Final Project Document IS 620

Health Insurance Management System

(Group 3)

Group Members

Afsha Shaikh
Kritesh Arora
Neel Patel
Smit Vasani

Instructor

Dr. Zhiyuan Chen

Table of Content

1.Drop Table and Sequence Statements

```
drop table PREMIUM;
drop table PREMIUM_LEVELS;
drop table MESSAGE;
drop table CLAIM;
drop table CLAIM_LINE;
drop table SERVICE_PROVIDER;
drop table PLAN;
drop table COVERAGE;
drop table POLICY;
drop table POLICY_DEPENDENT;
drop table SERVICE;
drop table DEPENDENT;
drop table CUSTOMER;
drop table USERLOGIN;
drop table USERTYPE;


drop sequence CLAIMID_SEQ;
drop sequence CUSTID_SEQ;
drop sequence DID_SEQ;
drop sequence MESSAGE_SEQ;
drop sequence MESSAGEID_SEQ;
drop sequence POLICY_SEQ;
drop sequence SPID_SEQ;
drop sequence USERID_SEQ;
```

2.Create Table Statement

```
CREATE TABLE "USERTYPE"
   (        "UT_ID" NUMBER(*,0) NOT NULL ENABLE,
            "USER_TYPE" VARCHAR2(50 BYTE),
             PRIMARY KEY ("UT_ID")
);

-----------------------------------------------------------------
```

```
CREATE TABLE "USERLOGIN"
  (      "USERID" NUMBER(*,0) NOT NULL ENABLE,
         "EMAIL_ID" VARCHAR2(255 BYTE),
         "PSWD" VARCHAR2(255 BYTE),
         "UT_ID" NUMBER(*,0) NOT NULL ENABLE,
          PRIMARY KEY ("USERID"),
          FOREIGN KEY ("UT_ID")
           REFERENCES "USERTYPE" ("UT_ID") ENABLE
  );

------------------------------------------------------------------------

CREATE TABLE "CUSTOMER"
  (      "CUST_ID" NUMBER(*,0) NOT NULL ENABLE,
         "CUST_NAME" VARCHAR2(255 BYTE) NOT NULL ENABLE,
         "EMAIL_ID" VARCHAR2(255 BYTE) NOT NULL ENABLE,
         "PASSWORD" VARCHAR2(50 BYTE) NOT NULL ENABLE,
         "DOB" DATE NOT NULL ENABLE,
         "GENDER" VARCHAR2(10 BYTE) NOT NULL ENABLE,
         "ADDRESS" VARCHAR2(255 BYTE) NOT NULL ENABLE,
         "PHONE_NO" VARCHAR2(15 BYTE),
         "USERID" NUMBER(*,0) NOT NULL ENABLE,
          PRIMARY KEY ("CUST_ID"),
          FOREIGN KEY ("USERID")
           REFERENCES "USERLOGIN" ("USERID") ENABLE
  );

------------------------------------------------------------------------

CREATE TABLE "DEPENDENT"
  (      "D_ID" NUMBER(*,0) NOT NULL ENABLE,
         "D_NAME" VARCHAR2(255 BYTE) NOT NULL ENABLE,
         "EMAIL_ID" VARCHAR2(255 BYTE) NOT NULL ENABLE,
         "PASSWORD" VARCHAR2(50 BYTE) NOT NULL ENABLE,
         "DOB" DATE NOT NULL ENABLE,
         "GENDER" VARCHAR2(10 BYTE) NOT NULL ENABLE,
         "ADDRESS" VARCHAR2(255 BYTE) NOT NULL ENABLE,
         "PHONE_NO" VARCHAR2(15 BYTE),
         "USERID" NUMBER(*,0) NOT NULL ENABLE,
         "RELATION" VARCHAR2(50 BYTE) NOT NULL ENABLE,
         "CUST_ID" NUMBER(*,0) NOT NULL ENABLE,
          PRIMARY KEY ("D_ID"),
          FOREIGN KEY ("USERID")
           REFERENCES "USERLOGIN" ("USERID") ENABLE,
            FOREIGN KEY ("CUST_ID")
             REFERENCES "CUSTOMER" ("CUST_ID") ENABLE
  );

--------------------------------------------------------------------------------

CREATE TABLE "SERVICE"
  (        "SERVICE_ID" NUMBER(*,0) NOT NULL ENABLE,
           "SERVICE_DESCRIPTION" VARCHAR2(255 BYTE) NOT NULL ENABLE,
            PRIMARY KEY ("SERVICE_ID")
);

-------------------------------------------------------------------------------

CREATE TABLE "COVERAGE"
  (        "COVERAGE_ID" NUMBER(*,0) NOT NULL ENABLE,
           "MAX_SERVICE_PERYEAR" NUMBER(*,0) NOT NULL ENABLE,
           "ALLOWED_SERVICE_CHARGES" NUMBER(*,0),
           "IN_NETWORK_COPAY" NUMBER(*,0),
           "OUT_NETWORK_COPAY" NUMBER(*,0),
           "IN_NETWORK_COINSURANCE" NUMBER(*,0),
           "OUT_NETWORK_COINSURANCE" NUMBER(*,0),
           "SERVICE_ID" NUMBER(*,0) NOT NULL ENABLE,
            PRIMARY KEY ("COVERAGE_ID", "SERVICE_ID"),
            FOREIGN KEY ("SERVICE_ID")
             REFERENCES "SERVICE" ("SERVICE_ID") ENABLE
  );
---------------------------------------------------------------------------------------
CREATE TABLE "PLAN"
  (        "PLAN_ID" NUMBER(*,0) NOT NULL ENABLE,
           "PLAN_NAME" VARCHAR2(255 BYTE) NOT NULL ENABLE,
           "PLAN_START_YEAR" DATE NOT NULL ENABLE,
           "PLAN_TENURE_MONTH" NUMBER(*,0) NOT NULL ENABLE,
           "DEDUCTABLE_AMOUNT" NUMBER(*,0),
           "MAX_OPC_PERMEMBER" NUMBER(*,0) NOT NULL ENABLE,
           "MAX_OPC_PERFAMILY" NUMBER(*,0) NOT NULL ENABLE,
           "STANDARD_ANNUAL_RATE" NUMBER(*,0) NOT NULL ENABLE,
           "COVERAGE_ID" NUMBER(*,0) NOT NULL ENABLE,
           "SERVICE_ID" NUMBER(*,0) NOT NULL ENABLE,
           "PLAN_END_DATE" DATE,
            PRIMARY KEY ("PLAN_ID", "COVERAGE_ID"),
            FOREIGN KEY ("COVERAGE_ID", "SERVICE_ID")
             REFERENCES "COVERAGE" ("COVERAGE_ID", "SERVICE_ID") ENABLE
  );
```

```
-----------------------------------------------------------------------------------------------

CREATE TABLE "POLICY"
  (       "POLICY_ID" NUMBER(*,0) NOT NULL ENABLE,
          "PLAN_ID" NUMBER(*,0) NOT NULL ENABLE,
          "COVERAGE_ID" NUMBER(*,0) NOT NULL ENABLE,
          "SERVICE_ID" NUMBER(*,0) NOT NULL ENABLE,
          "USER_ID" NUMBER(*,0) NOT NULL ENABLE,
           PRIMARY KEY ("POLICY_ID"),
           FOREIGN KEY ("USER_ID")
            REFERENCES "USERLOGIN" ("USERID") ENABLE,
           FOREIGN KEY ("PLAN_ID", "COVERAGE_ID")
            REFERENCES "PLAN" ("PLAN_ID", "COVERAGE_ID") ENABLE
  );
----------------------------------------------------------------------------------------

CREATE TABLE "POLICY_DEPENDENT"
  (       "POLICY_ID" NUMBER NOT NULL ENABLE,
          "D_ID" NUMBER NOT NULL ENABLE,
          "USERID" NUMBER NOT NULL ENABLE,
           PRIMARY KEY ("POLICY_ID", "D_ID"),
           FOREIGN KEY ("POLICY_ID")
            REFERENCES "POLICY" ("POLICY_ID") ENABLE,
           FOREIGN KEY ("USERID")
            REFERENCES "USERLOGIN" ("USERID") ENABLE,
           FOREIGN KEY ("D_ID")
            REFERENCES "DEPENDENT" ("D_ID") ENABLE
  );

------------------------------------------------------------------------------------------

CREATE TABLE "SERVICE_PROVIDER"
  (       "SP_ID" NUMBER(*,0) NOT NULL ENABLE,
          "SP_DESCRIPTION" VARCHAR2(255 BYTE),
          "SERVICE_ID" NUMBER(*,0) NOT NULL ENABLE,
          "USERID" NUMBER(*,0) NOT NULL ENABLE,
          "EMAIL_ID" VARCHAR2(255 BYTE) NOT NULL ENABLE,
          "PASSWORD" VARCHAR2(50 BYTE) NOT NULL ENABLE,
          "ADDRESS" VARCHAR2(255 BYTE) NOT NULL ENABLE,
          "PHONE_NO" VARCHAR2(15 BYTE),
          "SP_TYPE" VARCHAR2(50 BYTE) NOT NULL ENABLE,
           PRIMARY KEY ("SP_ID", "SERVICE_ID"),
           FOREIGN KEY ("SERVICE_ID")
```

```
              REFERENCES "SERVICE" ("SERVICE_ID") ENABLE,
             FOREIGN KEY ("USERID")
              REFERENCES "USERLOGIN" ("USERID") ENABLE
   ) ;

------------------------------------------------------------------------------------------------

CREATE TABLE "CLAIM"
   (        "CID" NUMBER(*,0) NOT NULL ENABLE,
            "TOTALCHARGEOFCUSTOMER" NUMBER(*,0) NOT NULL ENABLE,
            "TOTALCHARGEOFINSURANCE" NUMBER(*,0) NOT NULL ENABLE,
             PRIMARY KEY ("CID")
   );

---------------------------------------------------------------------------------------

CREATE TABLE "CLAIM_LINE"
   (        "CLAIM_ID" NUMBER(*,0) NOT NULL ENABLE,
            "STATUS" VARCHAR2(50 BYTE),
            "PROVIDERS_CHARGE" NUMBER(*,0) NOT NULL ENABLE,
            "AMOUNT_COPAY" NUMBER(*,0),
            "AMOUNT_DEDUCTABLE" NUMBER(*,0),
            "AMOUNT_OF_COINSURANCE" NUMBER(*,0),
            "AMOUNT_PAID_BYINSURANCE" NUMBER(*,0),
            "AMOUNT_PAID_BYCUSTOMER" NUMBER(*,0),
            "MESSAGE_ID" NUMBER(*,0),
            "SERVICE_ID" NUMBER(*,0) NOT NULL ENABLE,
            "SERVICE_DATE" DATE NOT NULL ENABLE,
            "POLICY_ID" NUMBER(*,0) NOT NULL ENABLE,
            "CLAIM_DATE" DATE NOT NULL ENABLE,
            "USER_ID" NUMBER(*,0),
            "SP_ID" NUMBER(*,0),
            "CID" NUMBER(*,0),
             PRIMARY KEY ("CLAIM_ID")
);

--------------------------------------------------------------

 CREATE TABLE "MESSAGE"
   (        "MESSAGE_ID" NUMBER(*,0) NOT NULL ENABLE,
            "MESSAGE_BODY" VARCHAR2(255 BYTE),
            "MESSAGE_DATE" DATE,
            "USERID" NUMBER(*,0),
             PRIMARY KEY ("MESSAGE_ID"),
             FOREIGN KEY ("USERID")
              REFERENCES "USERLOGIN" ("USERID") ENABLE
   );
```

```
 CREATE TABLE "PREMIUM_LEVELS"
  (        "LEVEL_ID" NUMBER(*,0) NOT NULL ENABLE,
           "LEVEL_DESCRIPTION" VARCHAR2(255 BYTE) NOT NULL ENABLE,
            PRIMARY KEY ("LEVEL_ID"));
```

```
 CREATE TABLE "PREMIUM"
  (        "PREMIUM_ID" NUMBER(*,0) NOT NULL ENABLE,
           "POLICY_ID" NUMBER(*,0) NOT NULL ENABLE,
           "LEVEL_ID" NUMBER(*,0) NOT NULL ENABLE,
           "PREMIUM_AMOUNT" NUMBER(*,0),
            PRIMARY KEY ("PREMIUM_ID"),
            FOREIGN KEY ("POLICY_ID")
             REFERENCES "POLICY" ("POLICY_ID") ENABLE,
            FOREIGN KEY ("LEVEL_ID")
             REFERENCES "PREMIUM_LEVELS" ("LEVEL_ID") ENABLE
  );
```

3. Sequences Created

```
CREATE SEQUENCE  "CLAIMID_SEQ"  MINVALUE 1 MAXVALUE
9999999999999999999999999999 INCREMENT BY 1 START WITH 6;


CREATE SEQUENCE  "CUSTID_SEQ"  MINVALUE 1 MAXVALUE
9999999999999999999999999999 INCREMENT BY 1 START WITH 6;

CREATE SEQUENCE  "DID_SEQ"  MINVALUE 1 MAXVALUE 9999999999999999999999999999
INCREMENT BY 1 START WITH 6;

CREATE SEQUENCE  "MESSAGE_SEQ"  MINVALUE 1 MAXVALUE
9999999999999999999999999999 INCREMENT BY 1 START WITH 6;
```

```
CREATE SEQUENCE  "MESSAGEID_SEQ"  MINVALUE 1 MAXVALUE
9999999999999999999999999999 INCREMENT BY 1 START WITH 6;

CREATE SEQUENCE "POLICY_SEQ"  MINVALUE 1 MAXVALUE
9999999999999999999999999999 INCREMENT BY 1 START WITH 6;

CREATE SEQUENCE  "SPID_SEQ"  MINVALUE 1 MAXVALUE 9999999999999999999999999999
INCREMENT BY 1 START WITH 6;

CREATE SEQUENCE  "USERID_SEQ"  MINVALUE 1 MAXVALUE
9999999999999999999999999999 INCREMENT BY 1 START WITH 16;
```

4. Insert Statements

Usertype Table:

```
INSERT ALL
  INTO usertype VALUES (1,'Customer')
  INTO usertype VALUES (2,'Dependent')
  INTO usertype VALUES (3,'Service Provider')
SELECT * FROM dual;
```

Userlogin Table:

```
INSERT ALL
  INTO userlogin VALUES (1,'va@gmail.com','VA$abc', 1)
  INTO userlogin VALUES (2,'bs@yahoo.com','BS@abc', 2)
  INTO userlogin VALUES (3,'jt@atna.com','JT6#xyz',3)
  INTO userlogin VALUES (4,'Magnusfava@gmail.com','mang79@a',1)
  INTO userlogin VALUES (5,'anthonyvarghese@yahoo.com','anth12$',2)
  INTO userlogin VALUES (6,'niky@ISO.com','iso12$',3)
  INTO userlogin VALUES (7,'braxton@healthcare.com','hc82$',3)
  INTO userlogin VALUES (8,'shimi@wellpoint.com','wellpoint12$',3)
  INTO userlogin VALUES (9,'drey@bluecross.com','blue47$',3)
  INTO userlogin VALUES (10,'cht@gmail.com','chat#12c', 2)
  INTO userlogin VALUES (11,'bspa@gmail.com','bspa@45', 2)
  INTO userlogin VALUES (12,'Sara Fava','fava@gmail.com', 2)
  INTO userlogin VALUES (13,'Lila Bhansari','Lila@yahoo.com', 1)
  INTO userlogin VALUES (14,'Drenta Kale','der@hotmail.com', 1)
```

```
 INTO userlogin VALUES (15,'Asit Madan','asit@gmail.com', 1)
SELECT * FROM dual;
```

Service Table:

```
INSERT ALL
  INTO service VALUES (1,'Blood Test')
  INTO service VALUES (2,'X-ray Test')
  INTO service VALUES (3,'Physician Visit')
  INTO service VALUES (4,'ECG')
  INTO service VALUES (5,'Physiotheraphy')
  INTO service VALUES (6,'MRI Scan')
SELECT * FROM dual;
```

Service_Provider Table:

```
INSERT ALL
  INTO service_provider VALUES
(1,'AETNA',1,3,'jt@atna.com','JT6#xyz','Maryland','+144536252','In-network')
  INTO service_provider VALUES
(2,'ISO',2,6,'niky@ISO.com','iso12$','California','+165943519','Out-network')
  INTO service_provider VALUES
(3,'AETNA',2,3,'jt@atna.com','JT6#xyz','Maryland','+144536252','In-network')
  INTO service_provider VALUES
(4,'HealthCare',3,7,'braxton@healthcare.com','hc82$','Columbia','+18797521','Out-network')
  INTO service_provider VALUES (5,'Health
Group',6,7,'braxton@healthcare.com','hc82$','Columbia','+18797521','Out-network')
  INTO service_provider VALUES (6,'Well
Point',4,8,'shimi@wellpoint.com','wellpoint12$','Texas','+19874563','In-network')
  INTO service_provider VALUES (7,'Blue
Cross',5,9,'drey@bluecross.com','blue47$','Maine','+156513887','In-network')
SELECT * FROM dual;
```

