Username:-ADMIN

Password:-Ajayanuhyatarun@2022

CREATE TABLE Insurance\_Company(

company\_id INT NOT NULL PRIMARY KEY,

employer\_id INT UNIQUE NOT NULL,

company\_name varchar(255) NOT NULL,

phone\_number varchar(10) NOT NULL,

street varchar(255) NOT NULL,

city varchar(255) NOT NULL,

zipcode int NOT NULL,

state varchar(255) NOT NULL,

password varchar(255) NOT NULL

);

CREATE TABLE Agent(

agent\_id INT NOT NULL PRIMARY KEY,

company\_id INT not NULL,

first\_name VARCHAR(255) not null,

last\_name Varchar(255) not null,

phone Varchar(10) NOT NULL,

email varchar(255) not null,

password varchar(255) not null

);

ALTER TABLE Agent

ADD FOREIGN KEY (company\_id) REFERENCES Insurance\_Company(company\_id);

CREATE TABLE corporate\_clients(

employer\_id INT NOT NULL PRIMARY KEY,

company\_id INT not NULL,

employer\_name varchar(255) not null,

total\_employees int not null

);

ALTER TABLE corporate\_clients

ADD FOREIGN KEY (company\_id) REFERENCES Insurance\_Company(company\_id);

create table customer(

customer\_id int not null primary key,

first\_name VARCHAR(255) not null,

last\_name Varchar(255) not null,

ssn Varchar(10) NOT NULL,

email varchar(255) not null,

address varchar(255) not null,

zipcode int not null,

state varchar(255) not null,

password varchar(255) not null

);

create table customer\_agent\_relation(

customer\_id int not null,

agent\_id int not null,

PRIMARY KEY(customer\_id,agent\_id)

);

ALTER TABLE customer\_agent\_relation

ADD FOREIGN KEY (customer\_id) REFERENCES Customer(customer\_id)

ADD FOREIGN KEY (agent\_id) REFERENCES Agent(agent\_id);

create table corporate\_customer\_relation(

customer\_id int not null,

employer\_id int not null,

PRIMARY KEY(customer\_id,employer\_id)

);

ALTER TABLE corporate\_customer\_relation

ADD FOREIGN KEY (customer\_id) REFERENCES Customer(customer\_id)

ADD FOREIGN KEY (employer\_id) REFERENCES corporate\_clients(employer\_id);

create table policy(

policy\_number int not null primary key,

company\_id int not null,

customer\_id int not null,

policy\_type varchar(255) not null,

premium int not null,

tenure int not null

);

ALTER TABLE policy

ADD FOREIGN KEY (company\_id) REFERENCES Insurance\_Company(company\_id)

ADD FOREIGN KEY (customer\_id) REFERENCES customer(customer\_id);

create table claim(

claim\_id int not null primary key,

policy\_number int not null,

bill\_id int not null,

claim\_amount varchar(255) not null,

date\_of\_claim date not null

);

ALTER TABLE claim

ADD FOREIGN KEY (policy\_number) REFERENCES policy(policy\_number)

ADD FOREIGN KEY (bill\_id) REFERENCES bill(bill\_id);

create table hospital(

hospital\_id int not null primary key,

hospital\_name varchar(255) not null,

street varchar(255) not null,

city varchar(255) not null,

state varchar(255) not null,

zipcode int not null

);

create table bill(

bill\_id int not null primary key,

hospital\_id int not null ,

treatment\_id int not null,

patient\_id int not null,

date\_of\_bill date not null,

amount int not null

);

ALTER TABLE bill

ADD FOREIGN KEY (hospital\_id) REFERENCES hospital(hospital\_id)

ADD FOREIGN KEY (treatment\_id) REFERENCES treatment(treatment\_id);

create table treatment(

treatment\_id int not null primary key,

patient\_id int not null,

prescription\_id int not null,

doctor\_id int not null,

treatment\_type varchar(255) not null

);

ALTER TABLE treatment

ADD FOREIGN KEY (patient\_id) REFERENCES patient(patient\_id)

ADD FOREIGN KEY (prescription\_id) REFERENCES prescription(prescription\_id)

ADD FOREIGN KEY (doctor\_id) REFERENCES doctor(doctor\_id);

create table doctor(

doctor\_id int not null primary key ,

hospital\_id int not null,

first\_name varchar(255) not null,

last\_name varchar(255) not null,

department varchar(255) not null

);

ALTER TABLE doctor

ADD FOREIGN KEY (hospital\_id) REFERENCES hospital(hospital\_id);

create table patient(

patient\_id int not null primary key,

customer\_id int not null ,

first\_name varchar(255) not null,

last\_name varchar(255) not null,

age int not null,

sex varchar(255) not null,

relationship varchar(255) not null

);

ALTER TABLE patient

ADD FOREIGN KEY (customer\_id) REFERENCES customer(customer\_id);

create table prescription(

prescription\_id int not null primary key,

doctor\_id int not null ,

patient\_id int not null ,

quantity int not null ,

frequncy int not null,

drug\_name varchar(255) not null

);

