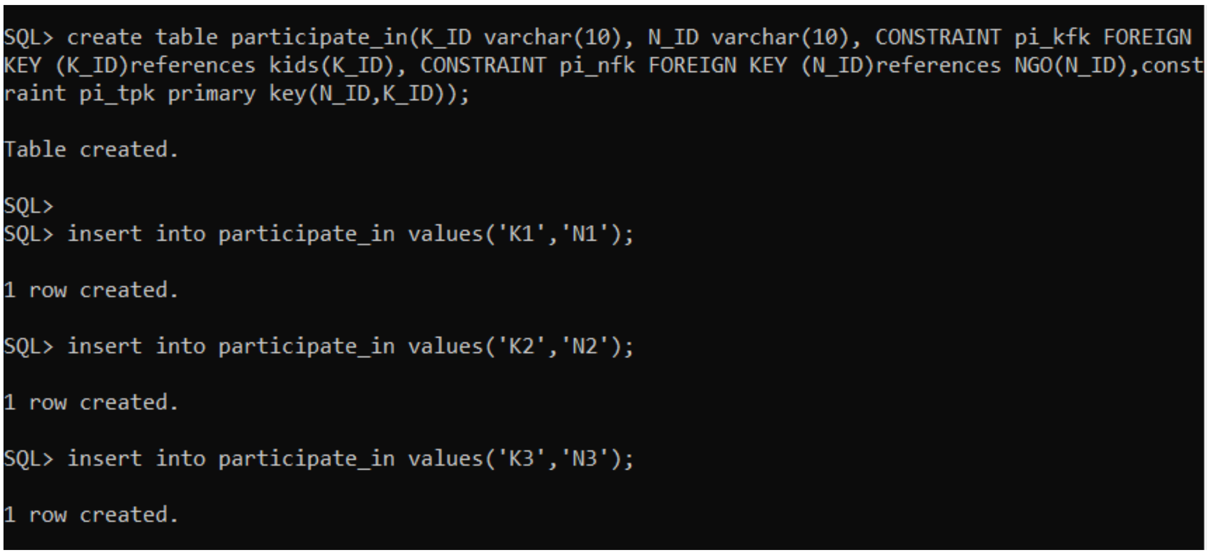


```
SQL> create table tieup_doctor(D_ID varchar(10), N_ID varchar(10), CONSTRAINT td_tfk FOREIGN KEY
(D_ID)references doctors(D_ID), CONSTRAINT td_nfk FOREIGN KEY (N_ID)references NGO(N_ID),constrai
nt td_tpk primary key(D_ID,N_ID));

Table created.

SQL> insert into tieup_doctor values('D1','N1');

1 row created.

SQL> insert into tieup_doctor values('D2','N2');

1 row created.

SQL> insert into tieup_doctor values('D3','N3');

1 row created.
```

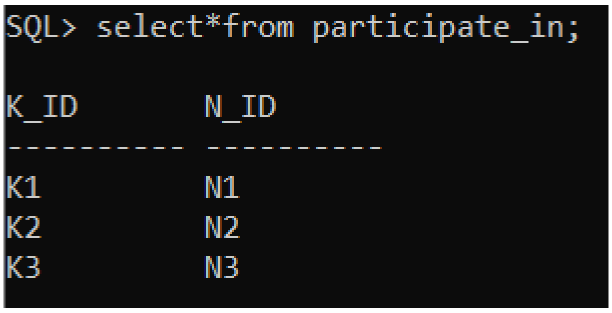


```
SQL> select*from tieup_doctor;

D_ID       N_ID
---------- ----------
D1         N1
D2         N2
D3         N3
```

PARTICIPATE_IN



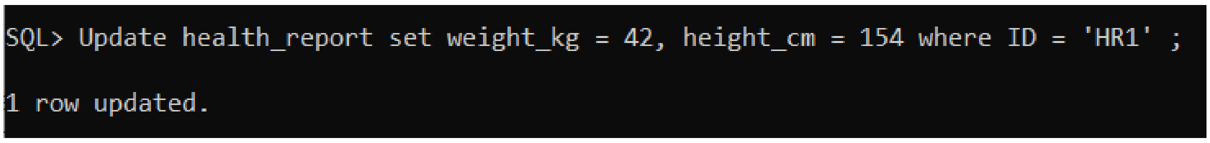
```
SQL> create table participate_in(K_ID varchar(10), N_ID varchar(10), CONSTRAINT pi_kfk FOREIGN
KEY (K_ID)references kids(K_ID), CONSTRAINT pi_nfk FOREIGN KEY (N_ID)references NGO(N_ID),const
raint pi_tpk primary key(N_ID,K_ID));

Table created.

SQL>
SQL> insert into participate_in values('K1','N1');

1 row created.

SQL> insert into participate_in values('K2','N2');

1 row created.

SQL> insert into participate_in values('K3','N3');

1 row created.
```



```
SQL> select*from participate_in;

K_ID       N_ID
---------- ----------
K1         N1
K2         N2
K3         N3
```

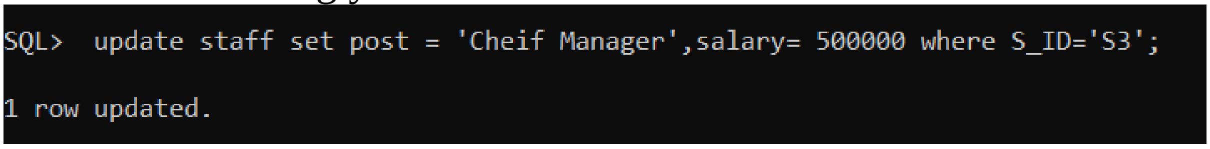
Q6. UPDATE statements according to the functional requirements (refer to review 1 Q2)

1)After the regular checkups the kids health report data (eg. height, weight) needs to be modified if any changes occur.



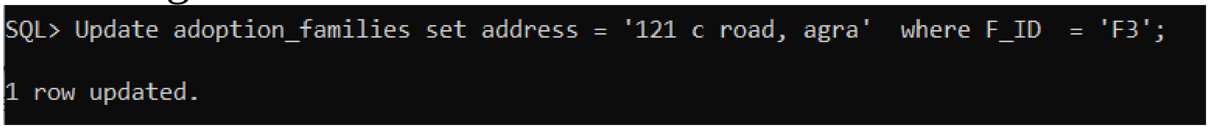
```
SQL> Update health_report set weight_kg = 42, height_cm = 154 where ID = 'HR1' ;

1 row updated.
```

2)If a staff member gets a promotion their post and salary should be modified accordingly



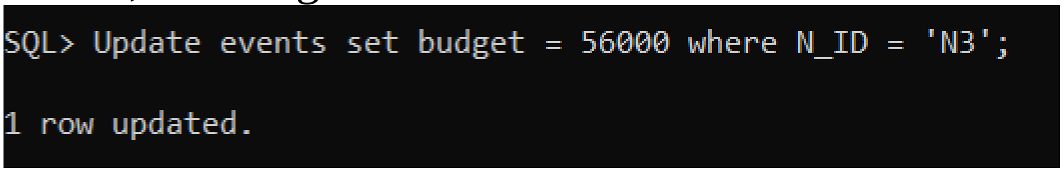
```
SQL>  update staff set post = 'Cheif Manager',salary= 500000 where S_ID='S3';

1 row updated.
```

3)If an adoptive family is relocating to a new address then the following details needs to be modified.



```
SQL> Update adoption_families set address = '121 c road, agra'  where F_ID  = 'F3';

1 row updated.
```