Coverage Table:

```
INSERT ALL
  INTO coverage VALUES (1,99999,100,20,40,5,10,1)
  INTO coverage VALUES (2,5,150,30,60,6,12,2)
  INTO coverage VALUES (3,4,200,40,70,5,10,3)
  INTO coverage VALUES (4,5,250,20,40,5,10,4)
  INTO coverage VALUES (5,4,350,20,40,6,9,5)
  INTO coverage VALUES (6,3,100,20,40,5,10,6)
  INTO coverage VALUES (7,99999,1000,100,150,10,20,6)
SELECT * FROM dual;
```

Plan Table:

```
INSERT ALL
  INTO plan VALUES (1,'PPO', date '2012-01-01',12,500,750,1000,1000,1,1,date '2012-12-31')
  INTO plan VALUES (2,'Family First', date '2013-01-01',12,500,800,1000,1500,2,2,date '2013-12-31')
  INTO plan VALUES (3,'Gateway PPO', date '2016-01-01',18,500,900,1000,2000,3,3,date '2016-12-31')
  INTO plan VALUES (4,'Medicare', date '2014-01-01',12,350,600,900,2500,4,4,date '2014-12-31')
  INTO plan VALUES (5,'Community First Health', date '2015-01-01',12,500,1000,1200,3000,5,5,date
'2015-12-31')
SELECT * FROM dual;
```

Policy Table:

```
INSERT ALL
  INTO policy VALUES (1,2,2,2,1)
  INTO policy VALUES (2,2,2,2,2)
  INTO policy VALUES (3,3,3,3,4)
  INTO policy VALUES (4,4,4,4,5)
  INTO policy VALUES (5,4,4,4,4)
SELECT * FROM dual;
```

Premium_level Table:

```
INSERT ALL
  INTO PREMIUM_LEVELS VALUES (1,'one adult ')
  INTO PREMIUM_LEVELS VALUES (2,'two adult ')
  INTO PREMIUM_LEVELS VALUES (3,'one adult and one or more children')
  INTO PREMIUM_LEVELS VALUES (4,'two adult and one or more children')
SELECT * FROM dual;
```

Premium Table:

```
INSERT ALL
```

```
  INTO premium VALUES (1,1,1,1000)
  INTO premium VALUES (2,2,3, 2000)
  INTO premium VALUES (3,3,1, 2000)
  INTO premium VALUES (4,4,1, 1500)
  INTO premium VALUES (5,2,1, 1500)
SELECT * FROM dual;
```

Customer Table:

```
INSERT ALL
  INTO customer VALUES (1,'Afsha Shaikh','va@gmail.com','VA$abc', date '1978-04-14', 'Female',
'Baltimore', '443-980-6987',1)
  INTO customer VALUES (2,'Neel Patel','Magnusfava@gmail.com','mang79@a', date '1980-11-01',
'Male', '4737 Aldgate 21227', '443-750-2587',4)
  INTO customer VALUES (3,'Kritesh Arora','Lila@yahoo.com','lila@3', date '1980-11-01', 'Male', '4737
Gateway 21227', '443-750-2587',13)
  INTO customer VALUES (4,'Smit Vasani','der@hotmail.com','der@45', date '1981-02-15', 'Male', '4736
Star Terrace 21227', '443-894-2587',14)
  INTO customer VALUES (5,'Asita Madan','asit@gmail.com','asit#7', date '1985-02-15', 'Female', '5001
Maidan Choice 21227', '443-745-2587',15)
SELECT * FROM dual;
```

Dependent Table:

```
INSERT ALL
   INTO dependent VALUES (1,'Borris','bs@yahoo.com','BS@abc', date '1990-04-21', 'Female', '2188
Arbutus 21227', '443-960-6987',2,'Wife',1)
   INTO dependent VALUES (2,'Anthony','anthonyvarghese@yahoo.com','anth12$', date '1980-11-01',
'Male', '4733 Aldgate 21227', '443-720-2587',5,'Father',2)
   INTO dependent VALUES (3,'Barbara','cht@gmail.com','chat#12', date '1982-04-21', 'Female', '4736
Arbutus 21227', '443-960-6987',10,'Mother',1)
   INTO dependent VALUES (4,'Brittany','bspa@gmail.com','bspa@45', date '2001-04-21', 'Female',
'4736 Arbutus 21227', '443-960-6987',11,'Spouse',1)
   INTO dependent VALUES (5,'Sara Fava','fava@gmail.com','fava@1', date '1990-04-21', 'Female', '4737
Aldgate 21227', '443-960-6987',12,'Wife',2)
SELECT * FROM dual;
```

Message Table:

```
INSERT ALL
  INTO message VALUES (1,'Bill amount due', date '2016-01-25',1)
  INTO message VALUES (2,'Bill amount paid', date '2017-03-10',1)
  INTO message VALUES (3,'Thank you for purchasing our insurance plan', date '2016-01-05',4)
  INTO message VALUES (4,'For any queries contact us at 2556', date '2017-01-15',4)
  INTO message VALUES (5,'Bill amount updated', date '2016-09-02',1)
SELECT * FROM dual;


--------------------------------------------------------------
Claim_line Table:

INSERT ALL
  INTO claim_line VALUES (1,'Accept', 250,20,30,50,150,0,1,1,date '2016-10-09',1, date '2016-10-
09',1,1,1)
  INTO claim_line VALUES (2,'Accept',1000,250,250,100,50,200,2,1,date '2016-12-01',2, date '2016-12-
01',1,2,1)
  INTO claim_line VALUES (3,'Declined',2500,200,200,100,1000,1500,4,4,date '2016-12-30',3, date
'2016-12-30',2,4,2)
  INTO claim_line VALUES (4,'Accept',1500,100,150,250,1000,0,3,4, date '2017-01-02',4, date '2017-01-
02',2,6,2)
  INTO claim_line VALUES (5,'Declined',1000,100,100,250,450,100,4,4, date '2017-02-15',5,date '2017-
02-15',3,7,3)
 SELECT * FROM dual;
-------------------------------------------------------------------------
Claim Table:

INSERT ALL
INTO CLAIM VALUES(1,2583,1698)
INTO CLAIM VALUES(2,2368,1589)
INTO CLAIM VALUES(3,2753,1115)
SELECT * FROM DUAL;
```

