```sql
---drop tables
drop table Review;
drop table payout;
drop table booking_information;
drop table Listing_Availability_Period;
drop table Listing;
drop table List_of_amenities;
drop table Messages;
drop table host;
drop table guest;
drop table System_User;

-- Drop Sequences
DROP sequence SYSTEM_USER_SEQ;
DROP sequence MESSAGES_SEQ;
drop sequence LISTING_SEQ;
drop sequence LISTING_AVAIL_PERIOD_SEQ;
drop sequence BOOKING_SEQ;
drop sequence PAYOUT_SEQ;

--Create Sequences
Create sequence SYSTEM_USER_SEQ start with 1 increment by 1 minvalue 1;
Create sequence MESSAGES_SEQ start with 1 increment by 1 minvalue 1;
Create sequence LISTING_SEQ start with 1 increment by 1 minvalue 1;
Create sequence LISTING_AVAIL_PERIOD_SEQ start with 1 increment by 1 minvalue 1;
Create sequence BOOKING_SEQ start with 1 increment by 1 minvalue 1;
Create sequence PAYOUT_SEQ start with 1 increment by 1 minvalue 1;

--Create Tables
-- Create system user table

create table System_User(
User_ID int not null,
Name varchar(50),
Phone_no varchar(50),
Email varchar(50),
Password varchar(50),
Mailing_address varchar(50),
user_type varchar(50),
primary key (User_ID)
);

-- create guest
create table Guest(
Guest_ID int not null,
Average_rating float,
Profile varchar(250),
primary key (Guest_ID),
foreign key (Guest_ID) references System_User
);

--Create Host
create table Host(
Host_ID int not null,
Average_rating float,
Payment_method varchar(15),
primary key (Host_ID),
foreign key (Host_ID) references System_User
);

--Create Table Messages
 create table Messages(
Message_ID int not null,
User_ID int not null,
Message varchar(500),
Message_date date,
primary key (Message_ID,User_ID),
foreign key (User_ID) references System_User
);
```

**--Create Table List_of_amenities**

```
create table List_of_amenities(
List_of_amenities_ID int not null,
Microwave int,
TV int,
Wifi int,
Washer_and_dryer int,
Free_parking int,
primary key (List_of_amenities_ID)
);
```

**--Create Table Listing**

```
create table Listing(
Listing_ID int not null,
Host_ID int  not null,
House_no int not null,
Street varchar(25),
City varchar(10),
State varchar(5),
Zipcode int,
Type varchar(15),
Maximal_Capacity int,
No_of_Bedrooms int,
No_of_Beds int,
No_of_Bathrooms int,
Min_No_of_Nights_To_Stay int,
Check_In_Time interval day(0) to second(0),
Check_Out_Time interval day(0) to second(0),
```

List_of_amenities int,

primary key (Listing_ID),

foreign key (Host_ID) references System_User,

foreign key (List_of_amenities) references List_of_amenities

);

**--Create Table Listing_Availability_Period**

create table Listing_Availability_Period(

LISTING_AVAILABILITY_PERIOD_ID int not null,

Listing_ID int not null,

Start_Date date,

End_Date date,

Price_per_night number,

primary key (LISTING_AVAILABILITY_PERIOD_ID),

foreign key (Listing_ID) references Listing);

**--Create Booking information**

create table Booking_information(

Booking_ID int,

Guest_ID int,

check_in_date date,

check_out_date date,

No_of_adults int,

No_of_children int,

Booking_status varchar(15),

Payout_status int,

Listing_ID int,

Total_cost int,

primary key (Booking_ID),

foreign key (Guest_ID) references Guest,

foreign key (Listing_ID) references Listing

);

**--Create Payout**

create table Payout

```
(Payout_ID int,

Host_ID int,

Payout_amount int,

Payout_date date,

primary key (Payout_ID),

foreign key (Host_ID) references Host

);

--Create Review

create table Review(

Host_ID int  not null,

Guest_ID int  not null,

Review varchar(50),

Stars int,

Flag int not null,  -- 1 is for guest to host and 2 for host to guest

primary key (Flag,Host_ID,Guest_ID),

foreign key (Guest_ID) references Guest,

foreign key (Host_ID) references Host

);


--Insert statements

-- Insert into system_user

insert into system_user values (system_user_seq.nextval, 'Sid', '443-251-8772','sid@umbc.edu','sid123','4751,Drayton Grn, Arbutus,MD,21227','Guest');

insert into system_user values (system_user_seq.nextval, 'Jas', '443-251-8773','jas@umbc.edu','jas123','4770,Aldgate Grn, Arbutus,MD,21227','Host');

insert into system_user values (system_user_seq.nextval, 'Tim', '443-251-1234','tim@gmail.com','tim123','4752,Greenville, Arbutus,MD,21227','Guest');

insert into system_user values (system_user_seq.nextval, 'Eve', '443-251-5678','eve@yahoo.com','eve123','4773,Gateway Terrace, Arbutus,MD,21227','Host');

insert into system_user values (system_user_seq.nextval, 'Jack', '443-251-3423','jack@gmail.com','jack123','4753,Maiden choice, Arbutus,MD,21227','Guest');

insert into system_user values (system_user_seq.nextval, 'Jared', '443-251-4346','jared@yahoo.com','jared123','4778,Gateway Terrace, Arbutus,MD,21227','Host');

insert into system_user values (system_user_seq.nextval, 'Daniel', '449-431-1245','daniel@gmail.com','danielbb','546,Charles Street,Baltimore Downtown,MD,21887','Both');
```