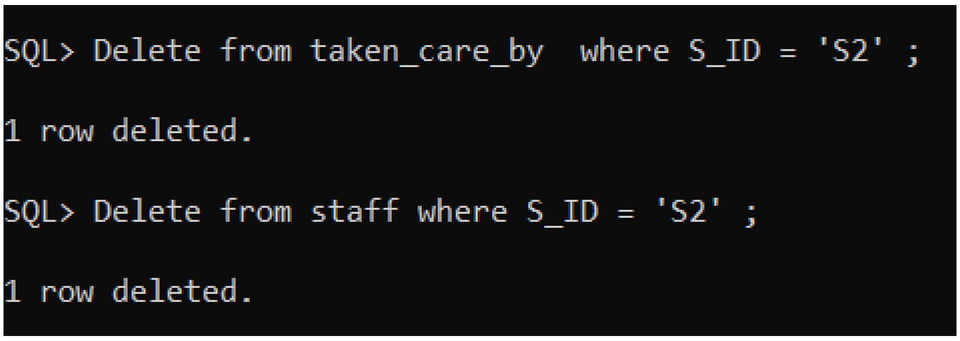
4) If the event budget is changed due to any unforeseen reason, the budget needs to be modified.



```
SQL> Update events set budget = 56000 where N_ID = 'N3';

1 row updated.
```

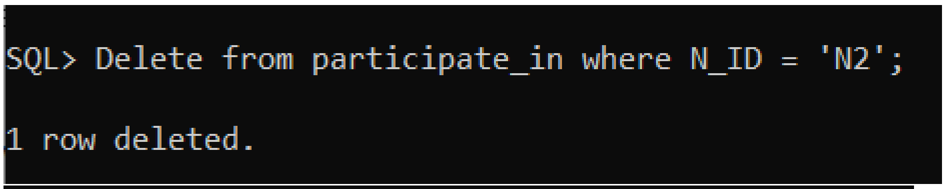
DELETE STATEMENTS

1)If any staff member resigns or is removed, then their data needs to be removed from the database.

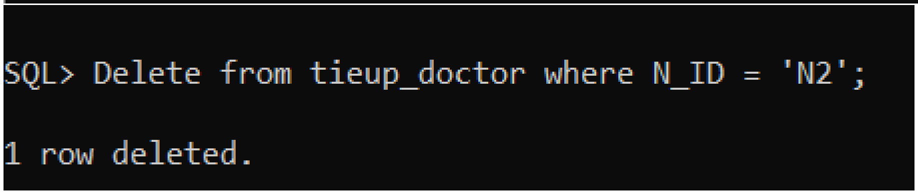


```
SQL> Delete from taken_care_by  where S_ID = 'S2' ;

1 row deleted.

SQL> Delete from staff where S_ID = 'S2' ;

1 row deleted.
```

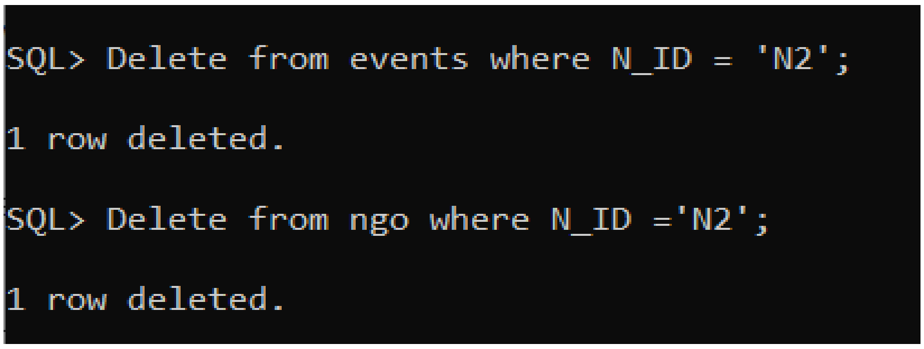
2)If a NGO withdraws their sponsorship from the orphanage, their data needs to be removed.



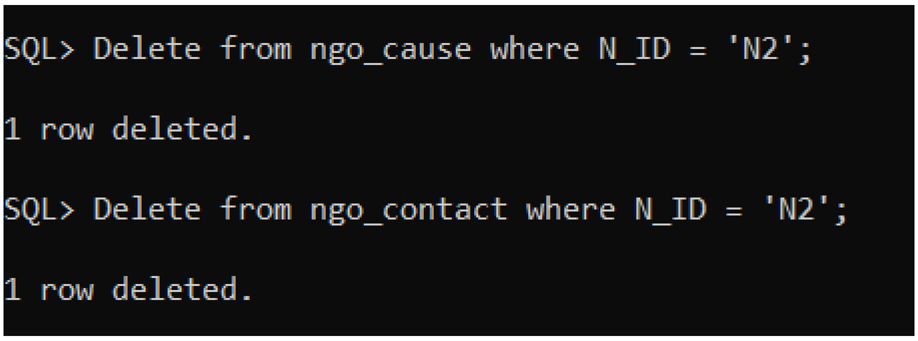
```
SQL> Delete from participate_in where N_ID = 'N2';

1 row deleted.
```



```
SQL> Delete from tieup_doctor where N_ID = 'N2';

1 row deleted.
```

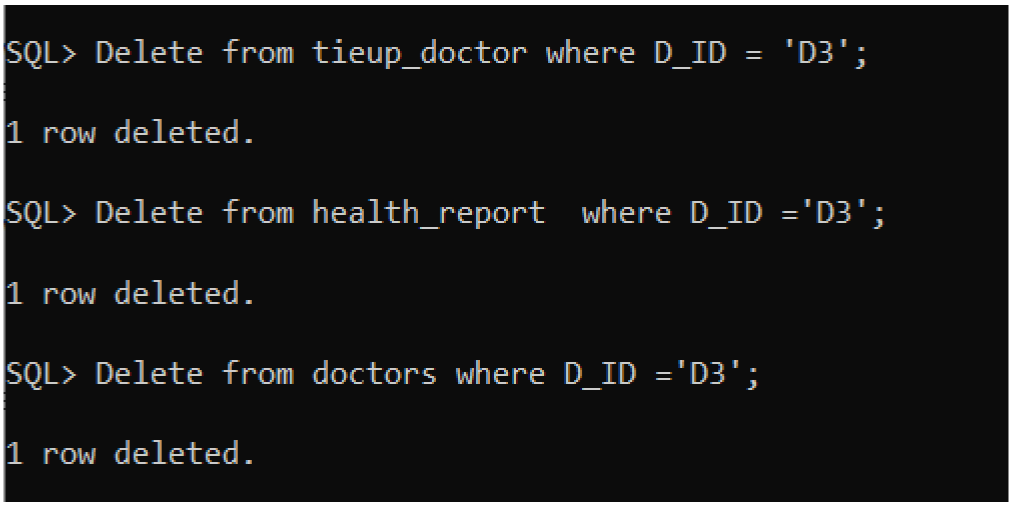


```
SQL> Delete from events where N_ID = 'N2';

1 row deleted.

SQL> Delete from ngo where N_ID ='N2';

1 row deleted.
```



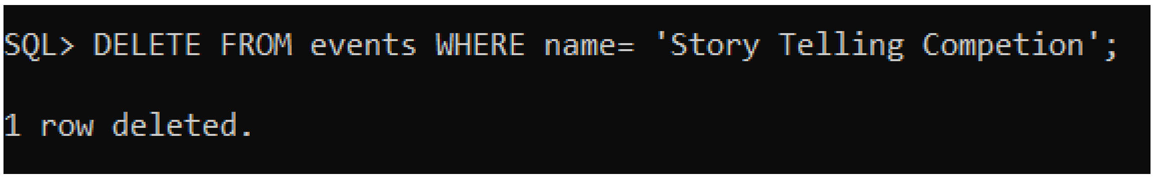
```
SQL> Delete from ngo_cause where N_ID = 'N2';

1 row deleted.

SQL> Delete from ngo_contact where N_ID = 'N2';

1 row deleted.
```

3) If a doctor or a tutor resigns or is removed, then their data needs to be removed from the database (for doctor)



```
SQL> Delete from tieup_doctor where D_ID = 'D3';

1 row deleted.

SQL> Delete from health_report  where D_ID ='D3';

1 row deleted.

SQL> Delete from doctors where D_ID ='D3';

1 row deleted.
```