ALTER TABLE prescription

ADD FOREIGN KEY (patient\_id) REFERENCES patient(patient\_id)

ADD FOREIGN KEY (doctor\_id) REFERENCES doctor(doctor\_id);

create table pharmacy\_billing(

pharmacy\_id int not null,

prescription\_id int not null,

primary key(pharmacy\_id,prescription\_id)

);

ALTER TABLE pharmacy\_billing

ADD FOREIGN KEY (prescription\_id) REFERENCES prescription(prescription\_id)

ADD FOREIGN KEY (pharmacy\_id) REFERENCES pharmacy(pharmacy\_id);

create table pharmacy(

pharmacy\_id int not null primary key,

name varchar(100) not null,

street varchar(255) not null,

city varchar(255) not null,

state varchar(255) not null,

zipcode int not null

);

create or replace PACKAGE BODY CUSTOMER\_ACTIONS AS

procedure insert\_customer(IN\_custid number,

IN\_fname VARCHAR,

IN\_lname VARCHAR,

IN\_ssn VARCHAR,

IN\_email VARCHAR,

IN\_address VARCHAR,

IN\_zipcode number,

IN\_state VARCHAR,

IN\_password VARCHAR

)

is

error\_custid exception;

error\_fname exception;

error\_lname exception;

error\_ssn exception;

error\_email exception;

error\_address exception;

error\_zipcode exception;

error\_state exception;

error\_password exception;

count\_cid number;

begin

SELECT count(\*) into count\_cid FROM Customer WHERE customer\_id = IN\_custid;

if(IN\_custid IS NULL or count\_cid !=0)

then

raise error\_custid;

elsif(IN\_fname IS NULL OR IN\_fname = '')

then

raise error\_fname;

elsif(IN\_lname IS NULL OR IN\_lname = '')

then

raise error\_lname;

elsif(IN\_ssn IS NULL OR IN\_ssn = '')

then

raise error\_ssn;

elsif(IN\_email IS NULL OR IN\_email = '')

then

raise error\_email;

elsif(IN\_address IS NULL OR IN\_address = '')

then

raise error\_address;

elsif(IN\_zipcode IS NULL)

then

raise error\_zipcode;

elsif(IN\_state IS NULL OR IN\_state = '')

then

raise error\_state;

elsif(IN\_password IS NULL OR IN\_password = '')

then

raise error\_password;

else

insert into customer(customer\_id,first\_name,last\_name,ssn,email,address,zipcode,state,password)VALUES(

IN\_custid,

IN\_fname,

IN\_lname,

IN\_ssn,

IN\_email,

IN\_address,

IN\_zipcode,

IN\_state,

IN\_password

);

end if;

commit;

dbms\_output.put\_line('You have sucessfully inserted customer');

-- exception

-- when others

-- then dbms\_output.put\_line(sqlerrm);

end insert\_customer;

procedure create\_patient(IN\_patientid number, IN\_customerid number ,IN\_fname varchar,IN\_lname varchar,IN\_age number,

IN\_sex varchar,IN\_relationship varchar)

IS

count\_pid number;

count\_cid number;

error\_customerid exception;

error\_patientid EXCEPTION;

error\_fname EXCEPTION;

error\_lname EXCEPTION;

error\_age exception;

error\_sex exception;

error\_relationship exception;

begin

SELECT count(\*) into count\_pid FROM patient

WHERE patient\_id = IN\_patientid;

Select count(\*) into count\_cid FROM customer

where customer\_id = IN\_customerid;

if(IN\_customerid IS NULL or count\_cid != 1)

then

raise error\_customerid;

elsif(IN\_patientid IS NULL or count\_pid != 0)

then

raise error\_patientid;

elsif(IN\_fname IS NULL OR IN\_fname = '')

then

raise error\_fname;

elsif(IN\_lname IS NULL OR IN\_lname = '')

then

raise error\_lname;

elsif(IN\_age IS NULL)

then

raise error\_age;

elsif(IN\_sex IS NULL OR IN\_sex = '')

then

raise error\_sex;

elsif(IN\_relationship IS NULL OR IN\_relationship = '')

then

raise error\_relationship;

else

insert into patient(patient\_id,customer\_id,first\_name ,last\_name,age,sex,relationship)VALUES(

IN\_patientid , IN\_customerid ,IN\_fname,IN\_lname,IN\_age ,IN\_sex ,IN\_relationship);

end if;

commit;

dbms\_output.put\_line('You have sucessfully inserted patient');

end create\_patient;

procedure delete\_patient(IN\_pateintid number,IN\_customerid number)

IS

cust\_id number;

begin

delete from patient where patient.customer\_id=IN\_customerid and patient.customer\_id=in\_pateintid;

commit;

dbms\_output.put\_line('delete of patient successful');

end delete\_patient;