insert into system_user values (system_user_seq.nextval, 'Brad', '556-233-6754','brad@yahoo.com','brad@43%','654,Broadway, New York,4356','Both');

insert into system_user values (system_user_seq.nextval, 'Sam', '443-251-8872','sam@umbc.edu','sam123','2175,Marton Grn, Catonsville,MD,21220','Guest');

insert into system_user values (system_user_seq.nextval, 'David', '411-332-8273','david@umbc.edu','david123','4770,Ellicott , Halethrope,MD,21321','Host');

insert into system_user values (system_user_seq.nextval, 'Krish', '310-110-8234','krish@gmail.com','krish123','4752,Parkville,Glen Burnie,MD,21977','Guest');

insert into system_user values (system_user_seq.nextval, 'James', '223-201-7678','james@yahoo.com','james123','2373,Hamilton Terrace, Ricerstown,MD,20011','Host');

insert into system_user values (system_user_seq.nextval, 'Shawn', '212-443-3013','shawn@gmail.com','shawn123','8753,Georgetown, Catonsville,MD,21112','Guest');

insert into system_user values (system_user_seq.nextval, 'Jennifer', '343-251-8872','jen@umbc.edu','jen123','4175,Boston Street, Catonsville,MD,21220','Guest');

insert into system_user values (system_user_seq.nextval, 'Mary', '201-302-2873','mary@umbc.edu','mary123','4770,Riverville, Gathesburg,MD,21321','Host');

insert into system_user values (system_user_seq.nextval, 'Sonia', '889-190-8324','son@gmail.com','son123','4952,Calvert Street,Glen Burnie,MD,21877','Host');

insert into system_user values (system_user_seq.nextval, 'Yammy', '505-221-6678','yam@yahoo.com','yam123','3773,Hanover, Silver spring,MD,21911','Host');

insert into system_user values (system_user_seq.nextval, 'Jones', '323-493-0313','jones@gmail.com','jones123','4953,Coca-cola Drive, Baltimore,MD,21012','Guest');

insert into system_user values (system_user_seq.nextval, 'Jammy', '211-801-6778','jammy@yahoo.com','jammy123','4793,Pratt Street, Downtown,MD,21911','Host');

insert into system_user values (system_user_seq.nextval, 'Max', '110-243-3223','max@gmail.com','max123','3753,Redwood street, Satellite,MD,21012','Host');

**-- Insert into Guest**

insert into Guest values (1, 3, 'Gender:Male, Family:4, Job:IT professional, Hobby:Reading');

insert into Guest values (3, 4, 'Gender:Male, Family:3, Occupation: Business, Hobby:Music');

insert into Guest values (5, 3, 'Gender:Male, Family:2, Occupation: Business Analyst, Hobby:Dancing');

insert into Guest values (7, 0, 'Gender:Male, Family:5, Occupation: Student, Hobby:Reading');

insert into Guest values (8, 0, 'Gender:Male, Family:4, Occupation: Professor, Hobby:Playing golf');

insert into Guest values (9, 0, 'Gender:Male, Family:4, Occupation: Interiior Decorator, Hobby:Playing golf');

insert into Guest values (11, 0, 'Gender:Male, Family:5, Occupation: Fashion Designer, Hobby:Collecting coins');

insert into Guest values (13, 0, 'Gender:Male, Family:6, Occupation: Doctor, Hobby:Dancing');

insert into Guest values (14, 0, 'Gender:Female, Family:3, Occupation: Cricketer, Hobby:Playing Music');

insert into Guest values (18, 0, 'Gender:Male, Family:4, Occupation: Data Analyst, Hobby:Reading');

**--Insert into Host**

insert into host values (2,3,'PNC123');

insert into host values (4,4,'BOFA987');

insert into host values (6,4,'CAPONE223');

insert into host values (7,3,'PNC232');

insert into host values (8,2,'NYB111');

insert into host values (10,3,'NYC123');

insert into host values (12,4,'WF332');

insert into host values (15,0,'AMEX9231');

insert into host values (16,0,'BOFA129881');

insert into host values (17,0,'PNC8362');

insert into host values (19,0,'PNC9823');

insert into host values (20,0,'CAPONE1827');