5. Additional Functions Created

```
-------------------Feature 1-------------------------------------
create or replace function return_userid (emailId in varchar,user_pswd in varchar)
return integer
IS
```

```
user_id integer;
user_password varchar(50);
BEGIN
            select userid,pswd into user_id,user_password from userlogin where email_id = emailId;
--    if(user_password = user_pswd) then
--     dbms_output.put_line('Login Successful');

     dbms_output.put_line('User already exists.');
            return 0;
exception
            WHEN OTHERS THEN
            dbms_output.put_line('New User. Creating New User id......');
            return -1;
End;


create or replace function return_custuserid (name in varchar,emailId in varchar,user_pswd in
varchar,cust_address in varchar, phone in varchar, birthdate in date, cust_gender in varchar)
return integer
IS
user_id number;
userType_id usertype.ut_id%type;
BEGIN
    SELECT ut_id INTO userType_id FROM USERTYPE WHERE user_type='Customer';
    INSERT INTO userlogin VALUES (userid_seq.nextval, emailId, user_pswd,userType_id);
    SELECT userid INTO user_id FROM userlogin WHERE email_id = emailId;
    INSERT INTO customer VALUES (custId_seq.nextval, name, emailId,user_pswd,
birthdate,cust_gender,cust_address,phone,user_id);
            return user_id;
exception
            WHEN OTHERS THEN
            dbms_output.put_line('Error in registration with Customer Type');
            return -1;
End;

create or replace function return_dependuserid (name in varchar,emailId in varchar,user_pswd in
varchar,cust_address in varchar, phone in varchar, birthdate in date, cust_gender in varchar, cust_userId
in integer,dependent_relation in varchar)
return integer
IS
user_id number;
userType_id usertype.ut_id%type;
customer_id integer;
BEGIN
```

```
    SELECT ut_id INTO userType_id FROM USERTYPE WHERE user_type='Dependent';
    INSERT INTO userlogin VALUES (userid_seq.nextval, emailId, user_pswd,userType_id);
    SELECT userid INTO user_id FROM userlogin WHERE email_id = emailId;
    SELECT cust_id INTO customer_id FROM CUSTOMER WHERE userid = cust_userId;
    INSERT INTO dependent VALUES (did_seq.nextval,name,emailId,user_pswd, birthdate, cust_gender,
cust_address, phone,user_id,dependent_relation,customer_id);
    dbms_output.put_line('Registration is done Successfully');
   return user_id;
exception
            WHEN OTHERS THEN
            dbms_output.put_line('Error in registration with Dependent Type');
            return -1;
End;

create or replace function return_provideruserid(name in varchar,emailId in varchar,user_pswd in varchar
,cust_address in varchar , phone in varchar ,provider_type in varchar,serviceid in varchar)
return integer
IS
user_id integer;
userType_id usertype.ut_id%type;
serviceProvider_id integer;
BEGIN
    SELECT ut_id INTO userType_id FROM USERTYPE WHERE user_type='Service Provider';
    INSERT INTO userlogin VALUES (userid_seq.nextval, emailId, user_pswd,userType_id);
    SELECT userid INTO user_id FROM userlogin WHERE email_id = emailId;
  --  SELECT cust_id INTO customer_id FROM CUSTOMER WHERE userid = cust_userId;
    INSERT INTO SERVICE_PROVIDER VALUES
(spid_seq.nextval,name,serviceid,userid_seq.currval, emailId, user_pswd,cust_address,
phone,provider_type);
    dbms_output.put_line('Registration is done Successfully');
   return user_id;
exception
            WHEN OTHERS THEN
            dbms_output.put_line('Error in registration with Service Provider Type');
            return -1;
End;



---------------Functions for Feature 2----------------------------
---------------------------

Create or replace function check_login (emailId in varchar,user_pswd in varchar) /* Allows user to login.
Checks whether email and password matches. */
```

```
return integer
IS

/* Variable declaration */

user_id integer;
user_password varchar(50);
BEGIN
            select userid,pswd into user_id,user_password from userlogin where email_id = emailId;

/* Check if logins successful */

    if(user_password = user_pswd) then
     dbms_output.put_line('Login Successful');
            return 1;

/* If password mis matches */

 elsif(user_password <> user_pswd) then
     dbms_output.put_line('Incorrect Password');

 return 0;
  End if;
 exception
            WHEN OTHERS THEN

/* If emailid mis matches */

            dbms_output.put_line('Incorrect email id');
            return 0;
End;

---------------------Function for Feature 4------------------
-----------------------

create or replace function return_policyid (us_id in number, planname in varchar, start_year in date) /*
Inserts a new policy into the policy table for the customer*/
return integer

/* Variable Declaration */

IS
planid plan.plan_id%type;
coverageid PLAN.COVERAGE_ID%type;
```

```
serviceid PLAN.SERVICE_ID%type;
pol_id policy.policy_id%type;
myexec EXCEPTION;
u_id integer;
pl_id integer;
count1 number;
BEGIN
    SELECT plan_id, COVERAGE_ID, SERVICE_ID INTO planid, coverageid, serviceid FROM plan
WHERE plan_name = planname;
   count1:= checkUserExist(us_id,planname);

    if( (count1 = 0)) then
    raise myexec;
    else
    INSERT INTO policy VALUES (policy_seq.nextval, planid, coverageid, serviceid,us_id);
    end if;
             return policy_seq.currval;
exception
             --WHEN OTHERS THEN
/* Printing the exception */
             --dbms_output.put_line('Error in adding the policy User');
  WHEN myexec THEN
   dbms_output.put_line('Policy already exists ');
             return -1;
End;

create or replace function checkUserExist (us_id in number, planname in varchar)
return integer
IS
/* Variable Declaration*/

u_id integer;
pl_id integer;

BEGIN

/* Check for already existing plan */

              select policy.user_id into u_id from plan,policy where plan.plan_id = policy.plan_id AND
policy.user_id=us_id AND plan.plan_name=planname;
     select policy.plan_id into pl_id from plan,policy where plan.plan_id = policy.plan_id AND
policy.user_id=us_id AND plan.plan_name=planname;
     dbms_output.put_line('User is already enroll with Insurance plan');
             return 0;
```

```
exception
	WHEN OTHERS THEN
	dbms_output.put_line('Inserting new insurance policy to respective user...........');
	return -1;
End;

create or replace function return_policyDepdid(user_id in number, planname in varchar, start_year in
date) /* Inserts a new policy into the policy table for the dependent*/
return integer

/* Variable Declaration */

IS
planid plan.plan_id%type;
coverageid PLAN.COVERAGE_ID%type;
serviceid PLAN.SERVICE_ID%type;
pol_id policy.policy_id%type;
cid number;
usid number;
deptid number;
BEGIN
   SELECT plan_id, COVERAGE_ID, SERVICE_ID INTO planid, coverageid, serviceid FROM plan
WHERE plan_name = planname;

   /* Fetching the user id of the dependent's related customer and inserting a policy with that customer's
user id.*/

   select cust_id into cid from dependent where userid = user_id;
   select userid into usid from customer where cust_id = cid;
   select d_id into deptid from dependent where userid = user_id;
   INSERT INTO policy VALUES (policy_seq.nextval, planid, coverageid, serviceid,usid);
   INSERT INTO POLICY_DEPENDENT values(policy_seq.currval, deptid, usid);

	return policy_seq.currval;
exception
	WHEN OTHERS THEN

/* Printing the exception */

	dbms_output.put_line('Error in adding the policy Dependent');
	return -1;
End;
```

```
create or replace function insert_msg(user_id in number, pol_id in number) /* Inserts into message table
*/
return number
IS

/* Variable Declaration */

msg_id message.message_id%type;
begin
  insert into message values (message_seq.nextval,' The customer has been enrolled into the policy with
the polic ID ' || pol_id,sysdate,user_id);
return message_seq.currval;
End;

-----------------------------------------------Feature 5-----------------------------------------------
-----------------------------------------------Function 1----------------------------------------------
create or replace function add_msg(user_id in number, dep_id in number, pol_id in number) --function
for inserting message in the message table
return number
IS
msg_id message.message_id%type;
begin
  insert into message values (message_seq.nextval,' the dependent ' || dep_id || ' has been added to the
policy ' || pol_id,sysdate,user_id);
 -- select message_seq.currval into msg_id from message;
return message_seq.currval;
End;

---------------------------------------------Function 2------------------------------------------------

create or replace function get_existing_depid (dep_id in number, pol_id in number) --function for
checking the existing entry of dependent in policy dependent
return number
IS
dep1_id dependent.d_id%type;
pol1_id POLICY_DEPENDENT.policy_id%type;
BEGIN
	Select d_id, policy_id into dep1_id, pol1_id from policy_dependent where d_id=dep_id and
policy_id=pol_id;

  return 0; --if the dependent already exists for the policy in the policy dependent table
exception
	when others then
```

```
            return -1; --if the dependent does not exist for the policy in the policy dependent table
End;

----------------------------------------------Feature 6------------------------------------------
----------------------------------------------Function 1------------------------------------------
create or replace function get_existing_deptid (dep_id in number, pol_id in number) --function for
checking the existing entry of dependent in policy dependent
return number
IS
dep1_id dependent.d_id%type;
pol1_id POLICY_DEPENDENT.policy_id%type;
BEGIN
            Select d_id, policy_id into dep1_id, pol1_id from policy_dependent where d_id=dep_id and
policy_id=pol_id;
  return 0; --if the dependent already exists for the policy in the policy dependent table
exception
            when others then
            return -1; --if the dependent does not exist for the policy in the policy dependent table
End;

--------------------------------------------Function 2-------------------------------------------

create or replace function add_msg1(user_id in number, dep_id in number, pol_id in number) --function
for inserting message in the message table
return number
IS
msg_id message.message_id%type;
begin
  insert into message values (message_seq.nextval,' the dependent ' || dep_id || ' has been removed from the
policy ' || pol_id,sysdate,user_id);
 return message_seq.currval;
End;

-----------------Function for Feature 7------------------------------
----------------------
create or replace function calculate_premium (pol_id in number) /* Calculating the premium amount for
the given policy id. */
return integer
IS

/*Variable Declaration*/

premium_amt PREMIUM.PREMIUM_AMOUNT%type;
levelid POLICY.POLICY_ID%type;
```

```
planid PLAN.PLAN_ID%type;
std_annualrate PLAN.STANDARD_ANNUAL_RATE%type;
count1 number;
myexec Exception;
BEGIN
count1:= checkPolicyExist(pol_id);
if( (count1 <> 0)) then
   raise myexec;
   else
   /* Fetching the level id */

   select level_id into levelid from premium where policy_id=pol_id;

   /* Fetching the plan id */

   select plan_id into planid from policy where policy_id=pol_id;

   /* Fetching the standard annual rate for that plan */

   select standard_annual_rate into std_annualrate from plan where plan_id=planid;

   /* Calculating the premium amount for the policy */


   premium_amt:= std_annualrate * levelid;
   dbms_output.put_line('The Premium Amount for policy ' || pol_id || ' with level id ' || levelid || ' with the
standar annual rate of ' || std_annualrate || ' is: ' || premium_amt);
   return premium_amt;
   end if;
   exception
          --WHEN OTHERS THEN
/* Printing the exception */
          --dbms_output.put_line('Error in adding the policy User');
  WHEN myexec THEN
  dbms_output.put_line('Policy does not exists ');
          return -1;
End;

create or replace function checkPolicyExist (pol_id  in number)
return integer
IS
po_id integer;

BEGIN
```

```
  select policy_id into po_id from policy where policy_id=pol_id;
                dbms_output.put_line('Policy does not exists');
            return 0;
exception
            WHEN OTHERS THEN
            --dbms_output.put_line('Calculating Premium for the policy..........');
            return -1;
End;

-------------------------------------------Feature 9---------------------------------------------------------
----------------------------------------------Function 1----------------------------------------------
create or replace function feature9_get_existing_policyid (pol_id in number)
return number
IS
pol1 policy.policy_id%type;
-----------check if the policy exist
BEGIN
            Select policy_id into pol1 from policy where policy_id=pol_id;
  return 0;
exception
            when others then
            return -1;
End;

----------------------------------------------Function 2----------------------------------------------
create or replace function feature9_get_existing_provid (prov in number)
return number
IS
prov1 service_provider.sp_id%type;
----checks if the provider exists
BEGIN
            Select sp_id into prov1 from service_provider where sp_id=prov;
  return 0;
exception
            when others then
            return -1;
End;

----------------------------------------------Function 3----------------------------------------------
create or replace function feature9_patientcheckincust (patient_name in VARCHAR2)
return number
IS
patient1 number;
---checks for patient in customer table
```

```
BEGIN
select userid into patient1 from customer where cust_name=patient_name;
return 0;
exception
when others then
return -1;
End;

----------------------------------------------Function 4----------------------------------------------
create or replace function feature9_patientcheckindepd (patient_name in VARCHAR2)
return number
IS
patient1 number;
----------checks for patient in dependent table
BEGIN
select userid into patient1 from dependent where d_name=patient_name;
return 0;
exception
when others then
return -1;
End;

----------------------------------------------Function 5----------------------------------------------
create or replace function feature9_patientpolicy_cust (pol_id in number,patient_name in VARCHAR2)
return number
IS
pol2 number;
patient1 number;
----------to check if the patient is linked to the policy
BEGIN
select userid into patient1 from customer,policy where policy.user_id=customer.userid and
cust_name=patient_name;

select policy_id into pol2 from policy where policy_id=pol_id and user_id=patient1;
return 0;
exception
when others then
return -1;
End;

----------------------------------------------Function 6----------------------------------------------
create or replace function feature9_patientpolicy_depd (pol_id in number,patient_name in VARCHAR2)
return number
IS
```

```
pol2 number;
patient1 number;
cid number;
uid number;
----------to check if the patient is linked to the policy

BEGIN
select d_id,cust_id into patient1,cid from dependent where d_name=patient_name;
select userid into uid from customer where cust_id = cid;
select policy_id into pol2 from POLICY_DEPENDENT where policy_id=pol_id and d_id=patient1;
return 0;
exception
when others then
return -1;
End;

-----------------------------------------------Function 7-----------------------------------------------
create or replace function feature9_checkdate (pol_id in number,date_of_service in date)
return number
IS
date1 claim_line.service_date%type;
BEGIN
---checks if the service date within allowed plan range
select plan_start_year into date1 from plan p, policy q
where p.plan_id = q.plan_id and  p.plan_start_year <=date_of_service and p.plan_end_date
>=date_of_service and q.policy_id=pol_id;
if (date1 is not null) then
return 0;
else return -1;
end if;
exception
when others then
return -1;
End;

-----------------------------------------------Function 8-----------------------------------------------
create or replace function f9_check_serviceinpolicy_cust (patient_name in varchar,serv_id in
number,pol_id in number)
return number
IS
u_id number;
serial_1 number;
serial_2 number;
serial_3 number;
```

```
BEGIN
------------fetches customer userid and count of claim id for accepted claim
select userid into u_id from customer where cust_name=patient_name;
Select count(SERVICE_ID) into serial_1 from policy where service_id=serv_id and policy_id=pol_id;
Select Coverage.Max_Service_Peryear into serial_2 from policy, coverage where
policy.service_id=serv_id and policy.policy_id=pol_id and policy.service_id=coverage.service_id;
select count(claim_id) into serial_3 from claim_line where user_id=u_id and
claim_line.service_id=serv_id and claim_line.STATUS='Accept';
---------checks if service linked to policy and if total services more than allowed number of accepted
service

if (serial_1>0) then
    if (serial_3>serial_2) then
  return 2;
 else
 return 1;
  end if;
else return 3;
end if;
exception
            when others then
            return 3;
End;

-------------------------------------------------Function 9------------------------------------------------
create or replace function f9_check_serviceinpolicy_depd (patient_name in varchar,serv_id in
number,pol_id in number)
return number
IS
u_id number;
serial_1 number;
serial_2 number;
serial_3 number;
BEGIN
------------fetches dependent userid and count of claim id for accepted claim
select userid into u_id from dependent where d_name=patient_name;
Select count(SERVICE_ID) into serial_1 from policy where service_id=serv_id and policy_id=pol_id;
Select Coverage.Max_Service_Peryear into serial_2 from policy, coverage where
policy.service_id=serv_id and policy.policy_id=pol_id and policy.service_id=coverage.service_id;
select count(claim_id) into serial_3 from claim_line where user_id=u_id and
claim_line.service_id=serv_id and claim_line.STATUS='Accept';
---------checks if service linked to policy and if total services more than allowed number of accepted
service
```

```
  if (serial_1>0) then
    if (serial_3>serial_2) then
  return 2;
 else
 return 1;
  end if;
else return 3;
end if;
exception
            when others then
            return 3;
End;

-----------------------------------------------Function 10-----------------------------------------------
create or replace function f9_3_check_duplicate_cust (patient_name in varchar,serv_id in
number,date_of_service in date)
return number
IS
--variable declaration
u_id number;
claimid number;
BEGIN

---fetches user id of the customer
select userid into u_id from customer where cust_name=patient_name;
---fetches claim_id from claim_line for accept status and same date to check for duplicate claim
select claim_id into claimid from claim_line where service_date=date_of_service and status='Accept' and
user_id=u_id;
---if claimid is not null means duplicate claim
if (claimid is not null) then
  return -1;
  else
  return 1;
end if;
exception
            when others then
            return 0;
End;

-----------------------------------------------Function 11-----------------------------------------------
create or replace function f9_3_check_duplicate_depd (patient_name in varchar,serv_id in
number,date_of_service in date)
return number
IS
```

```
u_id number;
claimid number;
BEGIN
--fetches user id of the dependent
select userid into u_id from dependent where d_name=patient_name;
select claim_id into claimid from claim_line where service_date=date_of_service and status='Accept' and
user_id=u_id;
---if claimid is not null means duplicate claim
if (claimid is not null) then
  return -1;
  else
  return 1;
end if;
exception
        when others then
        return 0;
End;

----------------------------------------------Function 12----------------------------------------------
create or replace function f9_4_Adjustcharge_cust (pol_id in number,serv_id in number,prov_id in
number,amnt in number, patient_name in varchar, date_of_service in date)
return number
IS
u_id number;
sew number;

BEGIN

select userid into u_id from customer where cust_name=patient_name;
--fetches the allowed service charge
select allowed_service_charges into sew from policy,coverage
where policy.coverage_id=coverage.COVERAGE_ID and policy.POLICY_ID=pol_id and
policy.user_id=u_id;
----compares the allowed service charge and providers charge
---if providers charge is lower than allowed charge, then providers charge is providers charge else allowed
charge
if(sew >= amnt) then
  insert into
claim_line(claim_id,service_id,policy_id,user_id,service_date,claim_date,sp_id,providers_charge)
values(claimid_seq.nextval,serv_id,pol_id,u_id,date_of_service,sysdate,prov_id,amnt);
  return 0;
else
```

```
  insert into
claim_line(claim_id,service_id,policy_id,user_id,service_date,claim_date,sp_id,providers_charge)
values(claimid_seq.nextval,serv_id,pol_id,u_id,date_of_service,sysdate,prov_id,sew);
  return 0;
End if;
exception
        when others then
        return -1;
End;

-----------------------------------------------Function 13-----------------------------------------------
create or replace function f9_4_Adjustcharge_depd (pol_id in number,serv_id in number,prov_id in
number,amnt in number, patient_name in varchar, date_of_service in date)
return number
IS
u_id number;
sew number;
BEGIN
select userid into u_id from dependent where d_name=patient_name;
select allowed_service_charges into sew from policy,coverage
where policy.coverage_id=coverage.COVERAGE_ID and policy.POLICY_ID=pol_id and
policy.user_id=u_id;
----compares the allowed service charge and providers charge
---if providers charge is lower than allowed charge, then providers charge is providers charge else allowed
charge

if(sew >= amnt) then
  insert into
claim_line(claim_id,service_id,policy_id,user_id,service_date,claim_date,sp_id,providers_charge)
values(claimid_seq.nextval,serv_id,pol_id,u_id,date_of_service,sysdate,prov_id,amnt);
  return 0;
else
  insert into
claim_line(claim_id,service_id,policy_id,user_id,service_date,claim_date,sp_id,providers_charge)
values(claimid_seq.nextval,serv_id,pol_id,u_id,date_of_service,sysdate,prov_id,sew);
  return 0;
End if;
exception
        when others then
        return -1;
End;

-----------------------------------------------Function 14-----------------------------------------------
```

```
create or replace Function F9_5_6_7_Pay_CUST (Pol_Id In Number,Serv_Id In Number,Prov_Id In
Number,Amnt In Number, Patient_Name In Varchar, Date_Of_Service In Date)
Return Number
Is
--variable declaration
sum20 integer;
var1 integer;
onci integer;
PD integer;
sew integer;
INCI integer;
U_Id integer;
Total_Cust integer;
Serv_Prov_Type Varchar(255);
Difference1 integer;
Mopm integer;
Onc integer;
Inc integer;
Sum1 integer;
Diff2 integer;
AMNT_copay integer;
amnt_coinsurance integer;
sum2 integer;
sum3 integer;
DIFF8 integer;
DIFF7 integer;
sum6 integer;
sum5 integer;
diff4 integer;
diff5 integer;
diff6 integer;
diff3 integer;
total integer;
sum4 integer;
sum7 integer;
calc1 integer;
sum8 integer;
diff9 integer;
diff10 integer;
diff11 integer;
amount_coinsurance integer;
sum21 integer;
var2 integer;
sum22 integer;
```

```
diff20 integer;
Begin
----select queries to fetch user id, maximum out of pocket, total amount paid by customer till date, service
provider type, plan deductable and providers charge
Select Userid Into U_Id From Customer Where Cust_Name=Patient_Name;
Select Max_Opc_Permember Into Mopm From Plan, Policy Where Plan.Plan_Id=Policy.Plan_Id And
Policy.Policy_Id=Pol_Id;
Select Sum(Amount_Paid_Bycustomer) Into Total_Cust From Claim_Line Where User_Id=U_Id;
Select Sp_Type Into Serv_Prov_Type From Service_Provider Where Sp_Id=Prov_Id And
Service_Id=Serv_Id;
Select Plan.Deductable_Amount Into Pd From Plan,Policy Where Policy.Plan_Id=Plan.Plan_Id And
Policy.Policy_Id=Pol_Id;
Select Providers_Charge Into Sew From Claim_Line Where User_Id=U_Id And Policy_Id=Pol_Id And
Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;

-----computes various values like checking the total paid by customer and current charge
Sum1:= Total_Cust+Sew;
----checks mam out of pocket and how much customer paid
Difference1 := Mopm - Total_Cust;
Diff2 := Mopm - Total_Cust;
---checks the provider type
 If(Serv_Prov_Type = 'In-network') Then
---selects the amount copay and coinsurance
  Select In_Network_Copay Into Inc From Policy,Plan,Coverage Where Policy.Plan_Id=Plan.Plan_Id
And Plan.Coverage_Id=Coverage.Coverage_Id And Policy.Policy_Id=Pol_Id;
  Select In_Network_Coinsurance Into Inci From Policy,Plan,Coverage Where