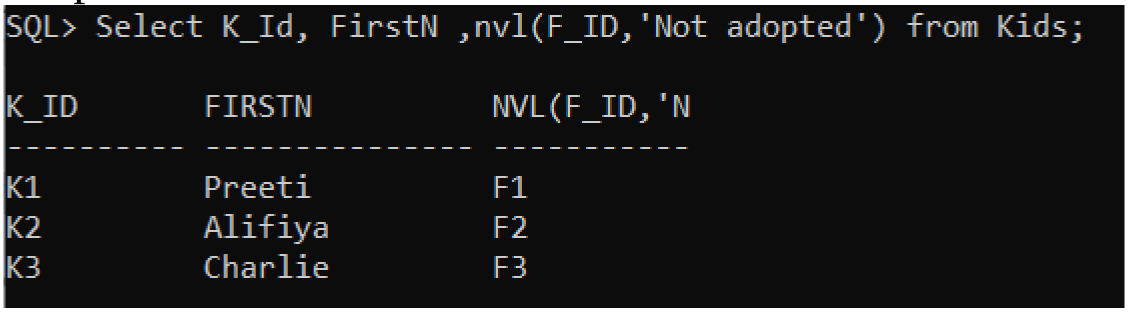
4) If an event that was supposed to be conducted gets cancelled due unforeseen reasons, its details need to be removed.



```
SQL> DELETE FROM events WHERE name= 'Story Telling Competion';

1 row deleted.
```

SELECT STATEMENT

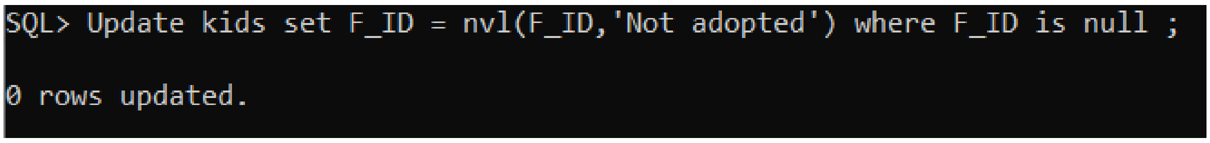
1) nvl
Print K_id, FirstN and not adopted for kids if they arent adopted



```
SQL> Select K_Id, FirstN ,nvl(F_ID,'Not adopted') from Kids;

K_ID       FIRSTN          NVL(F_ID,'N
---------- --------------- -----------
K1         Preeti          F1
K2         Alifiya         F2
K3         Charlie         F3
```

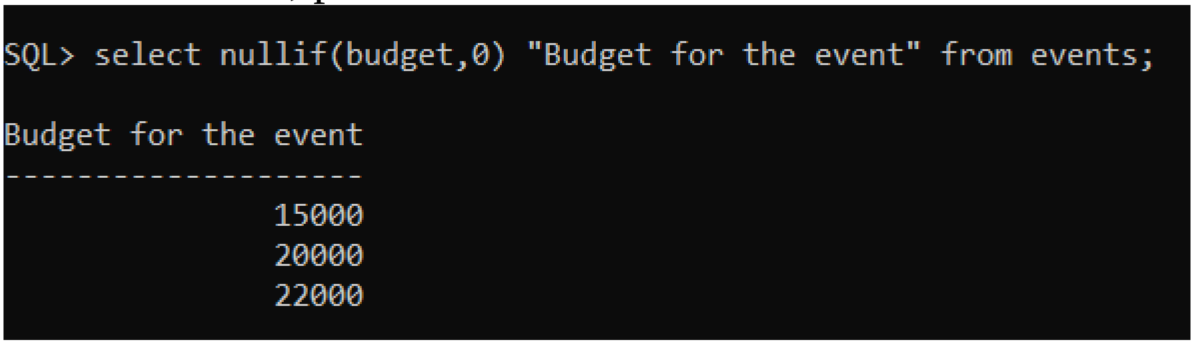
Update

Update those records with NoT adopted in kids where F_ID is null.



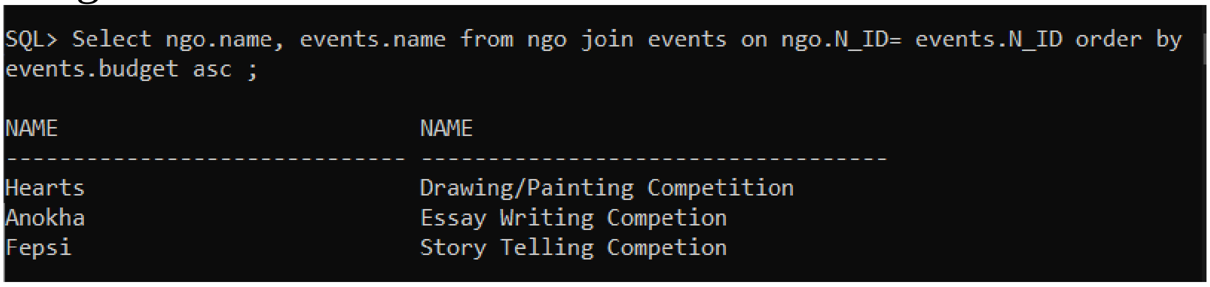
```
SQL> Update kids set F_ID = nvl(F_ID,'Not adopted') where F_ID is null ;

0 rows updated.
```

2)nullif
Display the budget details for the events of all the ngos, if any value is ZERO, print as NULL value.



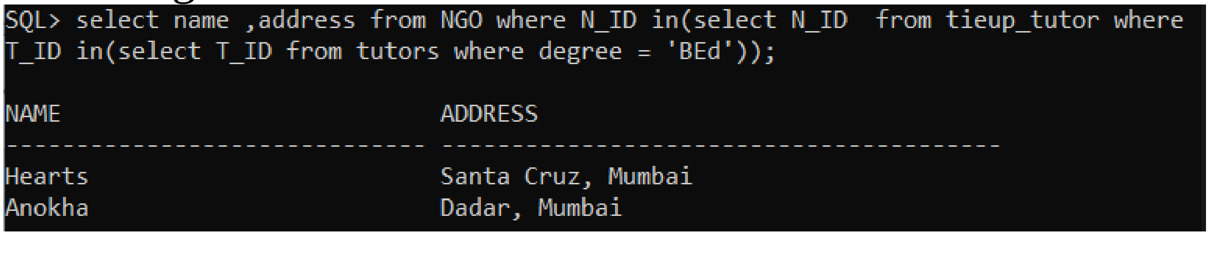
```
SQL> select nullif(budget,0) "Budget for the event" from events;

Budget for the event
--------------------
               15000
               20000
               22000
```

3)one join query order by
Display ngo names event names in the asc order of the event budgets.



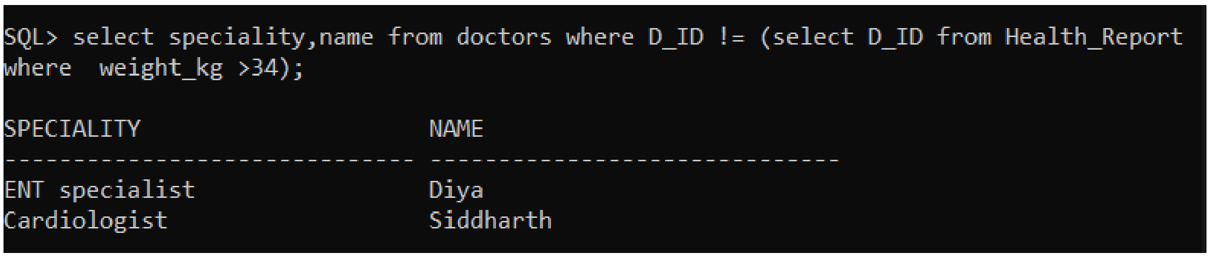
```
SQL> Select ngo.name, events.name from ngo join events on ngo.N_ID= events.N_ID order by
events.budget asc ;

NAME                           NAME
------------------------------ -----------------------------------
Hearts                         Drawing/Painting Competition
Anokha                         Essay Writing Competion
Fepsi                          Story Telling Competion
```

4)one uncorrelated nested query
1) Display the name and address of the ngo where tutors have a Bed degree