END CUSTOMER\_ACTIONS;

create or replace PACKAGE BODY AGENT\_ACTION AS

procedure insert\_customer(IN\_custid number,

IN\_fname VARCHAR,

IN\_lname VARCHAR,

IN\_ssn VARCHAR,

IN\_email VARCHAR,

IN\_address VARCHAR,

IN\_zipcode number,

IN\_state VARCHAR,

IN\_password VARCHAR,

IN\_agentid number

)

is

error\_custid exception;

error\_fname exception;

error\_lname exception;

error\_ssn exception;

error\_email exception;

error\_address exception;

error\_zipcode exception;

error\_state exception;

error\_password exception;

count\_cid number;

begin

SELECT count(\*) into count\_cid FROM Customer WHERE customer\_id = IN\_custid;

if(IN\_custid IS NULL or count\_cid !=0)

then

raise error\_custid;

elsif(IN\_fname IS NULL OR IN\_fname = '')

then

raise error\_fname;

elsif(IN\_lname IS NULL OR IN\_lname = '')

then

raise error\_lname;

elsif(IN\_ssn IS NULL OR IN\_ssn = '')

then

raise error\_ssn;

elsif(IN\_email IS NULL OR IN\_email = '')

then

raise error\_email;

elsif(IN\_address IS NULL OR IN\_address = '')

then

raise error\_address;

elsif(IN\_zipcode IS NULL)

then

raise error\_zipcode;

elsif(IN\_state IS NULL OR IN\_state = '')

then

raise error\_state;

elsif(IN\_password IS NULL OR IN\_password = '')

then

raise error\_password;

else

insert into customer(customer\_id,first\_name,last\_name,ssn,email,address,zipcode,state,password)VALUES(

IN\_custid,

IN\_fname,

IN\_lname,

IN\_ssn,

IN\_email,

IN\_address,

IN\_zipcode,

IN\_state,

IN\_password

);

insert into customer\_agent\_relation(customer\_id,agent\_id)values(IN\_custid,IN\_agentid);

end if;

commit;

dbms\_output.put\_line('You have sucessfully inserted customer');

-- exception

-- when others

-- then dbms\_output.put\_line(sqlerrm);

end insert\_customer;

procedure delete\_customer\_by\_agent(IN\_agentid number,IN\_customerid number)

IS

cust\_id number;

agent\_id number;

begin

delete from customer\_agent\_relation where customer\_agent\_relation.customer\_id=IN\_customerid and customer\_agent\_relation.agent\_id=in\_agentid;

DELETE FROM customer WHERE customer.customer\_id=IN\_customerid;

commit;

dbms\_output.put\_line('delete of customer of agent sucessfull');

end delete\_customer\_by\_agent;

END AGENT\_ACTION;

create or replace PACKAGE BODY CORPORATE\_CLIENTS\_ACTIONS AS

procedure insert\_customer\_corporate(IN\_custid number,

IN\_fname VARCHAR,

IN\_lname VARCHAR,

IN\_ssn VARCHAR,

IN\_email VARCHAR,

IN\_address VARCHAR,

IN\_zipcode number,

IN\_state VARCHAR,

IN\_password VARCHAR,

IN\_employerid number

)

is

error\_custid exception;

error\_fname exception;

error\_lname exception;

error\_ssn exception;

error\_email exception;

error\_address exception;

error\_zipcode exception;

error\_state exception;

error\_password exception;

count\_cid number;

begin

SELECT count(\*) into count\_cid FROM Customer WHERE customer\_id = IN\_custid;

if(IN\_custid IS NULL or count\_cid !=0)

then

raise error\_custid;

elsif(IN\_fname IS NULL OR IN\_fname = '')

then

raise error\_fname;

elsif(IN\_lname IS NULL OR IN\_lname = '')

then

raise error\_lname;

elsif(IN\_ssn IS NULL OR IN\_ssn = '')

then

raise error\_ssn;

elsif(IN\_email IS NULL OR IN\_email = '')

then

raise error\_email;

elsif(IN\_address IS NULL OR IN\_address = '')

then

raise error\_address;

elsif(IN\_zipcode IS NULL)

then

raise error\_zipcode;

elsif(IN\_state IS NULL OR IN\_state = '')

then

raise error\_state;

elsif(IN\_password IS NULL OR IN\_password = '')

then

raise error\_password;

else

insert into customer(customer\_id,first\_name,last\_name,ssn,email,address,zipcode,state,password)VALUES(

IN\_custid,

IN\_fname,

IN\_lname,

IN\_ssn,

IN\_email,

IN\_address,

IN\_zipcode,

IN\_state,

IN\_password

);

insert into corporate\_customer\_relation(customer\_id,employer\_id)values(IN\_custid,IN\_employerid);

end if;

commit;

dbms\_output.put\_line('You have sucessfully inserted customer');

end insert\_customer\_corporate;

procedure delete\_customer\_by\_corporate(IN\_employerid number,IN\_customerid number)

IS

cust\_id number;

begin

delete from corporate\_customer\_relation where corporate\_customer\_relation.customer\_id=IN\_customerid and corporate\_customer\_relation.employer\_id=in\_employerid;

DELETE FROM customer WHERE customer.customer\_id=IN\_customerid;

commit;

dbms\_output.put\_line('delete of customer of corporate sucessfull');

end delete\_customer\_by\_corporate;

END CORPORATE\_CLIENTS\_ACTIONS;