Policy.Plan_Id=Plan.Plan_Id And Plan.Coverage_Id=Coverage.Coverage_Id And
Policy.Policy_Id=Pol_Id;
----amount copay is fetched
  Amnt_Copay:=Inc;
----calculates amount of coinsurance
  Amnt_Coinsurance:=((Sew-Amnt_Copay)*Inci)/100;
----sum of total paid by customer and copay
  Sum2:=Amnt_Copay+Total_Cust;
----computes total copay, amount paid till date and coinsurance
-----below are various computations to calculate copay and coinsurance in various cases
Sum3:=Amnt_Coinsurance+Sum2;
  Diff4:=Sum3-Mopm;
  Sum4:=Inc+Diff4;
  Diff3:= Mopm-Sum2;
  Diff5:=Mopm - Sum2;
  Diff6:=Sew-Sum4;
  Sum5:=amnt_coinsurance+Inc;
  Diff7:=Sew-Sum5;
  Sum6:=Total_Cust+Sew;
  Diff8:=Pd-Total_Cust;
  diff20:=sew-diff8;
  Sum7:=Diff8+Inc;
  Calc1:=((Sew-Sum7)*Inci)/100;
  Sum8:=Inc+Calc1+Diff8;
  Diff9:=Sew-Sum8;
  Diff10:=Mopm-Total_Cust;
  Diff11:=Sew-Diff10;
  var1:=sum1-pd;
  sum20:= var1+diff8;
  sum21 := diff8 + inc;
  var2 := sew - sum21;
  sum22 := sum21 + var2;
         ---------checks if total paid by customer > max out of pocket and policy deductable
             If (Total_Cust >= Mopm And Total_Cust>Pd) Then
         Update Claim_Line Set Amount_Copay = 0 Where User_Id=U_Id And Policy_Id=Pol_Id And
Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
         Update Claim_Line Set Amount_of_Coinsurance = 0 Where User_Id=U_Id And Policy_Id=Pol_Id
And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
         Update Claim_Line Set Amount_Paid_Bycustomer = 0 Where User_Id=U_Id And Policy_Id=Pol_Id
And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
         Update Claim_Line Set Amount_Paid_Byinsurance = Sew Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
         update claim_line set amount_deductable = 0 where user_id=u_id and policy_id=pol_id and
service_id=serv_id and sp_id=prov_id and service_date=date_of_service;
         update claim_line set status = 'Accept' where user_id=u_id and policy_id=pol_id and
service_id=serv_id and sp_id=prov_id and service_date=date_of_service;
         update claim_line set cid=2 where user_id=u_id and policy_id=pol_id and service_id=serv_id and
sp_id=prov_id and service_date=date_of_service;
         Return 1;
             ---------checks if current copay, coinsurance more than max out of pocket expense and more than
policy deductable
         Elsif (Sum3 >= Mopm  And Sum2 >= Mopm And Total_Cust>Pd) Then
            Update Claim_Line Set Amount_Copay = Diff10 Where User_Id=U_Id And Policy_Id=Pol_Id
And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
            Update Claim_Line Set Amount_of_Coinsurance = 0 Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
            Update Claim_Line Set Amount_Paid_Bycustomer = Diff10 Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
            Update Claim_Line Set Amount_Paid_Byinsurance = Diff11 Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
            update claim_line set amount_deductable = 0 where user_id=u_id and policy_id=pol_id and
service_id=serv_id and sp_id=prov_id and service_date=date_of_service;
```

```
            update claim_line set status = 'Accept' where user_id=u_id and policy_id=pol_id and
service_id=serv_id and sp_id=prov_id and service_date=date_of_service;
            update claim_line set cid=2 where user_id=u_id and policy_id=pol_id and service_id=serv_id and
sp_id=prov_id and service_date=date_of_service;
        Return 2;
        -----------if after adding coinsurance, patient pays more than max out of pocket and less if only copay
is paid
                Elsif(Sum3>Mopm And Sum2 < Mopm And Total_Cust>Pd) Then
            Update Claim_Line Set Amount_Copay = Inc Where User_Id=U_Id And Policy_Id=Pol_Id And
Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
            Update Claim_Line Set Amount_of_Coinsurance = Diff4 Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
            Update Claim_Line Set Amount_Paid_Bycustomer = Sum4 Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
            Update Claim_Line Set Amount_Paid_Byinsurance = Diff6 Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
            update claim_line set amount_deductable = 0 where user_id=u_id and policy_id=pol_id and
service_id=serv_id and sp_id=prov_id and service_date=date_of_service;
            update claim_line set status = 'Accept' where user_id=u_id and policy_id=pol_id and
service_id=serv_id and sp_id=prov_id and service_date=date_of_service;
            update claim_line set cid=2 where user_id=u_id and policy_id=pol_id and service_id=serv_id and
sp_id=prov_id and service_date=date_of_service;
        return 3;
            --------------------after adding copay and coinsrance total amnt less than max out of pocket
                Elsif(Sum3<=Mopm And Sum2 < Mopm And Total_Cust>Pd) Then
            Update Claim_Line Set Amount_Copay = Inc Where User_Id=U_Id And Policy_Id=Pol_Id And
Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
            Update Claim_Line Set Amount_of_Coinsurance = amnt_coinsurance Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
            Update Claim_Line Set Amount_Paid_Bycustomer = Sum5 Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
            Update Claim_Line Set Amount_Paid_Byinsurance = Diff7 Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
            update claim_line set amount_deductable = 0 where user_id=u_id and policy_id=pol_id and
service_id=serv_id and sp_id=prov_id and service_date=date_of_service;
            update claim_line set status = 'Accept' where user_id=u_id and policy_id=pol_id and
service_id=serv_id and sp_id=prov_id and service_date=date_of_service;
            update claim_line set cid=2 where user_id=u_id and policy_id=pol_id and service_id=serv_id and
sp_id=prov_id and service_date=date_of_service;
 return 4;
        -------------if total cust less than policy deductable and even after current charge it doesnt exist policy
deductable
             Elsif (total_cust < pd and Sum1<= Pd ) Then
        Update Claim_Line Set Amount_Copay = 0 Where User_Id=U_Id And Policy_Id=Pol_Id And
Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
        Update Claim_Line Set Amount_of_Coinsurance = 0 Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
        Update Claim_Line Set Amount_Paid_Bycustomer = Sew Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
        Update Claim_Line Set Amount_Paid_Byinsurance = 0 Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
        update claim_line set status = 'Accept' where user_id=u_id and policy_id=pol_id and
service_id=serv_id and sp_id=prov_id and service_date=date_of_service;
        update claim_line set cid=2 where user_id=u_id and policy_id=pol_id and service_id=serv_id and
sp_id=prov_id and service_date=date_of_service;
        if (sew > diff8) then
        update claim_line set amount_deductable = (sew-diff8) where user_id=u_id and policy_id=pol_id
and service_id=serv_id and sp_id=prov_id and service_date=date_of_service;
        else
        update claim_line set amount_deductable = (diff8-sew) where user_id=u_id and policy_id=pol_id
and service_id=serv_id and sp_id=prov_id and service_date=date_of_service;
        end if;
        Return 5;
      ------if total cust less than policy deductable and after current charge, it exceeds policy deductable
         Elsif(Total_Cust <Pd And Sum6 > Pd ) then
        if(var1<inc ) then
        Update Claim_Line Set Amount_Copay = var1 Where User_Id=U_Id And Policy_Id=Pol_Id And
Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
        Update Claim_Line Set Amount_of_Coinsurance = 0 Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
        Update Claim_Line Set Amount_Paid_Bycustomer = Sum20 Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
        Update Claim_Line Set Amount_Paid_Byinsurance = 0 Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
        update claim_line set amount_deductable = 0 where user_id=u_id and policy_id=pol_id and
service_id=serv_id and sp_id=prov_id and service_date=date_of_service;
        update claim_line set status = 'Accept' where user_id=u_id and policy_id=pol_id and
service_id=serv_id and sp_id=prov_id and service_date=date_of_service;
        update claim_line set cid=2 where user_id=u_id and policy_id=pol_id and service_id=serv_id and
sp_id=prov_id and service_date=date_of_service;
        return 6;
        elsif (var1 > inc and var2 < amnt_coinsurance) then
        Update Claim_Line Set Amount_Copay = inc Where User_Id=U_Id And Policy_Id=Pol_Id And
Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
        Update Claim_Line Set Amount_of_Coinsurance = var2 Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
```

```
          Update Claim_Line Set Amount_Paid_Bycustomer = Sum22 Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
          Update Claim_Line Set Amount_Paid_Byinsurance = 0 Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
          update claim_line set amount_deductable = 0 where user_id=u_id and policy_id=pol_id and
service_id=serv_id and sp_id=prov_id and service_date=date_of_service;
          update claim_line set status = 'Accept' where user_id=u_id and policy_id=pol_id and
service_id=serv_id and sp_id=prov_id and service_date=date_of_service;
          update claim_line set cid=2 where user_id=u_id and policy_id=pol_id and service_id=serv_id and
sp_id=prov_id and service_date=date_of_service;
          Return 16;
          else
          Update Claim_Line Set Amount_Copay = inc Where User_Id=U_Id And Policy_Id=Pol_Id And
Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
          Update Claim_Line Set Amount_of_Coinsurance = Calc1 Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
          Update Claim_Line Set Amount_Paid_Bycustomer = Sum8 Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
          Update Claim_Line Set Amount_Paid_Byinsurance = Diff9 Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
          update claim_line set amount_deductable = 0 where user_id=u_id and policy_id=pol_id and
service_id=serv_id and sp_id=prov_id and service_date=date_of_service;
          update claim_line set status = 'Accept' where user_id=u_id and policy_id=pol_id and
service_id=serv_id and sp_id=prov_id and service_date=date_of_service;
          update claim_line set cid=2 where user_id=u_id and policy_id=pol_id and service_id=serv_id and
sp_id=prov_id and service_date=date_of_service;
          Return 17;
        end if;
  End If;
-------checks for out network
 Elsif(Serv_Prov_Type='Out-network') Then
  Select Out_Network_Copay Into Onc From Policy,Plan,Coverage Where Policy.Plan_Id=Plan.Plan_Id
And Plan.Coverage_Id=Coverage.Coverage_Id And Policy.Policy_Id=Pol_Id;
  Select Out_Network_Coinsurance Into Onci From Policy,Plan,Coverage Where
Policy.Plan_Id=Plan.Plan_Id And Plan.Coverage_Id=Coverage.Coverage_Id And
Policy.Policy_Id=Pol_Id;
  Amnt_Copay:=Onc;
  Amnt_Coinsurance:=((Sew-Amnt_Copay)*Onci)/100;
  Sum2:=Amnt_Copay+Total_Cust;
  Sum3:=Amnt_Coinsurance+Sum2;
  Diff4:=Sum3-Mopm;
  Sum4:=Onc+Diff4;
  Diff3:=Mopm-Sum2;
  Diff5:=Mopm - Sum2;
  Diff6:=Sew-Sum4;
  Sum5:=amnt_coinsurance+Onc;
  Diff7:=Sew-Sum5;
  Sum6:=Total_Cust+Sew;
  Diff8:=Pd-Total_Cust;
  diff20:=sew-diff8;
  Sum7:=Diff8+Onc;
  Calc1:=((Sew-Sum7)*Onci)/100;
  Sum8:=Onc+Calc1+Diff8;
  Diff9:=Sew-Sum8;
  Diff10:=Mopm-Total_Cust;
  Diff11:=Sew-Diff10;
  var1:=sum1-pd;
  sum20:= var1+diff8;
  sum21 := diff8 + onc;
  var2 := sew - sum21;
  sum22 := sum21 + var2;
        ----------------if total paid by customer already exceeds max out of pocket expense
            If (Total_Cust >= Mopm And Total_Cust>Pd) Then
        Update Claim_Line Set Amount_Copay = 0 Where User_Id=U_Id And Policy_Id=Pol_Id And
Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
        Update Claim_Line Set Amount_of_Coinsurance = 0 Where User_Id=U_Id And Policy_Id=Pol_Id
And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
        Update Claim_Line Set Amount_Paid_Bycustomer = 0 Where User_Id=U_Id And Policy_Id=Pol_Id
And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
        Update Claim_Line Set Amount_Paid_Byinsurance = Sew Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
        update claim_line set amount_deductable = 0 where user_id=u_id and policy_id=pol_id and
service_id=serv_id and sp_id=prov_id and service_date=date_of_service;
        update claim_line set status = 'Accept' where user_id=u_id and policy_id=pol_id and
service_id=serv_id and sp_id=prov_id and service_date=date_of_service;
        update claim_line set cid=2 where user_id=u_id and policy_id=pol_id and service_id=serv_id and
sp_id=prov_id and service_date=date_of_service;
        Return 8;
        ---------------------if after current charge, coinsurance and copay, it exceeds max out of pocket
expense
            Elsif (Sum3 >= Mopm And Sum2 >= Mopm And Total_Cust>Pd) Then
          Update Claim_Line Set Amount_Copay = Diff10 Where User_Id=U_Id And Policy_Id=Pol_Id
And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
          Update Claim_Line Set Amount_of_Coinsurance = 0 Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
          Update Claim_Line Set Amount_Paid_Bycustomer = Diff10 Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
```

```
          Update Claim_Line Set Amount_Paid_Byinsurance = Diff11 Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
          update claim_line set amount_deductable = 0 where user_id=u_id and policy_id=pol_id and
service_id=serv_id and sp_id=prov_id and service_date=date_of_service;
          update claim_line set status = 'Accept' where user_id=u_id and policy_id=pol_id and
service_id=serv_id and sp_id=prov_id and service_date=date_of_service;
          update claim_line set cid=2 where user_id=u_id and policy_id=pol_id and service_id=serv_id and
sp_id=prov_id and service_date=date_of_service;
       Return 9;
           -----------------if after coinsurance, person pays more than max out of pocket but not with copay
       Elsif(Sum3>Mopm And Sum2 < Mopm And Total_Cust>Pd) Then
          Update Claim_Line Set Amount_Copay = Onc Where User_Id=U_Id And Policy_Id=Pol_Id And
Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
          Update Claim_Line Set Amount_of_Coinsurance = Diff4 Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
          Update Claim_Line Set Amount_Paid_Bycustomer = Sum4 Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
          Update Claim_Line Set Amount_Paid_Byinsurance = Diff6 Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
          update claim_line set amount_deductable = 0 where user_id=u_id and policy_id=pol_id and
service_id=serv_id and sp_id=prov_id and service_date=date_of_service;
          update claim_line set status = 'Accept' where user_id=u_id and policy_id=pol_id and
service_id=serv_id and sp_id=prov_id and service_date=date_of_service;
          update claim_line set cid=2 where user_id=u_id and policy_id=pol_id and service_id=serv_id and
sp_id=prov_id and service_date=date_of_service;
          return 10;
           -------------check if after current charge, copay and coinsurance, it exceed total max out of pocket
and it already exceeds policy deductable
       Elsif(Sum3<=Mopm And Sum2 < Mopm And Total_Cust>Pd) Then
          Update Claim_Line Set Amount_Copay = Onc Where User_Id=U_Id And Policy_Id=Pol_Id And
Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
          Update Claim_Line Set Amount_of_Coinsurance = amnt_coinsurance Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
          Update Claim_Line Set Amount_Paid_Bycustomer = Sum5 Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
          Update Claim_Line Set Amount_Paid_Byinsurance = Diff7 Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
          update claim_line set amount_deductable = 0 where user_id=u_id and policy_id=pol_id and
service_id=serv_id and sp_id=prov_id and service_date=date_of_service;
          update claim_line set status = 'Accept' where user_id=u_id and policy_id=pol_id and
service_id=serv_id and sp_id=prov_id and service_date=date_of_service;
          update claim_line set cid=2 where user_id=u_id and policy_id=pol_id and service_id=serv_id and
sp_id=prov_id and service_date=date_of_service;
          return 11;
```

```
    ------------if total cust less than policy deductable and after current charge more than policy
deductable
         Elsif (total_cust < pd and Sum1<= Pd ) Then
        Update Claim_Line Set Amount_Copay = 0 Where User_Id=U_Id And Policy_Id=Pol_Id And
Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
        Update Claim_Line Set Amount_of_Coinsurance = 0 Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
        Update Claim_Line Set Amount_Paid_Bycustomer = Sew Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
        Update Claim_Line Set Amount_Paid_Byinsurance = 0 Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
        update claim_line set status = 'Accept' where user_id=u_id and policy_id=pol_id and
service_id=serv_id and sp_id=prov_id and service_date=date_of_service;
        update claim_line set cid=2 where user_id=u_id and policy_id=pol_id and service_id=serv_id and
sp_id=prov_id and service_date=date_of_service;
        if (sew > diff8) then
        update claim_line set amount_deductable = (sew-diff8) where user_id=u_id and policy_id=pol_id
and service_id=serv_id and sp_id=prov_id and service_date=date_of_service;
        else
        update claim_line set amount_deductable = (diff8-sew) where user_id=u_id and policy_id=pol_id
and service_id=serv_id and sp_id=prov_id and service_date=date_of_service;
        end if;
        Return 12;
    ------if total cust less than policy deductable and after current charge, it exceeds policy deductable
         Elsif(Total_Cust <Pd And Sum1 > Pd) then
        if(var1<onc and var2 < onci) then
        Update Claim_Line Set Amount_Copay = var1 Where User_Id=U_Id And Policy_Id=Pol_Id And
Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
        Update Claim_Line Set Amount_of_Coinsurance = 0 Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
        Update Claim_Line Set Amount_Paid_Bycustomer = sum20 Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
        Update Claim_Line Set Amount_Paid_Byinsurance = 0 Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
        update claim_line set amount_deductable = 0 where user_id=u_id and policy_id=pol_id and
service_id=serv_id and sp_id=prov_id and service_date=date_of_service;
        update claim_line set status = 'Accept' where user_id=u_id and policy_id=pol_id and
service_id=serv_id and sp_id=prov_id and service_date=date_of_service;
        update claim_line set cid=2 where user_id=u_id and policy_id=pol_id and service_id=serv_id and
sp_id=prov_id and service_date=date_of_service;
        return 13;
        elsif (var1 > onc and var2 < amnt_coinsurance) then
        Update Claim_Line Set Amount_Copay = Onc Where User_Id=U_Id And Policy_Id=Pol_Id And
Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
```

```
	Update Claim_Line Set Amount_of_Coinsurance = var2 Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
	Update Claim_Line Set Amount_Paid_Bycustomer = Sum22 Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
	Update Claim_Line Set Amount_Paid_Byinsurance = 0 Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
	update claim_line set amount_deductable = 0 where user_id=u_id and policy_id=pol_id and
service_id=serv_id and sp_id=prov_id and service_date=date_of_service;
	update claim_line set status = 'Accept' where user_id=u_id and policy_id=pol_id and
service_id=serv_id and sp_id=prov_id and service_date=date_of_service;
	update claim_line set cid=2 where user_id=u_id and policy_id=pol_id and service_id=serv_id and
sp_id=prov_id and service_date=date_of_service;
	Return 14;
	else
	Update Claim_Line Set Amount_Copay = Onc Where User_Id=U_Id And Policy_Id=Pol_Id And
Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
	Update Claim_Line Set Amount_of_Coinsurance = Calc1 Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
	Update Claim_Line Set Amount_Paid_Bycustomer = Sum8 Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
	Update Claim_Line Set Amount_Paid_Byinsurance = Diff9 Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
	update claim_line set amount_deductable = 0 where user_id=u_id and policy_id=pol_id and
service_id=serv_id and sp_id=prov_id and service_date=date_of_service;
	update claim_line set status = 'Accept' where user_id=u_id and policy_id=pol_id and
service_id=serv_id and sp_id=prov_id and service_date=date_of_service;
	update claim_line set cid=2 where user_id=u_id and policy_id=pol_id and service_id=serv_id and
sp_id=prov_id and service_date=date_of_service;
	Return 15;
end if;
else
return -2;
  End If;
End If;

Exception
When Others Then
Return -1;
End;

-----------------------------------------------Function 15-----------------------------------------------
create or replace Function F9_5_6_7_Pay_DEPD (Pol_Id In Number,Serv_Id In Number,Prov_Id In
Number,Amnt In Number, Patient_Name In Varchar, Date_Of_Service In Date)
Return Number
```

```
Is

--variable declaration
sum20 integer;
var1 integer;
onci integer;
PD integer;
sew integer;
INCI integer;
U_Id integer;
Total_Cust integer;
Serv_Prov_Type Varchar(255);
Difference1 integer;
Mopm integer;
Onc integer;
Inc integer;
Sum1 integer;
Diff2 integer;
AMNT_copay integer;
amnt_coinsurance integer;
sum2 integer;
sum3 integer;
DIFF8 integer;
DIFF7 integer;
sum6 integer;
sum5 integer;
diff4 integer;
diff5 integer;
diff6 integer;
diff3 integer;
total integer;
sum4 integer;
sum7 integer;
calc1 integer;
sum8 integer;
diff9 integer;
diff10 integer;
diff11 integer;
amount_coinsurance integer;
sum21 integer;
var2 integer;
sum22 integer;
diff20 integer;
Begin
```

```
----select queries to fetch user id, maximum out of pocket, total amount paid by customer till date, service
provider type, plan deductable and providers charge

Select Userid Into U_Id From Dependent Where d_Name=Patient_Name;
Select Max_Opc_Permember Into Mopm From Plan, Policy Where Plan.Plan_Id=Policy.Plan_Id And
Policy.Policy_Id=Pol_Id;
Select Sum(Amount_Paid_Bycustomer) Into Total_Cust From Claim_Line Where User_Id=U_Id;
Select Sp_Type Into Serv_Prov_Type From Service_Provider Where Sp_Id=Prov_Id And
Service_Id=Serv_Id;
Select Plan.Deductable_Amount Into Pd From Plan,Policy Where Policy.Plan_Id=Plan.Plan_Id And
Policy.Policy_Id=Pol_Id;
Select Providers_Charge Into Sew From Claim_Line Where User_Id=U_Id And Policy_Id=Pol_Id And
Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
-----computes various values like checking the total paid by customer and current charge

Sum1:= Total_Cust+Sew;

----checks mam out of pocket and how much customer paid

Difference1 := Mopm - Total_Cust;
Diff2 := Mopm - Total_Cust;
---checks the provider type

 If(Serv_Prov_Type = 'In-network') Then
  Select In_Network_Copay Into Inc From Policy,Plan,Coverage Where Policy.Plan_Id=Plan.Plan_Id
And Plan.Coverage_Id=Coverage.Coverage_Id And Policy.Policy_Id=Pol_Id;
  Select In_Network_Coinsurance Into Inci From Policy,Plan,Coverage Where
Policy.Plan_Id=Plan.Plan_Id And Plan.Coverage_Id=Coverage.Coverage_Id And
Policy.Policy_Id=Pol_Id;
  Amnt_Copay:=Inc;
  Amnt_Coinsurance:=((Sew-Amnt_Copay)*Inci)/100;
  Sum2:=Amnt_Copay+Total_Cust;
  Sum3:=Amnt_Coinsurance+Sum2;
  Diff4:=Sum3-Mopm;
  Sum4:=Inc+Diff4;
  Diff3:= Mopm-Sum2;
  Diff5:=Mopm - Sum2;
  Diff6:=Sew-Sum4;
  Sum5:=amnt_coinsurance+Inc;
  Diff7:=Sew-Sum5;
  Sum6:=Total_Cust+Sew;
  Diff8:=Pd-Total_Cust;
  diff20:=sew-diff8;
  Sum7:=Diff8+Inc;
  Calc1:=((Sew-Sum7)*Inci)/100;
  Sum8:=Inc+Calc1+Diff8;
  Diff9:=Sew-Sum8;
  Diff10:=Mopm-Total_Cust;
  Diff11:=Sew-Diff10;
  var1:=sum1-pd;
  sum20:= var1+diff8;
  sum21 := diff8 + inc;
  var2 := sew - sum21;
  sum22 := sum21 + var2;
               ---------checks if total paid by customer > max out of pocket and policy deductable

             If (Total_Cust >= Mopm And Total_Cust>Pd) Then
         Update Claim_Line Set Amount_Copay = 0 Where User_Id=U_Id And Policy_Id=Pol_Id And
Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
         Update Claim_Line Set Amount_of_Coinsurance = 0 Where User_Id=U_Id And Policy_Id=Pol_Id
And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
         Update Claim_Line Set Amount_Paid_Bycustomer = 0 Where User_Id=U_Id And Policy_Id=Pol_Id
And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