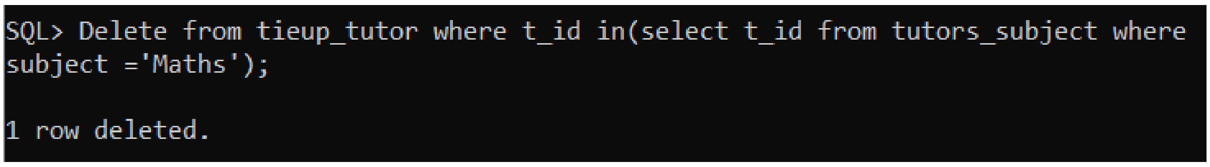
```
SQL> select name ,address from NGO where N_ID in(select N_ID  from tieup_tutor where
T_ID in(select T_ID from tutors where degree = 'BEd'));

NAME                           ADDRESS
------------------------------ ------------------------------------------
Hearts                         Santa Cruz, Mumbai
Anokha                         Dadar, Mumbai
```

2) Display the speciality and name of the doctors where the health report they created has kids weight less than 34.



```
SQL> select speciality,name from doctors where D_ID != (select D_ID from Health_Report
where  weight_kg >34);

SPECIALITY                     NAME
------------------------------ ------------------------------
ENT specialist                 Diya
Cardiologist                   Siddharth
```

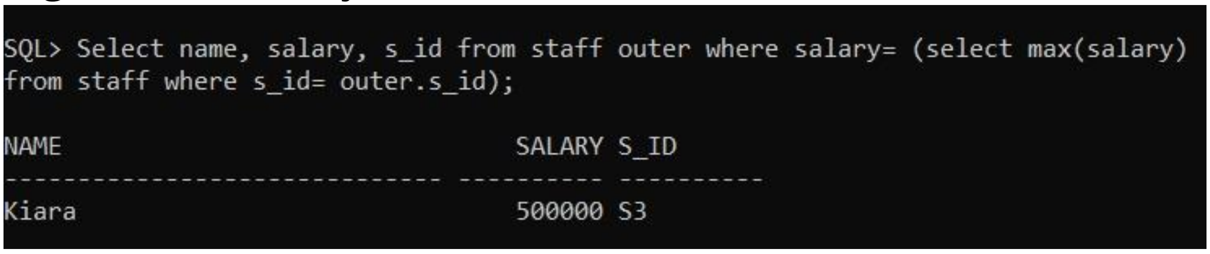
3) Delete
Delete the record of the ngo id and tid where the teacher teaches math.



```
SQL> Delete from tieup_tutor where t_id in(select t_id from tutors_subject where
subject ='Maths');

1 row deleted.
```

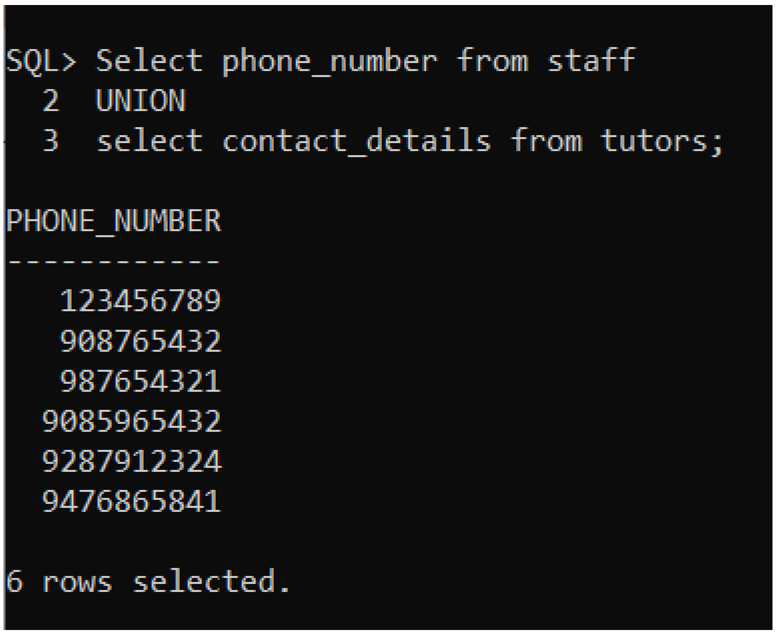
5)One correlated query

Retrieve the staff name, salary and staff id who has the highest/max salary.



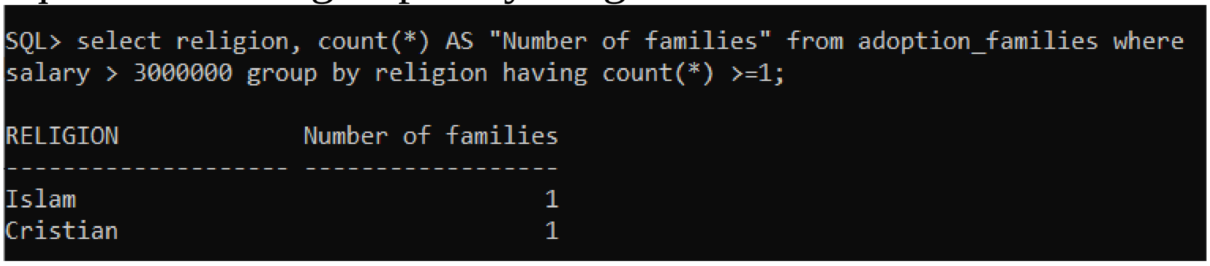
```
SQL> Select name, salary, s_id from staff outer where salary= (select max(salary)
from staff where s_id= outer.s_id);

NAME                               SALARY S_ID
------------------------------ ---------- ----------
Kiara                              500000 S3
```

6) one set operation
Retrieve all the contact details of tutors and staff.



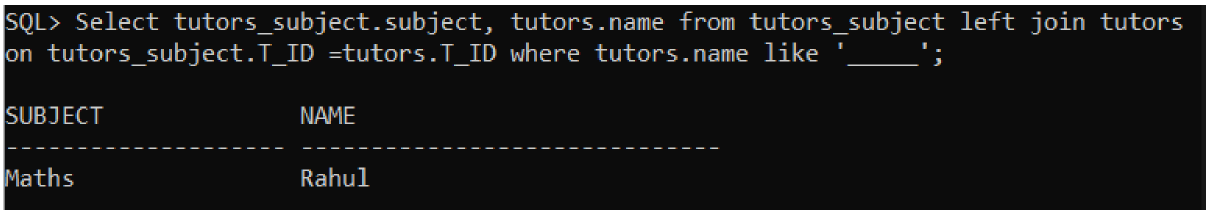
```
SQL> Select phone_number from staff
  2  UNION
  3  select contact_details from tutors;

PHONE_NUMBER
------------
   123456789
   908765432
   987654321
  9085965432
  9287912324
  9476865841

6 rows selected.
```

7) one group by having where
Display the religion, number of families where the salary is higher than 300000 and the no. Of families is greater than or equal to one all grouped by religion.



```
SQL> select religion, count(*) AS "Number of families" from adoption_families where
salary > 3000000 group by religion having count(*) >=1;

RELIGION             Number of families
-------------------- ------------------
Islam                                 1
Cristian                              1
```

8) left/right/outer join
Display the subjects and tutors name where the tutors name is of 5 letters.