         Update Claim_Line Set Amount_Paid_Byinsurance = Sew Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
         update claim_line set amount_deductable = 0 where user_id=u_id and policy_id=pol_id and
service_id=serv_id and sp_id=prov_id and service_date=date_of_service;
         update claim_line set status = 'Accept' where user_id=u_id and policy_id=pol_id and
service_id=serv_id and sp_id=prov_id and service_date=date_of_service;
         update claim_line set cid=2 where user_id=u_id and policy_id=pol_id and service_id=serv_id and
sp_id=prov_id and service_date=date_of_service;
         Return 1;
---------checks if current copay, coinsurance more than max out of pocket expense and more than policy
deductable

         Elsif (Sum3 >= Mopm  And Sum2 >= Mopm And Total_Cust>Pd) Then
             Update Claim_Line Set Amount_Copay = Diff10 Where User_Id=U_Id And Policy_Id=Pol_Id
And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
             Update Claim_Line Set Amount_of_Coinsurance = 0 Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
             Update Claim_Line Set Amount_Paid_Bycustomer = Diff10 Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
             Update Claim_Line Set Amount_Paid_Byinsurance = Diff11 Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
             update claim_line set amount_deductable = 0 where user_id=u_id and policy_id=pol_id and
service_id=serv_id and sp_id=prov_id and service_date=date_of_service;
             update claim_line set status = 'Accept' where user_id=u_id and policy_id=pol_id and
service_id=serv_id and sp_id=prov_id and service_date=date_of_service;
```

```
            update claim_line set cid=2 where user_id=u_id and policy_id=pol_id and service_id=serv_id and
sp_id=prov_id and service_date=date_of_service;
        Return 2;
                ----------if after adding coinsurance, patient pays more than max out of pocket and less if only
copay is paid

             Elsif(Sum3>Mopm And Sum2 < Mopm And Total_Cust>Pd) Then
            Update Claim_Line Set Amount_Copay = Inc Where User_Id=U_Id And Policy_Id=Pol_Id And
Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
            Update Claim_Line Set Amount_of_Coinsurance = Diff4 Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
            Update Claim_Line Set Amount_Paid_Bycustomer = Sum4 Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
            Update Claim_Line Set Amount_Paid_Byinsurance = Diff6 Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
            update claim_line set amount_deductable = 0 where user_id=u_id and policy_id=pol_id and
service_id=serv_id and sp_id=prov_id and service_date=date_of_service;
            update claim_line set status = 'Accept' where user_id=u_id and policy_id=pol_id and
service_id=serv_id and sp_id=prov_id and service_date=date_of_service;
            update claim_line set cid=2 where user_id=u_id and policy_id=pol_id and service_id=serv_id and
sp_id=prov_id and service_date=date_of_service;
        return 3;
                     -------------------after adding copay and coinsrance total amnt less than max out of pocket

        Elsif(Sum3<=Mopm And Sum2 < Mopm And Total_Cust>Pd) Then
            Update Claim_Line Set Amount_Copay = Inc Where User_Id=U_Id And Policy_Id=Pol_Id And
Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
            Update Claim_Line Set Amount_of_Coinsurance = amnt_coinsurance Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
            Update Claim_Line Set Amount_Paid_Bycustomer = Sum5 Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
            Update Claim_Line Set Amount_Paid_Byinsurance = Diff7 Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
            update claim_line set amount_deductable = 0 where user_id=u_id and policy_id=pol_id and
service_id=serv_id and sp_id=prov_id and service_date=date_of_service;
            update claim_line set status = 'Accept' where user_id=u_id and policy_id=pol_id and
service_id=serv_id and sp_id=prov_id and service_date=date_of_service;
            update claim_line set cid=2 where user_id=u_id and policy_id=pol_id and service_id=serv_id and
sp_id=prov_id and service_date=date_of_service;

 return 4;
        ------------if total cust less than policy deductable and even after current charge it doesnt exist policy
deductable
```

```
        Elsif (total_cust < pd and Sum1<= Pd ) Then
            Update Claim_Line Set Amount_Copay = 0 Where User_Id=U_Id And Policy_Id=Pol_Id And
Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
            Update Claim_Line Set Amount_of_Coinsurance = 0 Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
            Update Claim_Line Set Amount_Paid_Bycustomer = Sew Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
            Update Claim_Line Set Amount_Paid_Byinsurance = 0 Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
            update claim_line set status = 'Accept' where user_id=u_id and policy_id=pol_id and
service_id=serv_id and sp_id=prov_id and service_date=date_of_service;
            update claim_line set cid=2 where user_id=u_id and policy_id=pol_id and service_id=serv_id and
sp_id=prov_id and service_date=date_of_service;
            if (sew > diff8) then
            update claim_line set amount_deductable = (sew-diff8) where user_id=u_id and policy_id=pol_id
and service_id=serv_id and sp_id=prov_id and service_date=date_of_service;
            else
            update claim_line set amount_deductable = (diff8-sew) where user_id=u_id and policy_id=pol_id
and service_id=serv_id and sp_id=prov_id and service_date=date_of_service;
            end if;
            Return 5;
                ------if total cust less than policy deductable and after current charge, it exceeds policy
deductable

             Elsif(Total_Cust <Pd And Sum6 > Pd ) then
            if(var1<inc ) then
            Update Claim_Line Set Amount_Copay = var1 Where User_Id=U_Id And Policy_Id=Pol_Id And
Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
            Update Claim_Line Set Amount_of_Coinsurance = 0 Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
            Update Claim_Line Set Amount_Paid_Bycustomer = Sum20 Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
            Update Claim_Line Set Amount_Paid_Byinsurance = 0 Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
            update claim_line set amount_deductable = 0 where user_id=u_id and policy_id=pol_id and
service_id=serv_id and sp_id=prov_id and service_date=date_of_service;
            update claim_line set status = 'Accept' where user_id=u_id and policy_id=pol_id and
service_id=serv_id and sp_id=prov_id and service_date=date_of_service;
            update claim_line set cid=2 where user_id=u_id and policy_id=pol_id and service_id=serv_id and
sp_id=prov_id and service_date=date_of_service;
            return 6;
            elsif (var1 > inc and var2 < amnt_coinsurance) then
            Update Claim_Line Set Amount_Copay = inc Where User_Id=U_Id And Policy_Id=Pol_Id And
Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
```

```
            Update Claim_Line Set Amount_of_Coinsurance = var2 Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
            Update Claim_Line Set Amount_Paid_Bycustomer = Sum22 Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
            Update Claim_Line Set Amount_Paid_Byinsurance = 0 Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
            update claim_line set amount_deductable = 0 where user_id=u_id and policy_id=pol_id and
service_id=serv_id and sp_id=prov_id and service_date=date_of_service;
            update claim_line set status = 'Accept' where user_id=u_id and policy_id=pol_id and
service_id=serv_id and sp_id=prov_id and service_date=date_of_service;
            update claim_line set cid=2 where user_id=u_id and policy_id=pol_id and service_id=serv_id and
sp_id=prov_id and service_date=date_of_service;
            Return 16;
            else
            Update Claim_Line Set Amount_Copay = inc Where User_Id=U_Id And Policy_Id=Pol_Id And
Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
            Update Claim_Line Set Amount_of_Coinsurance = Calc1 Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
            Update Claim_Line Set Amount_Paid_Bycustomer = Sum8 Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
            Update Claim_Line Set Amount_Paid_Byinsurance = Diff9 Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
            update claim_line set amount_deductable = 0 where user_id=u_id and policy_id=pol_id and
service_id=serv_id and sp_id=prov_id and service_date=date_of_service;
            update claim_line set status = 'Accept' where user_id=u_id and policy_id=pol_id and
service_id=serv_id and sp_id=prov_id and service_date=date_of_service;
            update claim_line set cid=2 where user_id=u_id and policy_id=pol_id and service_id=serv_id and
sp_id=prov_id and service_date=date_of_service;
            Return 17;
        end if;
  End If;
-------checks for out network

 Elsif(Serv_Prov_Type='Out-network') Then
  Select Out_Network_Copay Into Onc From Policy,Plan,Coverage Where Policy.Plan_Id=Plan.Plan_Id
And Plan.Coverage_Id=Coverage.Coverage_Id And Policy.Policy_Id=Pol_Id;
  Select Out_Network_Coinsurance Into Onci From Policy,Plan,Coverage Where
Policy.Plan_Id=Plan.Plan_Id And Plan.Coverage_Id=Coverage.Coverage_Id And
Policy.Policy_Id=Pol_Id;
  Amnt_Copay:=Onc;
  Amnt_Coinsurance:=((Sew-Amnt_Copay)*Onci)/100;
  Sum2:=Amnt_Copay+Total_Cust;
  Sum3:=Amnt_Coinsurance+Sum2;
  Diff4:=Sum3-Mopm;
  Sum4:=Onc+Diff4;
  Diff3:=Mopm-Sum2;
  Diff5:=Mopm - Sum2;
  Diff6:=Sew-Sum4;
  Sum5:=amnt_coinsurance+Onc;
  Diff7:=Sew-Sum5;
  Sum6:=Total_Cust+Sew;
  Diff8:=Pd-Total_Cust;
  diff20:=sew-diff8;
  Sum7:=Diff8+Onc;
  Calc1:=((Sew-Sum7)*Onci)/100;
  Sum8:=Onc+Calc1+Diff8;
  Diff9:=Sew-Sum8;
  Diff10:=Mopm-Total_Cust;
  Diff11:=Sew-Diff10;
  var1:=sum1-pd;
  sum20:= var1+diff8;
  sum21 := diff8 + onc;
  var2 := sew - sum21;
  sum22 := sum21 + var2;
          ----------------if total paid by customer already exceeds max out of pocket expense
      If (Total_Cust >= Mopm And Total_Cust>Pd) Then
      Update Claim_Line Set Amount_Copay = 0 Where User_Id=U_Id And Policy_Id=Pol_Id And
Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
      Update Claim_Line Set Amount_of_Coinsurance = 0 Where User_Id=U_Id And Policy_Id=Pol_Id
And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
      Update Claim_Line Set Amount_Paid_Bycustomer = 0 Where User_Id=U_Id And Policy_Id=Pol_Id
And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
      Update Claim_Line Set Amount_Paid_Byinsurance = Sew Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
      update claim_line set amount_deductable = 0 where user_id=u_id and policy_id=pol_id and
service_id=serv_id and sp_id=prov_id and service_date=date_of_service;
      update claim_line set status = 'Accept' where user_id=u_id and policy_id=pol_id and
service_id=serv_id and sp_id=prov_id and service_date=date_of_service;
      update claim_line set cid=2 where user_id=u_id and policy_id=pol_id and service_id=serv_id and
sp_id=prov_id and service_date=date_of_service;
      Return 8;
              ----------------------if after current charge, coinsurance and copay, it exceeds max out of
pocket expense

      Elsif (Sum3 >= Mopm And Sum2 >= Mopm And Total_Cust>Pd) Then
         Update Claim_Line Set Amount_Copay = Diff10 Where User_Id=U_Id And Policy_Id=Pol_Id
And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
```

```
          Update Claim_Line Set Amount_of_Coinsurance = 0 Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
          Update Claim_Line Set Amount_Paid_Bycustomer = Diff10 Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
          Update Claim_Line Set Amount_Paid_Byinsurance = Diff11 Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
          update claim_line set amount_deductable = 0 where user_id=u_id and policy_id=pol_id and
service_id=serv_id and sp_id=prov_id and service_date=date_of_service;
          update claim_line set status = 'Accept' where user_id=u_id and policy_id=pol_id and
service_id=serv_id and sp_id=prov_id and service_date=date_of_service;
          update claim_line set cid=2 where user_id=u_id and policy_id=pol_id and service_id=serv_id and
sp_id=prov_id and service_date=date_of_service;
       Return 9;
           -----------------if after coinsurance, person pays more than max out of pocket but not with copay

       Elsif(Sum3>Mopm And Sum2 < Mopm And Total_Cust>Pd) Then
          Update Claim_Line Set Amount_Copay = Onc Where User_Id=U_Id And Policy_Id=Pol_Id And
Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
          Update Claim_Line Set Amount_of_Coinsurance = Diff4 Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
          Update Claim_Line Set Amount_Paid_Bycustomer = Sum4 Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
          Update Claim_Line Set Amount_Paid_Byinsurance = Diff6 Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
          update claim_line set amount_deductable = 0 where user_id=u_id and policy_id=pol_id and
service_id=serv_id and sp_id=prov_id and service_date=date_of_service;
          update claim_line set status = 'Accept' where user_id=u_id and policy_id=pol_id and
service_id=serv_id and sp_id=prov_id and service_date=date_of_service;
          update claim_line set cid=2 where user_id=u_id and policy_id=pol_id and service_id=serv_id and
sp_id=prov_id and service_date=date_of_service;
          return 10;
           -------------check if after current charge, copay and coinsurance, it exceed total max out of pocket
and it already exceeds policy deductable

       Elsif(Sum3<=Mopm And Sum2 < Mopm And Total_Cust>Pd) Then
          Update Claim_Line Set Amount_Copay = Onc Where User_Id=U_Id And Policy_Id=Pol_Id And
Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
          Update Claim_Line Set Amount_of_Coinsurance = amnt_coinsurance Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
          Update Claim_Line Set Amount_Paid_Bycustomer = Sum5 Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
          Update Claim_Line Set Amount_Paid_Byinsurance = Diff7 Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
```

```
            update claim_line set amount_deductable = 0 where user_id=u_id and policy_id=pol_id and
service_id=serv_id and sp_id=prov_id and service_date=date_of_service;
            update claim_line set status = 'Accept' where user_id=u_id and policy_id=pol_id and
service_id=serv_id and sp_id=prov_id and service_date=date_of_service;
            update claim_line set cid=2 where user_id=u_id and policy_id=pol_id and service_id=serv_id and
sp_id=prov_id and service_date=date_of_service;
            return 11;
        ------------if total cust less than policy deductable and after current charge more than policy
deductable

Elsif (total_cust < pd and Sum1<= Pd ) Then
            Update Claim_Line Set Amount_Copay = 0 Where User_Id=U_Id And Policy_Id=Pol_Id And
Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
            Update Claim_Line Set Amount_of_Coinsurance = 0 Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
            Update Claim_Line Set Amount_Paid_Bycustomer = Sew Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
            Update Claim_Line Set Amount_Paid_Byinsurance = 0 Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
            update claim_line set status = 'Accept' where user_id=u_id and policy_id=pol_id and
service_id=serv_id and sp_id=prov_id and service_date=date_of_service;
            update claim_line set cid=2 where user_id=u_id and policy_id=pol_id and service_id=serv_id and
sp_id=prov_id and service_date=date_of_service;
            if (sew > diff8) then
            update claim_line set amount_deductable = (sew-diff8) where user_id=u_id and policy_id=pol_id
and service_id=serv_id and sp_id=prov_id and service_date=date_of_service;
            else
            update claim_line set amount_deductable = (diff8-sew) where user_id=u_id and policy_id=pol_id
and service_id=serv_id and sp_id=prov_id and service_date=date_of_service;
            end if;
            Return 12;
        ------if total cust less than policy deductable and after current charge, it exceeds policy deductable

Elsif(Total_Cust <Pd And Sum1 > Pd) then
            if(var1<onc and var2 < onci) then
            Update Claim_Line Set Amount_Copay = var1 Where User_Id=U_Id And Policy_Id=Pol_Id And
Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
            Update Claim_Line Set Amount_of_Coinsurance = 0 Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
            Update Claim_Line Set Amount_Paid_Bycustomer = sum20 Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
            Update Claim_Line Set Amount_Paid_Byinsurance = 0 Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
```

```
            update claim_line set amount_deductable = 0 where user_id=u_id and policy_id=pol_id and
service_id=serv_id and sp_id=prov_id and service_date=date_of_service;
            update claim_line set status = 'Accept' where user_id=u_id and policy_id=pol_id and
service_id=serv_id and sp_id=prov_id and service_date=date_of_service;
            update claim_line set cid=2 where user_id=u_id and policy_id=pol_id and service_id=serv_id and
sp_id=prov_id and service_date=date_of_service;
            return 13;
            elsif (var1 > onc and var2 < amnt_coinsurance) then
            Update Claim_Line Set Amount_Copay = Onc Where User_Id=U_Id And Policy_Id=Pol_Id And
Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
            Update Claim_Line Set Amount_of_Coinsurance = var2 Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
            Update Claim_Line Set Amount_Paid_Bycustomer = Sum22 Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
            Update Claim_Line Set Amount_Paid_Byinsurance = 0 Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
            update claim_line set amount_deductable = 0 where user_id=u_id and policy_id=pol_id and
service_id=serv_id and sp_id=prov_id and service_date=date_of_service;
            update claim_line set status = 'Accept' where user_id=u_id and policy_id=pol_id and
service_id=serv_id and sp_id=prov_id and service_date=date_of_service;
            update claim_line set cid=2 where user_id=u_id and policy_id=pol_id and service_id=serv_id and
sp_id=prov_id and service_date=date_of_service;
            Return 14;
            else
            Update Claim_Line Set Amount_Copay = Onc Where User_Id=U_Id And Policy_Id=Pol_Id And
Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
            Update Claim_Line Set Amount_of_Coinsurance = Calc1 Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
            Update Claim_Line Set Amount_Paid_Bycustomer = Sum8 Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
            Update Claim_Line Set Amount_Paid_Byinsurance = Diff9 Where User_Id=U_Id And
Policy_Id=Pol_Id And Service_Id=Serv_Id And Sp_Id=Prov_Id And Service_Date=Date_Of_Service;
            update claim_line set amount_deductable = 0 where user_id=u_id and policy_id=pol_id and
service_id=serv_id and sp_id=prov_id and service_date=date_of_service;
            update claim_line set status = 'Accept' where user_id=u_id and policy_id=pol_id and
service_id=serv_id and sp_id=prov_id and service_date=date_of_service;
            update claim_line set cid=2 where user_id=u_id and policy_id=pol_id and service_id=serv_id and
sp_id=prov_id and service_date=date_of_service;
            Return 15;
end if;
else
return -2;
  End If;
End If;
```

```
Exception
When Others Then
Return -1;
End;

----------------------------------------------Function 16----------------------------------------------
create or replace function feature_message_9_8_cust(Pol_Id In Number,Serv_Id In Number,Prov_Id In
Number,Amnt In Number, Patient_Name In Varchar, Date_Of_Service In Date)
return number
IS
msg_id message.message_id%type;
u_id integer;
serv_desc varchar(255);
amnt_servchrg integer;
pc integer;
amnt_cp integer;
amnt_ded integer;
amnt_coinc integer;
amnt_bycust integer;
amnt_byins integer;
begin
-----fetches data from claim line table for the accepted claim and inserts in message table
Select Userid Into U_Id From Customer Where Cust_Name=Patient_Name;
select service_description into serv_desc from service where service_id=serv_id;
select allowed_service_charges into amnt_servchrg from coverage where service_id=serv_id;
select providers_charge into pc from claim_line where policy_id=pol_id and sp_id=prov_id and
service_date=date_of_service and user_id=u_id;
select amount_copay into amnt_cp from claim_line where policy_id=pol_id and sp_id=prov_id and
service_date=date_of_service and user_id=u_id ;
select amount_deductable into amnt_ded from claim_line where policy_id=pol_id and sp_id=prov_id and
service_date=date_of_service and user_id=u_id;
select amount_of_coinsurance into amnt_coinc from claim_line where policy_id=pol_id and
sp_id=prov_id and service_date=date_of_service and user_id=u_id;
select amount_paid_bycustomer into amnt_bycust from claim_line where policy_id=pol_id and
sp_id=prov_id and service_date=date_of_service and user_id=u_id;
select amount_paid_byinsurance into amnt_byins from claim_line where policy_id=pol_id and
sp_id=prov_id and service_date=date_of_service and user_id=u_id;
insert into message values (message_seq.nextval,' service identification number ' || serv_id ||' service
description ' || serv_desc || ' allowed service charge ' || amnt_servchrg || ' providers charge ' || pc || ' amount
copay ' || amnt_cp ||' amount deductable ' || amnt_ded ||' amount of coinsurance ' || amnt_coinc ||' amount
paid by customer ' || amnt_bycust ||' amount paid by insurance ' || amnt_byins,sysdate,u_id);
update claim_line set message_id = message_seq.currval where policy_id=pol_id and sp_id=prov_id and
service_date=date_of_service and user_id=u_id;
```

```
return 0;

exception
        when others then
        return -1;

End;

----------------------------------------------Function 17----------------------------------------------
create or replace function feature_message_9_8_depd(Pol_Id In Number,Serv_Id In Number,Prov_Id In
Number,Amnt In Number, Patient_Name In Varchar, Date_Of_Service In Date)
return number
IS
msg_id message.message_id%type;
u_id integer;
serv_desc varchar(255);
amnt_servchrg integer;
pc integer;
amnt_cp integer;
amnt_ded integer;
amnt_coinc integer;
amnt_bycust integer;
amnt_byins integer;
begin

-----fetches data from claim line table for the accepted claim and inserts in message table
Select Userid Into U_Id From dependent Where d_name=Patient_Name;
select service_description into serv_desc from service where service_id=serv_id;
select allowed_service_charges into amnt_servchrg from coverage where service_id=serv_id;
select providers_charge into pc from claim_line where policy_id=pol_id and sp_id=prov_id and
service_date=date_of_service and user_id=u_id;
select amount_copay into amnt_cp from claim_line where policy_id=pol_id and sp_id=prov_id and
service_date=date_of_service and user_id=u_id ;
select amount_deductable into amnt_ded from claim_line where policy_id=pol_id and sp_id=prov_id and
service_date=date_of_service and user_id=u_id;
select amount_of_coinsurance into amnt_coinc from claim_line where policy_id=pol_id and
sp_id=prov_id and service_date=date_of_service and user_id=u_id;
select amount_paid_bycustomer into amnt_bycust from claim_line where policy_id=pol_id and
sp_id=prov_id and service_date=date_of_service and user_id=u_id;
select amount_paid_byinsurance into amnt_byins from claim_line where policy_id=pol_id and
sp_id=prov_id and service_date=date_of_service and user_id=u_id;
insert into message values (message_seq.nextval,' service identification number ' || serv_id ||' service
description ' || serv_desc || ' allowed service charge ' || amnt_servchrg || ' providers charge ' || pc || ' amount
copay ' || amnt_cp ||' amount deductable ' || amnt_ded ||' amount of coinsurance ' || amnt_coinc ||' amount
paid by customer ' || amnt_bycust ||' amount paid by insurance ' || amnt_byins,sysdate,u_id);
update claim_line set message_id = message_seq.currval where policy_id=pol_id and sp_id=prov_id and
service_date=date_of_service and user_id=u_id;
return 0;
End;
```