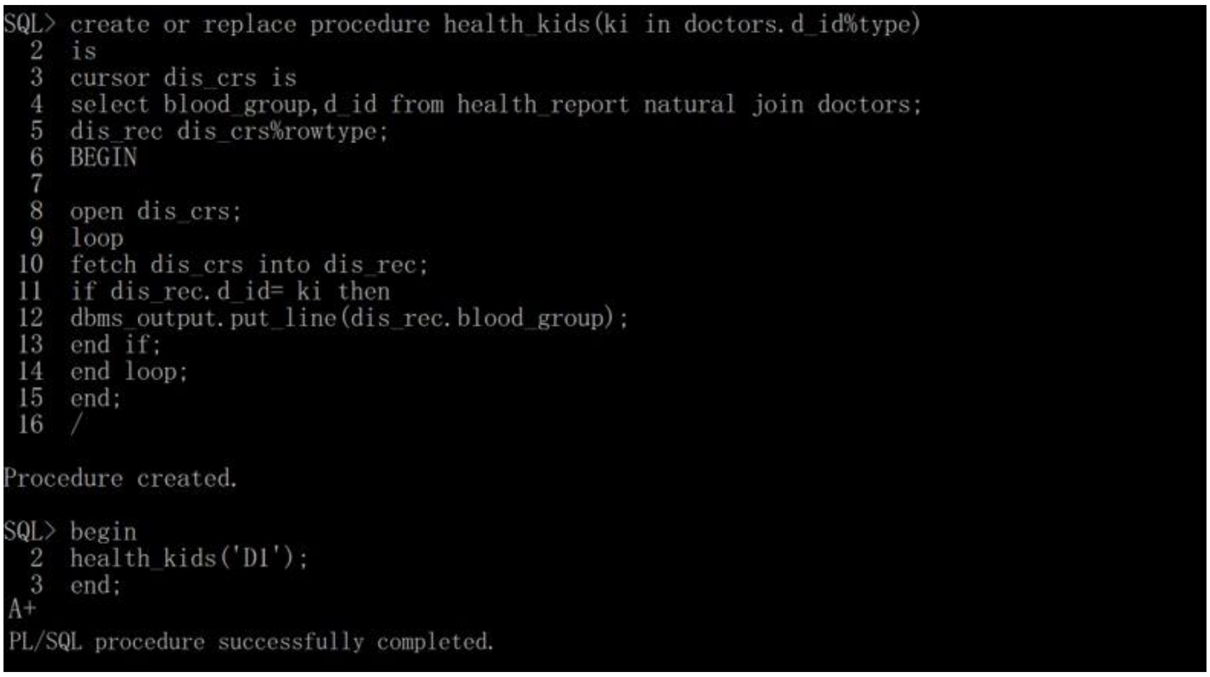


```
SQL> Select tutors_subject.subject, tutors.name from tutors_subject left join tutors
on tutors_subject.T_ID =tutors.T_ID where tutors.name like '_____';

SUBJECT              NAME
-------------------- ------------------------------
Maths                Rahul
```

Q7

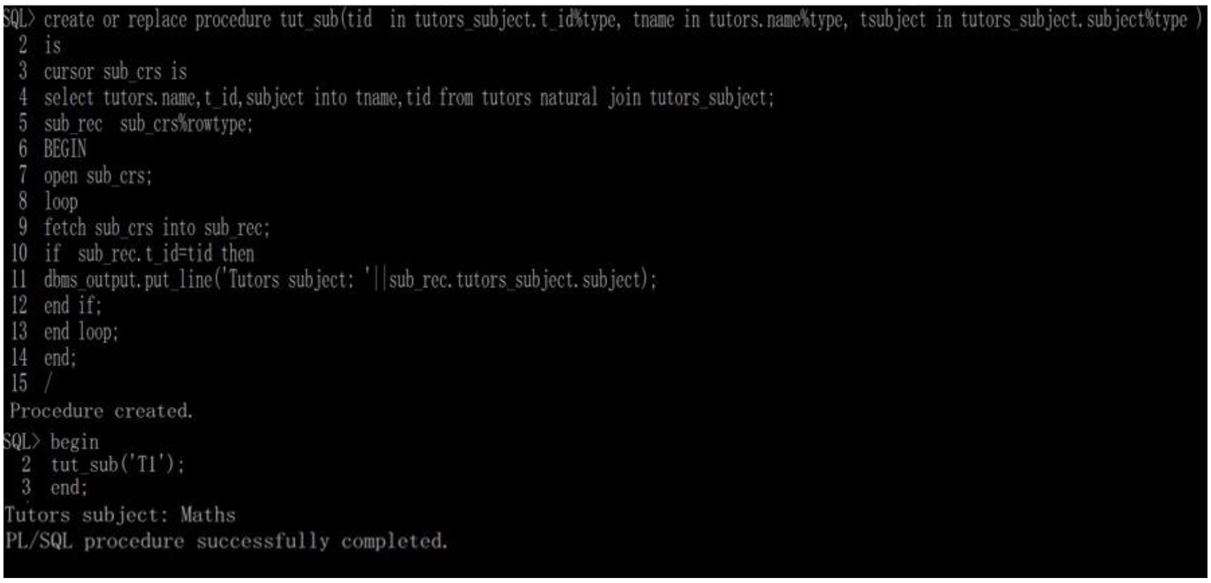
Procedure with cursor

1. When the user types the doctor id, it directs to the report they have made and the blood group in that report is printed.



```
SQL> create or replace procedure health_kids(ki in doctors.d_id%type)
is
cursor dis_crs is
select blood_group,d_id from health_report natural join doctors;
dis_rec dis_crs%rowtype;
BEGIN

open dis_crs;
loop
fetch dis_crs into dis_rec;
if dis_rec.d_id= ki then
dbms_output.put_line(dis_rec.blood_group);
end if;
end loop;
end;
/

Procedure created.

SQL> begin
health_kids('D1');
end;
A+
PL/SQL procedure successfully completed.
```

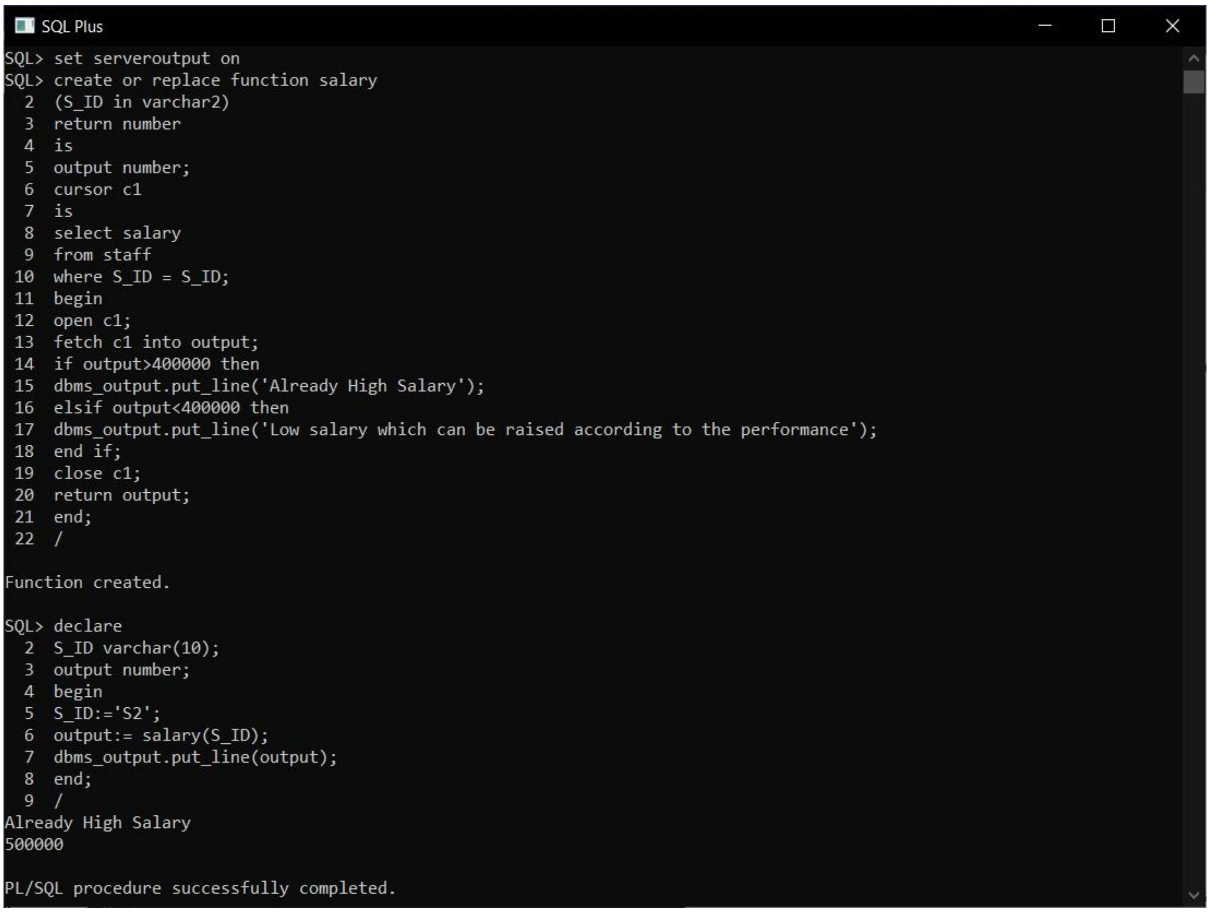
2. User enters the id of the user and based on the subject the tutor teaches is displayed.



```
SQL> create or replace procedure tut_sub(tid  in tutors_subject.t_id%type, tname in tutors.name%type, tsubject in tutors_subject.subject%type )
is
cursor sub_crs is
select tutors.name,t_id,subject into tname,tid from tutors natural join tutors_subject;
sub_rec  sub_crs%rowtype;
BEGIN
open sub_crs;
loop
fetch sub_crs into sub_rec;
if  sub_rec.t_id=tid then
dbms_output.put_line('Tutors subject: '||sub_rec.tutors_subject.subject);
end if;
end loop;
end;
/
Procedure created.
SQL> begin
tut_sub('T1');
end;
Tutors subject: Maths
PL/SQL procedure successfully completed.
```