------------Feature 11---

```
create or replace function claimavailable(claimId in int)
return number
is
countofcid claim.cid%type;
begin
select count(*) into countofcid from claim where cid = claimId;
return countofcid;
end;
```

6. Procedures Created

```
-------------------------------Feature 1----------------------------------

create or replace PROCEDURE registration (userType in varchar, name in varchar, address in varchar,
phone in varchar, emailId in varchar, user_pswd in varchar, birthdate date,gender in varchar,cust_userId
in Integer,relation in varchar,provider_type in varchar,serviceid in integer)
IS
user_id integer;
BEGIN
     user_id:= return_userid(emailId, user_pswd);
     if(user_id = -1) then
        if(userType = 'Customer') then
            user_id := return_custuserid(name,emailId,user_pswd,address, phone, birthdate, gender);
            dbms_output.put_line('New User Id of customer is: ' || user_id);
         elsif (userType = 'Dependent') then
            user_id := return_dependuserid(name,emailId,user_pswd,address, phone, birthdate,
gender,cust_userId,relation);
            dbms_output.put_line('New User Id of dependent is: ' || user_id);
          elsif (userType = 'Service Provider') then
           user_id := return_provideruserid(name,emailId,user_pswd,address,
phone,provider_type,serviceid);
           dbms_output.put_line('New User Id of service provider is: ' || user_id);
         End if;

     End if;
End;

---------------Procedure for Feature 2--------------------
----------------------------
Create or replace PROCEDURE login (emailId in varchar, user_pswd in varchar)
IS
user_id integer;
BEGIN
             /* Calling the function to check the login */

     user_id:= check_login(emailId, user_pswd);

End;

----------------------Procedure for Feature 3-------------------
----------------------------
```

```
Create or replace PROCEDURE read_message (user_id in integer,m_date in date) IS /* Fetches the
message for the user with the starting date */

/* Cursor and Variable declaration */

Cursor c1 is select message_body
from message where userid = user_id and message_date >= m_date;
m_body varchar(200);
BEGIN
Open c1;
Loop
        fetch c1 into m_body;
        exit when c1%notfound;

/*Prints the message */

        dbms_output.put_line('Mesage is: ' || m_body);
End loop;
exception
        WHEN OTHERS THEN
        dbms_output.put_line('Incorrect input parameters');
END;

-----------------Procedure for Feature 4-----------------------
---------------------------

create or replace PROCEDURE policy_number (user_id in number, planname in varchar, start_year in
date)
IS

/* Variable Declaration */

pol_id POLICY.POLICY_ID%type;
msg_id message.message_id%type;
usid number;
BEGIN
    select count(userid) into usid from userlogin where UT_ID = 1 and userid = user_id;
    if(usid > 0) then

    /* Calling the function to create policy for the customer.*/

    pol_id:= return_policyid(USER_ID,PLANNAME,START_YEAR);
    else
```

```
      /* Calling the function to create policy for the dependent.*/

       pol_id:= return_policyDepdid(USER_ID,PLANNAME,START_YEAR);
      end if;
     if(pol_id <> -1) then

     /* Calling the function to insert message.*/

        msg_id := insert_msg(user_id, pol_id);

     /* Printing that the message id. */

        dbms_output.put_line('Message is inserted successfully with message id '||msg_id);
     End if;
End;

---------------------------------------Feature 5--------------------------------------------
create or replace procedure add_policydependent (dep_id IN number, pol_id IN number) --function to add
a dependent to the policy dependent table for a user
IS
id number;
user_id userlogin.userid%type;
custid customer.cust_id%type;
msg_id message.message_id%type;
did dependent.d_id%type;
Begin
select d_id into did from dependent where d_id=dep_id;
id := get_existing_depid(dep_id,pol_id); --function call for checking if the dependent already exists for
that policy
if (id= 0) then -- id=0 when an entry exists in the policy dependent table
dbms_output.put_line('Entry already exists.');
elsif(id = -1) then
dbms_output.put_line('Creating new entry');
    select cust_id into custid from dependent where d_id= dep_id;
    select userid into user_id from customer where cust_id=custid;
    insert into policy_dependent values (pol_id,dep_id,user_id);
    msg_id := add_msg (user_id,dep_id,pol_id); --function for inserting the message into the message table
    dbms_output.put_line('Message is inserted successfully with message id '||msg_id);
  End if;
exception
          when no_data_found then
          dbms_output.put_line('No data found');
End;
```

```
--------------------------------------Feature 6----------------------------------------------
create or replace procedure remove_policydependent (dep_name IN varchar, pol_id IN number) --
function to remove a dependent to the policy dependent table for a user
IS
id number;
user_id userlogin.userid%type;
custid customer.cust_id%type;
msg_id message.message_id%type;
dep_id dependent.d_id%type;
Begin
select d_id into dep_id from dependent where d_name=dep_name;
id := get_existing_deptid(dep_id,pol_id); --function call for checking if the dependent already exists for
that policy
if (id = 0) then --remove the entry based on the response from the user
dbms_output.put_line('Removing entry');
   select cust_id into custid from dependent where d_id= dep_id;
   select userid into user_id from customer where cust_id=custid;
   delete from policy_dependent where d_id=dep_id and policy_id=pol_id and userid=user_id;
   msg_id := add_msg1 (user_id,dep_id,pol_id); --function for inserting the message into the message
table
   dbms_output.put_line('Message is inserted successfully with message id '||msg_id);
elsif(id = -1) then
         dbms_output.put_line('Entry does not exist.');
  End if;
exception
         when no_data_found then
         dbms_output.put_line('Invalid Dependent_name or policy_id');
End;

----------------Procedure for Feature 7--------------------------------
----------------------------
Create or replace PROCEDURE cal_premium (pol_id in number)
IS
premium_amt PREMIUM.PREMIUM_AMOUNT%type;
BEGIN
         /* Calling the function to calculate the premium amount. */

    premium_amt:= calculate_premium (pol_id);

End;



--------------------------------------Feature 8----------------------------------------------
create or replace procedure policy_coverage_details_8 (p_id in number,service_desc in varchar) IS
```

```
cursor c1 is
select
coverage.coverage_id,coverage.service_id,coverage.max_service_peryear,coverage.allowed_service_char
ges,coverage.in_network_copay,coverage.in_network_coinsurance,coverage.out_network_copay,coverag
e.out_network_coinsurance from coverage,service,policy where coverage.service_id = service.service_id
and policy.service_id=service.service_id and coverage.coverage_id= policy.coverage_id and
policy.policy_id=p_id and service.service_description like service_desc;
--the above query selects the coverage_id,service_id, max_service_peryear, allowed_service charge,
in_network_copay, in_network_coinsurance,out_network_copay,out_network_coinsurance from
coverage table for the given policy id and string of service description.
c_id coverage.coverage_id%type;
s_id coverage.service_id%type;
max_service coverage.max_service_peryear%type;
allowed_service coverage.allowed_service_charges%type;
in_copay coverage.in_network_copay%type;
in_coinsurance coverage.in_network_coinsurance%type;
out_copay coverage.out_network_copay%type;
out_coinsurance coverage.out_network_coinsurance%type;
--variable declaration
begin
open c1; --opens cursor
loop

fetch c1 into
c_id,s_id,max_service,allowed_service,in_copay,in_coinsurance,out_copay,out_coinsurance; --selects the
above values of c1 and stores into the variable here

if(c_id <> 0 ) then
dbms_output.put_line('coverage id = ' || c_id || ', service id is = ' || s_id||    ', Allowed Service ' ||
allowed_service ||
'In network copay is = ' || in_copay|| 'in network coinsurance is' || in_coinsurance || 'out network copay is '
|| out_copay||
' out network coinsurance is' || out_coinsurance);
else
DBMS_OUTPUT.PUT_LINE('Policy doesnot cover the required service');
end if;
 --print statement
EXIT when c1%notfound;
--print statement


END LOOP; --loop ends

exception
```

```
        WHEN no_data_found THEN
        dbms_output.put_line('');
Close c1; --end of cursor
END;


----------------------------------------Feature 9-------------------------------------------------
create or replace procedure feature9_final_va (
checksp spArrayListType, prov_id in integer, pol_id in integer, patient_name in varchar, date_of_service
in date)
--creating a procedure
IS

--variable declaration

case1 number;
case2 number;
case3 number;
case4 number;
case5 number;
case6 number;
case7 number;
case8 number;
case9 number;
case10 number;
case11 number;
case12 number;
case13 number;
case14 number;
case15 number;
case16 Number;
case17 number;
u_id number;
u_id1 number;
serv_id number;
amnt number;

--procedure begins
Begin

--for loop starts here for varray

FOR i IN 1..checksp.count LOOP
```

```
----here all the functions are called and depending on their return value the entire calculation happens
  serv_id := checksp(i).spid;
  amnt:=checksp(i).serviceAmount;
case1 := feature9_get_existing_provid(prov_id);
case2 := feature9_get_existing_policyid(pol_id);
case3 := feature9_patientcheckincust(patient_name);
case4 := feature9_patientcheckindepd(patient_name);
case5 := feature9_patientpolicy_cust(pol_id,patient_name);
case6 := feature9_patientpolicy_depd(pol_id,patient_name);
case7 := feature9_checkdate(pol_id,date_of_service);
case8 := f9_check_serviceinpolicy_cust(patient_name,serv_id,pol_id);
case9 := f9_check_serviceinpolicy_depd(patient_name,serv_id,pol_id);
case10 := f9_3_check_duplicate_cust(patient_name,serv_id,date_of_service);
case11 := f9_3_check_duplicate_depd(patient_name,serv_id,date_of_service);
/*case12 := f9_4_Adjustcharge_cust(pol_id ,serv_id,prov_id,amnt, patient_name, date_of_service);
case13 := f9_4_adjustcharge_depd(pol_id ,serv_id,prov_id,amnt, patient_name, date_of_service);
case14 := f9_5_6_7_pay_cust(pol_id ,serv_id,prov_id,amnt, patient_name, date_of_service);
case15 := f9_5_6_7_pay_depd(pol_id ,serv_id,prov_id,amnt, patient_name, date_of_service);
case16 := Feature_message_9_8_Cust(pol_id ,serv_id,prov_id,amnt, patient_name, date_of_service);*/

--checks the return value of feature9_get_existing_provid(prov_id) to check if provider exists
if (case1 = -1) then
dbms_output.put_line('Provider doesnt exist');
insert into message values ( message_seq.nextval,'Service provider does not exist',sysdate,0);


--checks the return value of feature9_get_existing_polid(prov_id) to check if policy exists
elsif(case2 = -1) then
        dbms_output.put_line('policy doesnt exist');
insert into message values ( message_seq.nextval,'Policy does not exist',sysdate,0);

--checks the return value of feature9_patientcheckincust(patient_name) and
feature9_patientcheckindepd(patient_name) to check if patient exists
elsif(case3 = -1 and case4 = -1) then
        dbms_output.put_line('patient doesnt exist in customer table');
insert into message values ( message_seq.nextval,'patient doesnt exist in records',sysdate,0);

--checks the return value of feature9_patientpolicy_cust and feature9_patientpolicy_cust to check if
patient linked to policy
elsif(case5 = -1 and case6 = -1) then
        dbms_output.put_line('patient not linked to policy case 5');
insert into message values ( message_seq.nextval,'Patient not linked to policy',sysdate,0);

--checks the return value of feature9_checkdate to check if date of service within allowable range or not
```

```
elsif(case7 = -1) then
            dbms_output.put_line('date of service is outside of the acceptable plan dates');
insert into message values ( message_seq.nextval,'date of service is outside of the acceptable plan
dates',sysdate,0);


elsif(case7=0) then
----checks the return value of f9_check_serviceinpolicy_cust and f9_check_serviceinpolicy_depd to see if
the user exceeds max allowed service or not and if patient linked to service
  if((case8 =2 and case9 = 3) or (case8 = 3 and case9 = 2))then
            dbms_output.put_line('Policy doesnt include mentioned service or more claims than allowed');
insert into message values ( message_seq.nextval,'Policy doesnt include mentioned service or more
claims than allowed',sysdate,0);
    if (case8=-1) then
    select userid into u_id from customer where cust_name=patient_name;
    insert into claim_line
values(claimid_seq.nextval,'Declined',amnt,0,0,0,0,0,message_seq.currval,serv_id,date_of_service,pol_id,
sysdate,u_id,prov_id,2);
    elsif(case9=-1) then
    select userid into u_id from dependent where d_name=patient_name;
    insert into claim_line
values(claimid_seq.nextval,'Declined',amnt,0,0,0,0,0,message_seq.currval,serv_id,date_of_service,pol_id,
sysdate,u_id,prov_id,2);
    end if;
elsif(case8=0 and case9 =0) then

dbms_output.put_line('Exception');
insert into message values ( message_seq.nextval,'Exception',sysdate,0);

------checks for duplicate claim
elsif (case8 =1 or case9 =1) then
  if(case10 = -1 or case11 = -1) then
  dbms_output.put_line('Duplicate claim');
insert into message values ( message_seq.nextval,'Duplicate claim',sysdate,0);
  if (case10 = -1 and case11=0) then
    select userid into u_id from customer where cust_name=patient_name;
            ----inserts into cliam_line for declined status
    insert into claim_line
values(claimid_seq.nextval,'Declined',amnt,0,0,0,0,0,message_seq.currval,serv_id,date_of_service,pol_id,
sysdate,u_id,prov_id,2);
  elsif (case11 = -1 and case10=0) then
    select userid into u_id from dependent where d_name=patient_name;
    --inserts into cliam_line for declined status
```

```
        insert into claim_line
values(claimid_seq.nextval,'Declined',amnt,0,0,0,0,0,message_seq.currval,serv_id,date_of_service,pol_id,
sysdate,u_id,prov_id,2);
  end if;

--calls the f9_4_Adjustcharge_cust and f9_4_Adjustcharge_depd function to adjust the min of charge
elsif(f9_4_Adjustcharge_cust(pol_id ,serv_id,prov_id,amnt, patient_name, date_of_service) = -1 and
f9_4_Adjustcharge_depd(pol_id ,serv_id,prov_id,amnt, patient_name, date_of_service) = -1) then
        dbms_output.put_line('Amount not adjusted');
insert into message values ( message_seq.nextval,'Amount not adjusted',sysdate,0);

--calculates the amount
elsif(f9_5_6_7_pay_cust(pol_id ,serv_id,prov_id,amnt, patient_name, date_of_service) = -1 and
f9_5_6_7_pay_depd(pol_id ,serv_id,prov_id,amnt, patient_name, date_of_service) = -1) then
        dbms_output.put_line('Adjustment and calculation error');
insert into message values ( message_seq.nextval,'Adjustment and calculation error',sysdate,0);

----inserts the message in message table with claim submit details for customer
elsif(Feature_message_9_8_Cust(pol_id ,serv_id,prov_id,amnt, patient_name, date_of_service)= 0) then
dbms_output.put_line('Claim submitted and message inserted into message table with message id' ||
message_seq.currval);


----inserts the message in message table with claim submit details for dependent
elsif(Feature_message_9_8_Cust(pol_id ,serv_id,prov_id,amnt, patient_name, date_of_service) =-1) then
    case17 := Feature_message_9_8_Depd(pol_id ,serv_id,prov_id,amnt, patient_name, date_of_service);
    dbms_output.put_line('Claim submitted and message inserted into message table with message id' ||
message_seq.currval);

else
  dbms_output.put_line('Claim submitted without message id');


end if;
end if;
end if;
end loop;
end;

-------------------Procedure for Feature 10---------------------------------
----------------------------
create or replace procedure search_ClaimDetails (u_id in integer,start_range in date,end_range in date) IS
/* Fetches claim ID, provider, patient name, service date for the given user and date range*/
/* Cursor and variable declaration */

cursor c1 is
select case when exists (select 1 from customer where userid=u_id) then (select cust_name from customer
where userid=u_id) else (select d_name from dependent where userid=u_id) end as PATIENT_NAME,
CL.CLAIM_ID,SP.SP_DESCRIPTION,CLAIM_DATE
FROM CLAIM_LINE CL, SERVICE_PROVIDER SP where CL.sp_id = SP.Sp_ID and (claim_date
BETWEEN start_range AND end_range) AND CL.USER_ID = u_id;
patient_name varchar(100);
claimid CLAIM_LINE.CLAIM_ID%type;
provider_name SERVICE_PROVIDER.SP_DESCRIPTION%type;
claimdate CLAIM_LINE.CLAIM_DATE%type;
myexec EXCEPTION;
begin
open c1;
loop
fetch c1 into patient_name,claimid,provider_name,claimdate;
if (claimid IS NULL AND provider_name IS NULL AND claimdate IS NULL) then
  RAISE myexec;
end if;
exit when c1%NOTFOUND;

/*Prints the message */

dbms_output.put_line('Claim ID = ' || claimid || ', Patient Name is = ' || patient_name || ', Service Provide
Name is = ' || provider_name ||    ', Claim date is = ' || claimdate);
END LOOP;
exception
            WHEN myexec THEN

/* Prints the exception */
   dbms_output.put_line('No Claims for Customer ID ' || u_id || ' for the year range ' || start_range || ' and ' ||
end_range );
Close c1;
END;



----------------------------------------
---------------------------------Feature 11-------------------------

create or replace PROCEDURE claimDetails (claimId in int)
IS
countofcid int;
```

```
cursor c1 is select providers_charge,service_description,
amount_copay,amount_deductable,amount_of_coinsurance, amount_paid_byinsurance,
amount_paid_byCustomer ,service_date from claim_line c join service s on s.service_id  = c.service_id
where cid=claimId;
     pcharge claim_line.providers_charge%type;
     copay claim_line.amount_copay%type;
     deductable claim_line.amount_deductable%type;
     coinsurance claim_line.amount_of_coinsurance%type;
     byinsurance claim_line.amount_paid_byinsurance%type;
     bycustomer claim_line.amount_paid_byCustomer%type;
     sdate claim_line.service_date%type;
     service_name service.service_description%type;
begin
countofcid:= claimavailable(claimId);
if(countofcid <> 0)then
    open c1;
    loop
    fetch  c1 into pcharge,service_name,copay,deductable,coinsurance,byinsurance,bycustomer,sdate;
    exit when c1%notfound;
    dbms_output.put_line('Service date: '||sdate||' Service Name: '||service_name||' Service Provider Charge: 
'||pcharge||' Copay Amount: '||copay||' Deductable Amount: '||deductable||' Coinsurance Amount: '||
    coinsurance||' Amount paid by insurance: '||byinsurance||' Amount paid by customer: '||bycustomer);
    end loop;
    close c1;
    else
    dbms_output.put_line('No such claim present');
    end if;
End;


------------------------Procedure for Feature 12-------------------------------------------------
----------------------------
create or replace PROCEDURE check_totalCost_1 (u_id in integer, planyear IN integer) /* total amount 
paid in a given plan year, the total deductible paid, the total co-pay paid, the total co-insurance paid, the 
total out-of-pocket cost for each member on the plan, and the total out-of-pocket cost for the whole family 
*/
IS

/* Variable Declaration */

totalamtpaid_cust CLAIM_LINE.AMOUNT_PAID_BYCUSTOMER%type;
totaldeductible_cust CLAIM_LINE.AMOUNT_DEDUCTABLE%type;
totalcopay_cust CLAIM_LINE.AMOUNT_COPAY%type;
totalcoinsurance_cust CLAIM_LINE.AMOUNT_OF_COINSURANCE%type;
```

```
opcper_family number;
opcper_member number;
planid PLAN.PLAN_ID%type;
myexec EXCEPTION;

BEGIN
select sum(max_opc_permember) into opcper_member from plan pl
join
policy p
on p.plan_id = pl.plan_id
where extract( year from plan_start_year) = planyear and p.user_id =u_id ;

select
sum(AMOUNT_PAID_BYCUSTOMER),sum(AMOUNT_DEDUCTABLE), sum(AMOUNT_COPAY),
sum(AMOUNT_OF_COINSURANCE), sum(AMOUNT_PAID_BYCUSTOMER) INTO
totalamtpaid_cust,totaldeductible_cust, totalcopay_cust ,totalcoinsurance_cust, opcper_family
FROM CLAIM_LINE WHERE extract (YEAR from claim_date) = planyear AND
CLAIM_LINE.USER_ID = u_id AND CLAIM_LINE.STATUS = 'Accept';

if (totalamtpaid_cust IS NULL AND totaldeductible_cust IS NULL AND totalcopay_cust IS NULL
AND totalcoinsurance_cust IS NULL) then
RAISE myexec;
end if;

/* Printing the values*/

dbms_output.put_line('The total amount paid = ' || totalamtpaid_cust || ', the total deductible paid = ' ||
totaldeductible_cust ||     ', the total co-pay paid = ' || totalcopay_cust
|| ', the total co-insurance paid = ' || totalcoinsurance_cust || ' The out-of-pocket cost per family = ' ||
opcper_family || ' The out-of-pocket cost per member = ' || opcper_member);
exception
        WHEN myexec THEN

/* Printing the exception */

  dbms_output.put_line('No details for customer id ' || u_id || ' for plan year ' || planyear);
END;


-----------------------------------Feature 13-----------------------

create or replace PROCEDURE displayPolicydetails (uid in int)
IS
cursor c1 is select extract(year from plan_start_year)  ,count(p.policy_id)  ,
sum(premium_amount)
from plan pl
join policy po
on
pl.plan_id = po.plan_id
join
PREMIUM p
on
p.policy_id = po.POLICY_ID
where extract(year from plan_start_year) <= 2016 and extract(year from plan_start_year) >= 2001
group by extract(year from plan_start_year);
Last5year int;
totalNoOfPolicy number;
totalAmountpaidpremium int;
begin
   open c1;
   loop
   fetch  c1 into Last5year,totalNoOfPolicy,totalAmountpaidpremium;
   exit when c1%notfound;
   dbms_output.put_line('Year: '||Last5year||', Total No of Policy: '||totalNoOfPolicy||', Total premium
Amount: '||totalAmountpaidpremium);
   end loop;
   close c1;
End;


create or replace PROCEDURE displayClaim (uid in int)
IS
cursor c1 is select extract(year from service_date) , count(claim_id), sum(amount_paid_bycustomer),
sum(cl.AMOUNT_PAID_BYINSURANCE)
from coverage c
join
policy p
on
p.coverage_id = c.coverage_id
join
claim_line cl
on
cl.policy_id = p.policy_id
where extract(year from service_date) <= 2016 and extract(year from service_date)>=2010
group by extract(year from service_date);
Last5year int;
claimCount number;
paidbycustomer number;
```

```
paidbyInsurance number;
begin
   open c1;
   loop
   fetch  c1 into Last5year,claimCount,paidbycustomer,paidbyInsurance;
   exit when c1%notfound;
   dbms_output.put_line('Year: '||Last5year||', Total no of claim: '||claimCount||', Total amount paid by
customer: '||paidbycustomer||', Toal amount paid by Insurance: '||paidbyInsurance);
   end loop;
   close c1;
End;


create or replace PROCEDURE displayResult_feature13 (uid in int)
IS
totalNoOfCustomer number;
totalNoOfSP number;
Last5year number;
totalNoOfPolicy number;
totalAmountpaidpremium int;
Begin
--Display Total Customer
select count(*) into totalNoOfCustomer from userlogin ul
join
usertype ut
on
ut.ut_id = ul.ut_id
and ut.user_type = 'Customer';

--Display in-network service provider
select count(*) into totalNoOfSP from USERLOGIN ul
join
usertype ut
on
ut.ut_id = ul.ut_id
join
service_provider sp
on
sp.userid=ul.userid
where sp.sp_type = 'In-network'
and ut.user_type = 'Service Provider';

dbms_output.put_line('Total No of Customer: '||totalNoOfCustomer||', Total no of in-n/w sp:
'||totalNoOfSP);
```

```
---Display no of  policies, total amount received in past 5 years
displayPolicydetails (uid) ;
--Display total Claims in past 5 years
displayClaim(uid);
exception
when no_data_found then
dbms_output.put_line('No data found');
end;


----------------------------------Procedure for Feature 14---------------------------
---------------------------

create or replace procedure cal_medserv_details_14_1 (year in integer) /* Computes the number of
services appeared in claims each year */
IS

/* Cursor and variable Declaration */

cursor c1 is
select extract(year from service_date),count(service_id),service_id from claim_line
where extract(year from service_date) <= 2017 and extract(year from service_date)>=2013
group by
extract(year from service_date),service_id;
servicedate number;
serviceid CLAIM_LINE.SERVICE_ID%type;
no_of_service integer;
BEGIN
OPEN C1;
LOOP
FETCH C1 into servicedate,no_of_service,serviceid;
exit when c1%NOTFOUND;

/* Printing the data */

dbms_output.put_line('The services id ' || serviceid || ' has appeared ' || no_of_service || ' times' || ' in the
year ' ||servicedate );
END LOOP;
CLOSE C1;
END;


create or replace procedure cal_medserv_details_14_2 (top_k in integer) /* Computes the top K (K as an
integer input) services with the most claims in each year in the past 5 years */
```

```
IS

/* Cursor and variable Declaration */

cursor c2 is
select datec,service_id from
(select count(service_id), service_id ,extract (year from claim_Date) as datec
from Claim_line where (extract(year from claim_date)=extract(year from sysdate)) group by extract(year
from claim_date), service_id order by count(service_id) desc) where rownum<=top_k
UNION ALL
select datec, service_id from
(select count(service_id), service_id ,extract (year from claim_Date) as datec
from Claim_line where (extract(year from claim_date)=extract(year from sysdate)-1) group by
extract(year from claim_date), service_id order by count(service_id) desc) where rownum<=top_k
UNION ALL
select datec, service_id from
(select count(service_id), service_id ,extract (year from claim_Date) as datec
from Claim_line where (extract(year from claim_date)=extract(year from sysdate)-2) group by
extract(year from claim_date), service_id order by count(service_id) desc) where rownum<=top_k
UNION ALL
select datec, service_id from
(select count(service_id), service_id ,extract (year from claim_Date) as datec
from Claim_line where (extract(year from claim_date)=extract(year from sysdate)-3) group by
extract(year from claim_date), service_id order by count(service_id) desc) where rownum<=top_k
UNION ALL
select datec, service_id from
(select count(service_id), service_id ,extract (year from claim_Date) as datec
from Claim_line where (extract(year from claim_date)=extract(year from sysdate)-4) group by
extract(year from claim_date), service_id order by count(service_id) desc) where rownum<=top_k;
serviceid CLAIM_LINE.SERVICE_ID%type;
no_of_service integer;
claimdate number;
BEGIN
OPEN C2;
LOOP
FETCH C2 INTO claimdate, serviceid;
EXIT WHEN C2%NOTFOUND;

/* Printing the data */

dbms_output.put_line('The top services id is ' || serviceid || ' in the year ' || claimdate);
END LOOP;
CLOSE C2;
END;
```

```
create or replace procedure search_medserv_details_14_3 (med_year in integer) IS /* Computes
percentage of patients (customers and their dependents) who have used the service at least once in each
year and the highest percentage of patients each year in past 5 years. */

/* Variable Declaration*/

c integer;
n integer;
PERCENTAGE integer;
begin
 select (n/c)*100 into PERCENTAGE
from ( select sum(amount) as c from
(
select count(cust_id) amount from customer union all select count(d_id) amount from dependent
))
, ( select count(user_id) n from claim_line where ((extract(year from claim_date)=2017)));
dbms_output.put_line('The highest percentage of patient is ' || PERCENTAGE || '%' || ' in the year 2017');

select (n/c)*100 into PERCENTAGE
from ( select sum(amount) as c from
(
select count(cust_id) amount from customer union all select count(d_id) amount from dependent
))
, ( select count(user_id) n from claim_line where ((extract(year from claim_date)=2016)));
dbms_output.put_line('The highest percentage of patient is ' || PERCENTAGE || '%' || ' in the year 2016');

select (n/c)*100 into PERCENTAGE
from ( select sum(amount) as c from
(
select count(cust_id) amount from customer union all select count(d_id) amount from dependent
))
, ( select count(user_id) n from claim_line where ((extract(year from claim_date)=2015)));
dbms_output.put_line('The highest percentage of patient is ' || PERCENTAGE || '%' || ' in the year 2015');

select (n/c)*100 into PERCENTAGE
from ( select sum(amount) as c from
(
select count(cust_id) amount from customer union all select count(d_id) amount from dependent
))
, ( select count(user_id) n from claim_line where ((extract(year from claim_date)=2014)));
dbms_output.put_line('The highest percentage of patient is ' || PERCENTAGE || '%' || ' in the year 2014');
```

```
select (n/c)*100 into PERCENTAGE
from ( select sum(amount) as c from
(
select count(cust_id) amount from customer union all select count(d_id) amount from dependent
))
, ( select count(user_id) n from claim_line where ((extract(year from claim_date)=2013)));
dbms_output.put_line('The highest percentage of patient is ' || PERCENTAGE || '%' || ' in the year 2013');

END;

create or replace PROCEDURE yearly_statistics(top_k in integer)
IS
user_id integer;
BEGIN

/* Calling the procedures */

    cal_medserv_details_14_1 (top_k);
    cal_medserv_details_14_2 (top_k);
    search_medserv_details_14_3 (top_k);

End;

-------------------Feature 15------------------------------------
create or replace procedure findFraudPolicies_15(threshold in number,thresholdAverage in number,
typeOfUser in varchar)
is
 TYPE fraudTableType IS RECORD (
    pid        number(10),
    nextpid number(10),
    claimYear       number(10),
    byinsurance number(10),
    difference number(10)
  );
   TYPE fraudTableType_tab IS TABLE OF fraudTableType;
   fraudTableType_rec fraudTableType_tab;

    TYPE fraudTableType_sp IS RECORD (
    spid        number(10),
    nextspid number(10),
    claimYear       number(10),
    byinsurance number(10),
    difference number(10)
  );
```

```
    TYPE fraudTableType_tab_sp IS TABLE OF fraudTableType_sp;
   fraudTableType_rec_sp fraudTableType_tab_sp;
username varchar2(50);
begin
if(typeOfUser = 'Customer') then

select policy_id, lead(POLICY_ID,1) over (order by policy_id,extract(year from service_date)) as nextpid
,extract(year from service_date) as year,
sum(amount_paid_byInsurance) as totalpaidbyinsurance,
lead(sum(amount_paid_byInsurance),1) over (order by policy_id,extract(year from service_date))  -
sum(amount_paid_byInsurance)  as difference
BULK COLLECT INTO fraudTableType_rec
from claim_line where status = 'Accept'
group by extract(year from service_date),policy_id
order by policy_id,extract(year from service_date);

for i in 1..fraudTableType_rec.count
loop
if((fraudTableType_rec(i).pid = fraudTableType_rec(i).nextpid) AND fraudTableType_rec(i).difference
> threshold ) then
select cname into username from (
select cust_name as cname  from customer c
join
policy p
on p.user_id = c.USERID where p.policy_id = fraudTableType_rec(i).pid
union
select d_name as cname from dependent d
join
policy p
on
p.user_id = d.USERID where p.policy_id = fraudTableType_rec(i).pid);

     dbms_output.put_line('Condition 1:Customer ');
    dbms_output.put_line( 'Policy id: '||fraudTableType_rec(i).pid||', Username is: '||username||', Year:
'||fraudTableType_rec(i).claimYear||', Difference in insurance amount:
'||fraudTableType_rec(i).difference);
    dbms_output.put_line('------------------------------------------------------------ ');
    end if;
  END LOOP;

select  policy_id,lead(POLICY_ID,1) over (order by policy_id,extract(year from service_date)) as
nextpid,extract(year from service_date) as year,
sum(amount_paid_byInsurance)/count(policy_id) as totalAveragepaidbyInsurance,
lead(sum(amount_paid_byInsurance)/count(policy_id),1) over (order by policy_id,extract(year from
service_date))  - (sum(amount_paid_byInsurance))/count(policy_id) as difference
BULK COLLECT INTO fraudTableType_rec
from claim_line where status = 'Accept'
group by extract(year from service_date), policy_id
order by policy_id,extract(year from service_date);

for i in 1..fraudTableType_rec.count
loop
if((fraudTableType_rec(i).pid = fraudTableType_rec(i).nextpid) AND fraudTableType_rec(i).difference
> thresholdAverage ) then
select cname into username from (
select cust_name as cname  from customer c
join
policy p
on p.user_id = c.USERID where p.policy_id = fraudTableType_rec(i).pid
union
select d_name as cname from dependent d
join
policy p
on
p.user_id = d.USERID where p.policy_id = fraudTableType_rec(i).pid);

     dbms_output.put_line('Condition 2:Customer ');
    dbms_output.put_line( 'Policy id: '||fraudTableType_rec(i).pid||', Username is: '||username||', Year:
'||fraudTableType_rec(i).claimYear||', Difference in insurance amount:
'||fraudTableType_rec(i).difference);
    dbms_output.put_line('---------------------------------------------------------- ');
    end if;
  END LOOP;
End if;
if(typeOfUser = 'Service Provider') then



select sp_id,lead(sp_id,1) over (order by sp_id,extract(year from service_date)) as nextspid, extract(year
from service_date) as year,
sum(amount_paid_byInsurance) as totalpaidbyInsurance,
lead(sum(amount_paid_byInsurance),1) over (order by sp_id,extract(year from service_date))  -
(sum(amount_paid_byInsurance)) as difference
BULK COLLECT INTO fraudTableType_rec_sp
from claim_line where status = 'Accept'
group by extract(year from service_date),sp_id
order by sp_id,extract(year from service_date);
```

```
for i in 1..fraudTableType_rec_sp.count
loop
if((fraudTableType_rec_sp(i).spid = fraudTableType_rec_sp(i).nextspid) AND
fraudTableType_rec_sp(i).difference >  threshold ) then
select distinct(sp_description) into username from
service_provider sp
join
claim_line cl
on
sp.sp_id = cl.sp_id  where cl.sp_id = fraudTableType_rec_sp(i).spid;



     dbms_output.put_line('Condition 1:Service Provider ');
    dbms_output.put_line( 'Service Provider id: '||fraudTableType_rec_sp(i).spid||', Service Provider Name:
'||username||', Year: '||fraudTableType_rec_sp(i).claimYear||', Difference in insurance amount:
'||fraudTableType_rec_sp(i).difference);
     dbms_output.put_line('------------------------------------------------------------ ');
    end if;
  END LOOP;


select  sp_id,lead(sp_id,1) over (order by sp_id,extract(year from service_date)) as nextspID,extract(year
from service_date) as claimYear,
sum(amount_paid_byInsurance)/count(sp_id) as averagepaidbyInsuranceAmount,
lead(sum(amount_paid_byInsurance)/(count(sp_id)),1) over (order by sp_id,extract(year from
service_date))  - (sum(amount_paid_byInsurance))/count(sp_id) as difference
BULK COLLECT INTO fraudTableType_rec_sp
from claim_line where status = 'Accept'
group by extract(year from service_date), sp_id
order by sp_id,extract(year from service_date);

for i in 1..fraudTableType_rec_sp.count
loop
if((fraudTableType_rec_sp(i).spid = fraudTableType_rec_sp(i).nextspid) AND
fraudTableType_rec_sp(i).difference >  thresholdAverage ) then
select distinct(sp_description) into username from
service_provider sp
join
claim_line cl
on
sp.sp_id = cl.sp_id  where cl.sp_id = fraudTableType_rec_sp(i).spid;
```

```
    dbms_output.put_line('Condition 2:Service Provider ');
    dbms_output.put_line( 'Service Provider id: '||fraudTableType_rec_sp(i).spid||', Service Provider Name: 
'||username||', Year: '||fraudTableType_rec_sp(i).claimYear||', Difference in insurance amount: 
'||fraudTableType_rec_sp(i).difference);
     dbms_output.put_line('------------------------------------------------------------ ');
    end if;
  END LOOP;

end if;

end;
```