Function with cursor

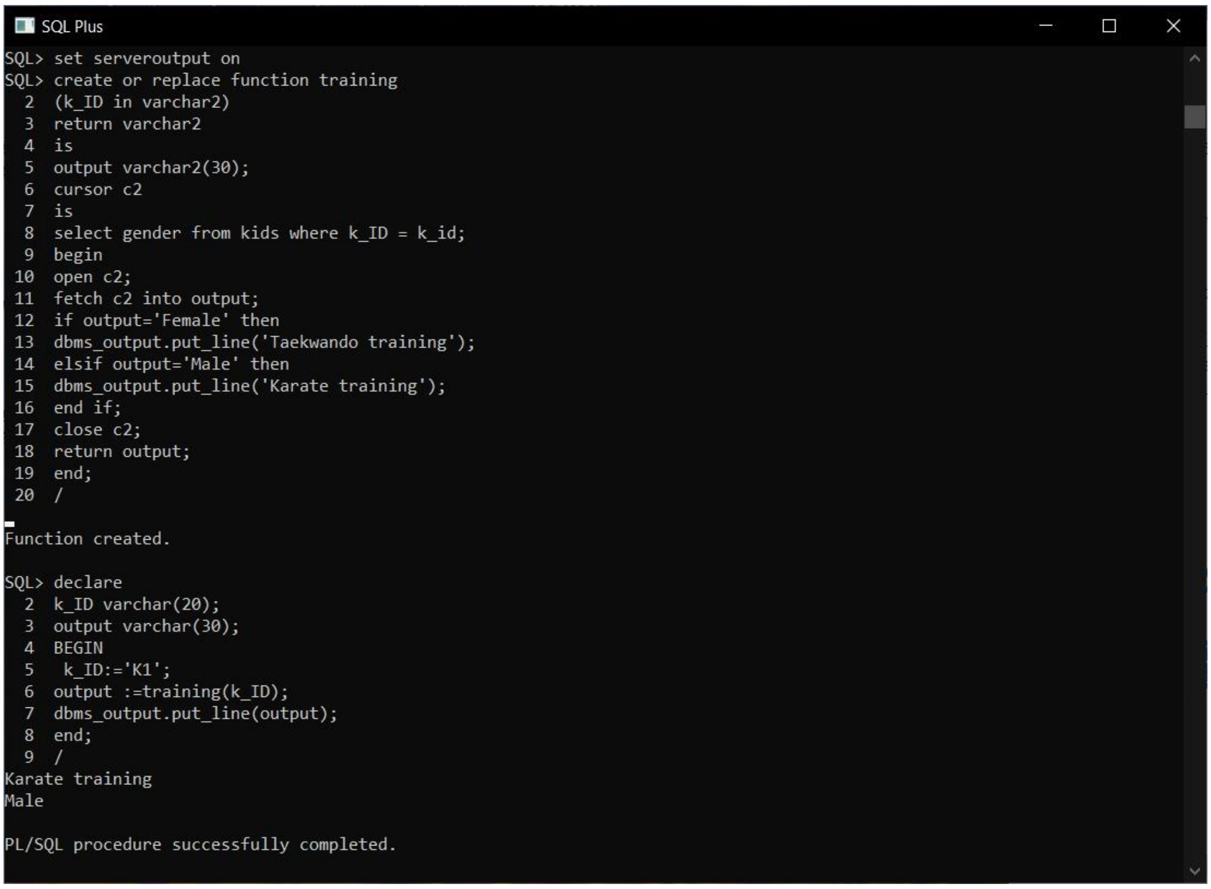
1.This function is created to display whether a staff member already has high salary (greater than 400000) or low salary (less than 400000). If he/she has a low salary then it can be incremented on the basis of their performance.



```
SQL> set serveroutput on
SQL> create or replace function salary
(S_ID in varchar2)
return number
is
output number;
cursor c1
is
select salary
from staff
where S_ID = S_ID;
begin
open c1;
fetch c1 into output;
if output>400000 then
dbms_output.put_line('Already High Salary');
elsif output<400000 then
dbms_output.put_line('Low salary which can be raised according to the performance');
end if;
close c1;
return output;
end;
/

Function created.

SQL> declare
S_ID varchar(10);
output number;
begin
S_ID:='S2';
output:= salary(S_ID);
dbms_output.put_line(output);
end;
/
Already High Salary
500000

PL/SQL procedure successfully completed.
```

2.User enters the Id of the kid and based on the gender of the kid they are assigned for martial arts training.

If a girl- Taekwondo training and if a boy- Karate training.

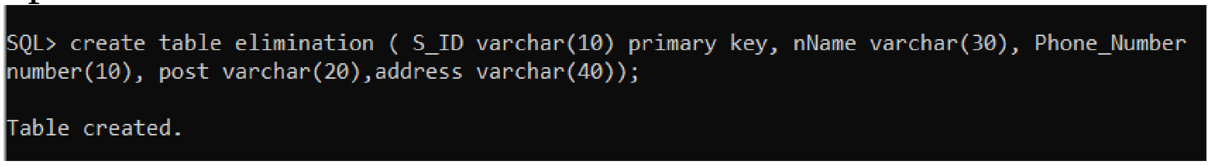


```
SQL> set serveroutput on
SQL> create or replace function training
(k_ID in varchar2)
return varchar2
is
output varchar2(30);
cursor c2
is
select gender from kids where k_ID = k_id;
begin
open c2;
fetch c2 into output;
if output='Female' then
dbms_output.put_line('Taekwando training');
elsif output='Male' then
dbms_output.put_line('Karate training');
end if;
close c2;
return output;
end;
/

Function created.

SQL> declare
k_ID varchar(20);
output varchar(30);
BEGIN
 k_ID:='K1';
output :=training(k_ID);
dbms_output.put_line(output);
end;
/
Karate training
Male

PL/SQL procedure successfully completed.
```

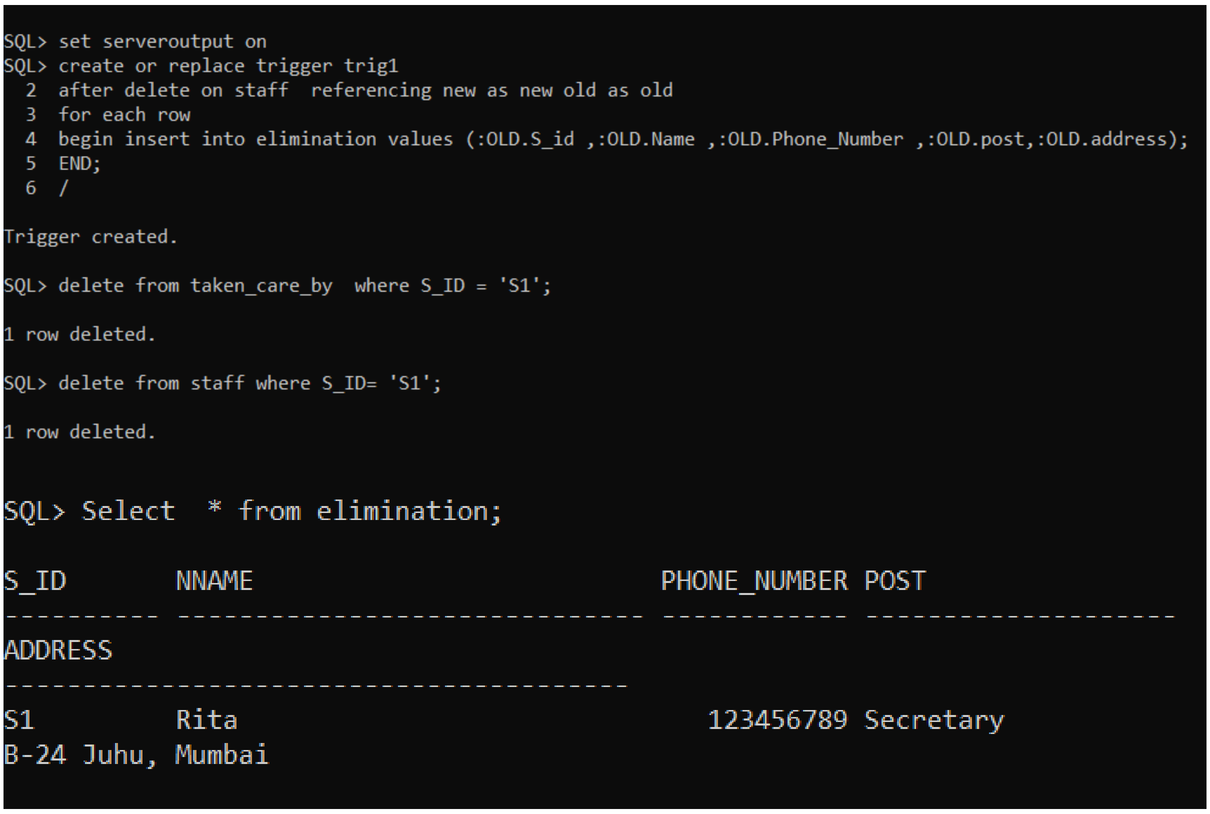
Q8

Trigger 1

Q) When a staff leaves the job do the necessary process and update the elimination table.



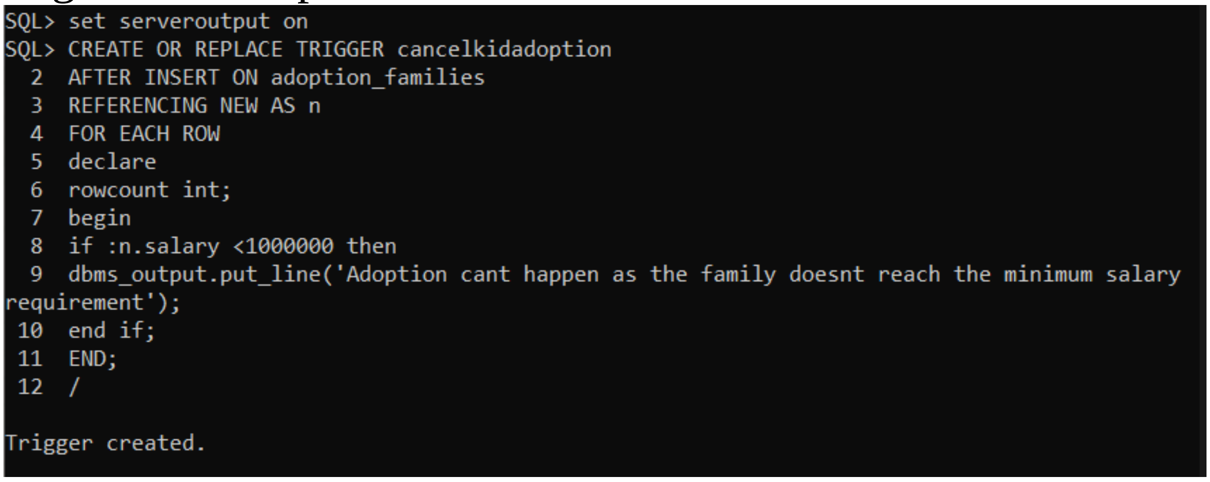
```
SQL> create table elimination ( S_ID varchar(10) primary key, nName varchar(30), Phone_Number
number(10), post varchar(20),address varchar(40));

Table created.
```



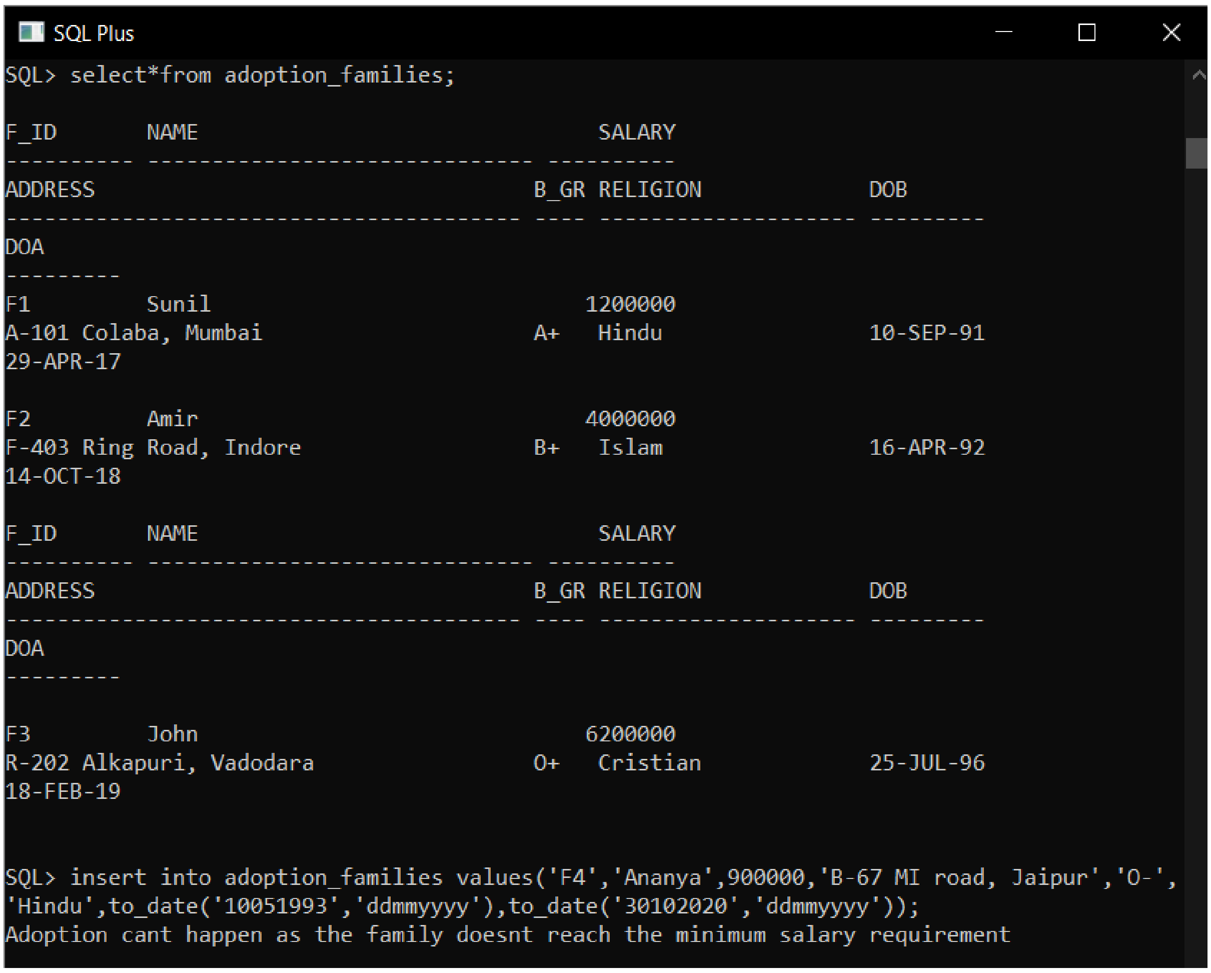
```
SQL> set serveroutput on
SQL> create or replace trigger trig1
  2  after delete on staff  referencing new as new old as old
  3  for each row
  4  begin insert into elimination values (:OLD.S_id ,:OLD.Name ,:OLD.Phone_Number ,:OLD.post,:OLD.address);
  5  END;
  6  /

Trigger created.

SQL> delete from taken_care_by  where S_ID = 'S1';

1 row deleted.

SQL> delete from staff where S_ID= 'S1';

1 row deleted.

SQL> Select  * from elimination;

S_ID       NNAME                          PHONE_NUMBER POST
---------- ------------------------------ ------------ --------------------
ADDRESS
----------------------------------------
S1         Rita                              123456789 Secretary
B-24 Juhu, Mumbai
```

Trigger 2

Q) While inserting a record if the adoption families salary is less than 10,00,000 than record cant be inserted as they arent eligible for adoption.