7. Executable Statements

```
-------Feature 1-----------------------------------

-- New Customer Entry

set SERVEROUTPUT ON;
exec registration ('Customer', 'Jeel', 'USA', '444-789-7495', 'jeel@gmail.com', 'jeel123', date
'1991-01-02','Male',null,null,null,null);
select * from userlogin
select * from customer;

-- Duplicate Customrer Entry
set SERVEROUTPUT ON;
exec registration ('Customer', 'Jeel', 'USA', '444-789-7495', 'jeel@gmail.com', 'jeel123', date
'1991-01-02','Male',null,null,null,null);
select * from userlogin
select * from customer;


--New Dependent Entry
set SERVEROUTPUT ON;
exec registration ('Dependent', 'Lissa', 'USA', '444-789-7495', 'Lissa@gmail.com', 'Lissa123',
date '1991-01-02','Female',16,'Sister',null,null);
select * from userlogin
select * from dependent;
```

```
--Duplicate Dependent Entry
set SERVEROUTPUT ON;
exec registration ('Dependent', 'Lissa', 'USA', '444-789-7495', 'Lissa@gmail.com', 'Lissa123',
date '1991-01-02','Female',16,'Sister',null,null);
select * from userlogin
select * from dependent;


-- New Service Provider
set SERVEROUTPUT ON;
exec registration ('Service Provider', 'Red cross', 'USA', '444-789-7495', 'redcross@gmail.com',
'redc123', date '1991-01-02',null,null,null,'Out-network',3);
select * from userlogin
select * from SERVICE_PROVIDER;


-- Duplicate Service Provider
set SERVEROUTPUT ON;
exec registration ('Service Provider', 'Red cross', 'USA', '444-789-7495', 'redcross@gmail.com',
'redc123', date '1991-01-02',null,null,null,'Out-network',3);
select * from userlogin
select * from SERVICE_PROVIDER;

/* Feature:2 */
/*--------------*/

/* Allows user to login */

set serveroutput on;
exec login('bspa@gmail.com','bspa@45');

/* Incorrect Password */

set serveroutput on;
exec login('bspa@gmail.com','b@45');

/* Incorrect email id*/

set serveroutput on;
exec login('bspa@gmail.m','bspa@45');

/* Feature:3 */
/*--------------*/
```

```
/* Allows user to read the message */
/*-------------------------------- */

set serveroutput on;
exec read_message (4, date '2016-01-01');

/* Feature: 4 */
/*-------------- */

/*Inserting into the policy table for customer*/

set serveroutput on;
exec policy_number (60,'Family First', date '2013-01-01');

/* Inserting into the policy table for dependent */
/*Insert into the policy with the user id of the related customer. Dependent 37 is related to
customer 36.*/

set serveroutput on;
exec policy_number (37,'Family First', date '2013-01-01');

/* Check for existing policy */

set serveroutput on;
exec policy_number (60,'Family First', date '2013-01-01');

--------------------Feature 5-------------------------------------

--Case 1: Adding a new dependent to the policy
set serveroutput on;
exec add_policydependent(4,1);

--Case 2: Trying to add an existing dependent to the policy
set serveroutput on;
exec add_policydependent(4,1);

-----------------------Feature 6---------------------------------
--Case 3: Removing a dependent from the table whose entry exists in the table
set serveroutput on;
exec remove_policydependent('Britny Spassky',1);

--Case 4: Removing a dependent from the table whose entry does not exist in the table
set serveroutput on;
exec remove_policydependent('Britny Spassky',1);
```

```
/* Feature: 7 */
/*-------------- */

/* Calculating Premium for the given policy id */

set serveroutput on;
exec cal_premium(5);

/* Checking if the policy exists or not */

set serveroutput on;
exec cal_premium(50);

----------------------Feature 8---------------------------------
--Case1: When the sub string of service description exists with given policy id.

set serveroutput on;
exec policy_coverage_details_8(2,'%e%');


--Case2: When the sub string of service description does not exists with given policy id.

set serveroutput on;
exec policy_coverage_details_8(3,'%e%');

----------------------Feature 9---------------------------------
--Case1: When service provider doesnt exist

set serveroutput on;
exec feature9_final_va(sparraylisttype(sparray(3,225)),40,7,'Smit',date'2006-04-30');

--Case2: When policy doesnt exist

set serveroutput on;
exec feature9_final_va(sparraylisttype(sparray(3,225)),4,70,'Smit',date'2006-04-30');

--Case3: When date range is outside acceptable plan dates

set serveroutput on;
exec feature9_final_va(sparraylisttype(sparray(3,225)),4,7,'Smit',date'2016-04-30');

select * from message;
```

```
--Case4: To check whether a provider and policy exists and submit a new claim with existing
customer for policy owner

set serveroutput on;
exec feature9_final_va(sparraylisttype(sparray(3,225)),4,7,'Smit',date'2006-04-30');

--Case5: To check whether a provider and policy exists and submit a duplicate claim with same
service id, policy id, service date for policy owner

set serveroutput on;
exec feature9_final_va(sparraylisttype(sparray(3,225)),4,7,'Smit',date'2006-04-30');

--Case6: When policy deductable is not reached and a new bill is submitted

set serveroutput on;
exec feature9_final_va(sparraylisttype(sparray(3,225)),4,7,'Smit',date'2006-05-01');

--Case7: Another claim submission to reach maximum allowed service

set serveroutput on;
exec feature9_final_va(sparraylisttype(sparray(3,225)),4,7,'Smit',date'2006-05-02');

--Case8: When submitting more claims than allowed per year, message should be displayed
that more claims than allowed and enter a declined claim into claim_line table.

set serveroutput on;
exec feature9_final_va(sparraylisttype(sparray(3,225)),4,7,'Smit',date'2006-05-03');

--Case9: When dependent not linked to policy

exec feature9_final_va(sparraylisttype(sparray(3,225)),4,7,'Boris Spassky',date'2006-05-01');

--Case10: When date of service is outside acceptable plan date for dependent

set serveroutput on;
exec feature9_final_va(sparraylisttype(sparray(2,225)),2,2,'Boris Spassky',date'2016-05-09');

--Case11: Submit a new claim with existing customer for policy dependent

set serveroutput on;
exec feature9_final_va(sparraylisttype(sparray(2,300)),2,2,'Boris Spassky',date'2002-05-16');

--Case12: Submit a duplicate claim with same service id, policy id, service date for policy
dependent
```

```
set serveroutput on;
exec feature9_final_va(sparraylisttype(sparray(2,300)),2,2,'Boris Spassky',date'2002-05-16');


/* Feature: 10 */
/*-------------- */

/* Allows user to search for claims */

set serveroutput on;
exec search_ClaimDetails (4, date '2016-01-01',date '2017-06-26');

/* No Claims for the given Customer */
set serveroutput on;
exec search_ClaimDetails (12, date '2017-01-01',date '2017-06-26');

----------------------Feature 11 -----------------------------------
-- Entry is present
set serveroutput on;
exec claimDetails(2);

-- No claim Present
set serveroutput on;
exec claimDetails(2);


/* Feature: 12 */
/*-------------- */

/* Calculating totals for the customer */

set serveroutput on;
exec check_totalCost_1(12,2015);

/* No details for given customer */

set serveroutput on;
exec check_totalCost_1(12,2016);


---------------------Feature 13---------------------------------
--Year from 2001 to 2016--Statistic Result
```

```
set serveroutput on;
exec displayResult_feature13 (2);



/* Feature: 14 */
/*-------------- */

/* compute the yearly usage statistics for the past 5 years */

set serveroutput on;
exec yearly_statistics(2);



-----------------------Feature 15-------------------------------
set serveroutput on;
exec findFraudPolicies_15(900,400,'Customer');
exec findFraudPolicies_15(900,400,'Service Provider');
```