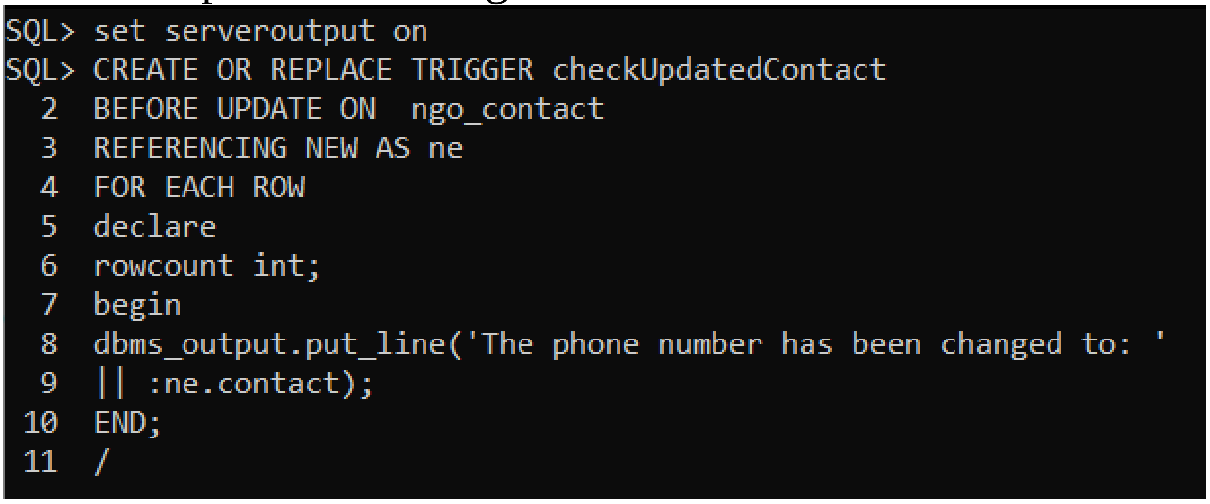
```
SQL> set serveroutput on
SQL> CREATE OR REPLACE TRIGGER cancelkidadoption
  2  AFTER INSERT ON adoption_families
  3  REFERENCING NEW AS n
  4  FOR EACH ROW
  5  declare
  6  rowcount int;
  7  begin
  8  if :n.salary <1000000 then
  9  dbms_output.put_line('Adoption cant happen as the family doesnt reach the minimum salary
requirement');
 10  end if;
 11  END;
 12  /

Trigger created.
```



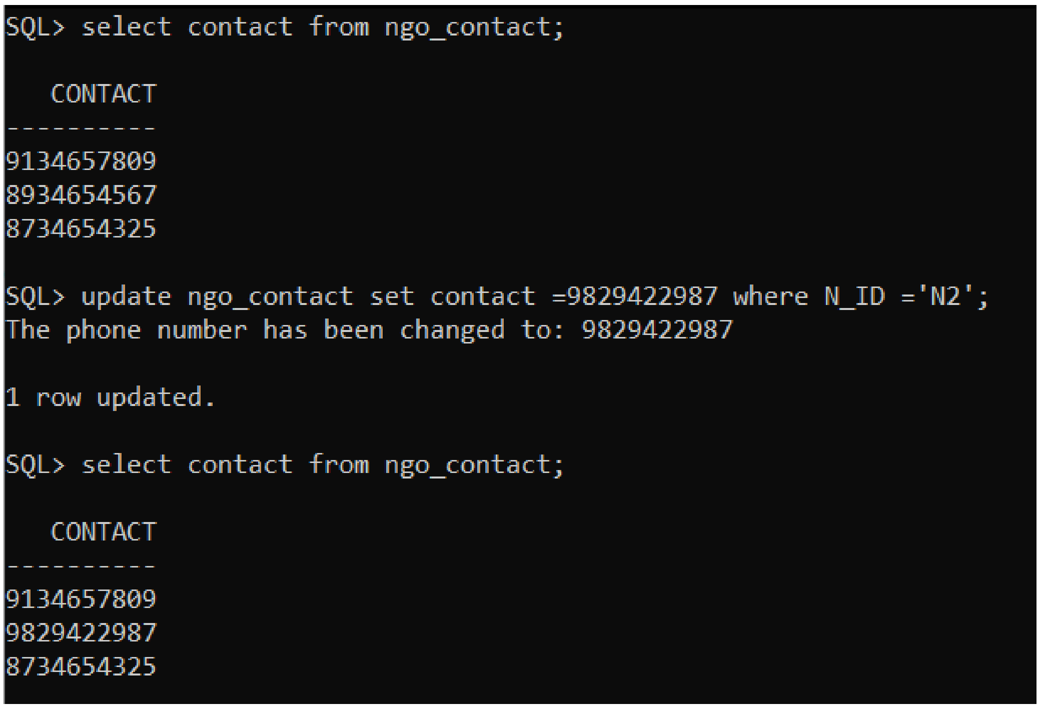
```
SQL> select*from adoption_families;

F_ID       NAME                           SALARY
---------- ------------------------------ ----------
ADDRESS                                  B_GR RELIGION             DOB
---------------------------------------- ---- -------------------- ---------
DOA
---------
F1         Sunil                             1200000
A-101 Colaba, Mumbai                     A+   Hindu                10-SEP-91
29-APR-17

F2         Amir                              4000000
F-403 Ring Road, Indore                  B+   Islam                16-APR-92
14-OCT-18

F_ID       NAME                           SALARY
---------- ------------------------------ ----------
ADDRESS                                  B_GR RELIGION             DOB
---------------------------------------- ---- -------------------- ---------
DOA
---------

F3         John                              6200000
R-202 Alkapuri, Vadodara                 O+   Cristian             25-JUL-96
18-FEB-19


SQL> insert into adoption_families values('F4','Ananya',900000,'B-67 MI road, Jaipur','O-',
'Hindu',to_date('10051993','ddmmyyyy'),to_date('30102020','ddmmyyyy'));
Adoption cant happen as the family doesnt reach the minimum salary requirement
```

Trigger 3

Q)Give Notification to Admin of contact change of a NGO when its updated in the ngo_contact table.



```
SQL> set serveroutput on
SQL> CREATE OR REPLACE TRIGGER checkUpdatedContact
  2  BEFORE UPDATE ON  ngo_contact
  3  REFERENCING NEW AS ne
  4  FOR EACH ROW
  5  declare
  6  rowcount int;
  7  begin
  8  dbms_output.put_line('The phone number has been changed to: '
  9  || :ne.contact);
 10  END;
 11  /
```



```
SQL> select contact from ngo_contact;

   CONTACT
----------
9134657809
8934654567
8734654325

SQL> update ngo_contact set contact =9829422987 where N_ID ='N2';
The phone number has been changed to: 9829422987

1 row updated.

SQL> select contact from ngo_contact;

   CONTACT
----------
9134657809
9829422987
8734654325
```