**--Insert into MESSAGES**

insert into Messages values (MESSAGES_SEQ.nextval,9,'Maximal capacity exceded.',date '2017-10-07');

insert into Messages values (MESSAGES_SEQ.nextval,14,'Minimum number of stay not met.',date '2017-11-08');

insert into Messages values (MESSAGES_SEQ.nextval,4,'Payout maade by company for previous month.',date '2017-12-18');

insert into Messages values (MESSAGES_SEQ.nextval,4,'Payment has been made for your listing.',date '2017-12-01');

**--Insert into List_of_amenities**

insert into List_of_amenities values(1,1,1,1,1,0);

insert into List_of_amenities values(2,0,1,0,1,0);

insert into List_of_amenities values(3,1,0,1,1,1);

insert into List_of_amenities values(4,1,1,1,1,1);

insert into List_of_amenities values(5,0,1,0,1,1);

**--Insert of Listing**

insert into LISTING values (LISTING_SEQ.nextval,2,4770,'Aldgate Grn','Arbutus','MD',21227,'Townhouse',4,2,4,1,2,to_dsinterval('0 12:00:00'),to_dsinterval('0 20:00:00'),1);

```
insert into LISTING values (LISTING_SEQ.nextval,4,4752,'Maiden
Choice','Arbutus','MD',21227,'Apartment',5,2,4,2,3,to_dsinterval('0 12:00:00'),to_dsinterval('0 14:00:00'),2);

insert into LISTING values (LISTING_SEQ.nextval,6,3232,'Circle
Drive','Arbutus','MD',21227,'House',4,2,4,1,2,to_dsinterval('0 12:00:00'),to_dsinterval('0 20:00:00'),3);

insert into LISTING values (LISTING_SEQ.nextval,6,2145,'Charles
Drive','Arbutus','MD',21345,'Apartment',4,2,4,2,2,to_dsinterval('0 12:00:00'),to_dsinterval('0 20:00:00'),4);

insert into LISTING values (LISTING_SEQ.nextval,7,1001,' East Broadway','New
York','NY',3244,'Apartment',4,2,4,2,1,to_dsinterval('0 12:00:00'),to_dsinterval('0 20:00:00'),5);

insert into LISTING values (LISTING_SEQ.nextval,8,654,'Broadway','New
York','NY',24356,'House',4,2,4,1,2,to_dsinterval('0 12:00:00'),to_dsinterval('0 20:00:00'),3);

insert into LISTING values
(LISTING_SEQ.nextval,10,4770,'Ellicott','Halethrope','MD',21321,'Apartment',4,2,4,2,2,to_dsinterval('0
12:00:00'),to_dsinterval('0 20:00:00'),4);

insert into LISTING values (LISTING_SEQ.nextval,12,2373,'Hamilton
Terrace','Ricerstown','MD',20011,'Apartment',4,2,4,2,1,to_dsinterval('0 12:00:00'),to_dsinterval('0 20:00:00'),5);

insert into LISTING values
(LISTING_SEQ.nextval,15,4770,'Riverville','Gathesburg','MD',21321,'House',4,2,4,1,2,to_dsinterval('0
12:00:00'),to_dsinterval('0 20:00:00'),3);

insert into LISTING values (LISTING_SEQ.nextval,16,4952,'Calvert
Street','Glen','MD',21877,'Apartment',4,2,4,2,2,to_dsinterval('0 12:00:00'),to_dsinterval('0 20:00:00'),4);

insert into LISTING values
(LISTING_SEQ.nextval,17,3773,'Hanover','Silver','MD',21911,'Apartment',4,2,4,2,1,to_dsinterval('0
12:00:00'),to_dsinterval('0 20:00:00'),5);
```

**--Insert into listing availability**

```
insert into LISTING_AVAILABILITY_PERIOD values (LISTING_AVAIL_PERIOD_SEQ.nextval,1, date '2017-10-
11', date '2017-10-21', 40);

insert into LISTING_AVAILABILITY_PERIOD values (LISTING_AVAIL_PERIOD_SEQ.nextval,1, date '2017-10-
21', date '2017-11-25', 50);

insert into LISTING_AVAILABILITY_PERIOD values (LISTING_AVAIL_PERIOD_SEQ.nextval,2, date '2018-03-
01', date '2018-03-20', 60);

insert into LISTING_AVAILABILITY_PERIOD values (LISTING_AVAIL_PERIOD_SEQ.nextval,2, date '2018-03-
22', date '2018-03-30', 100);

insert into LISTING_AVAILABILITY_PERIOD values (LISTING_AVAIL_PERIOD_SEQ.nextval,3, date '2018-05-
18', date '2018-05-25', 70);

insert into LISTING_AVAILABILITY_PERIOD values (LISTING_AVAIL_PERIOD_SEQ.nextval,4, date '2017-10-
11', date '2017-10-21', 200);

insert into LISTING_AVAILABILITY_PERIOD values (LISTING_AVAIL_PERIOD_SEQ.nextval,4, date '2017-10-
21', date '2017-11-25', 150);
```

insert into LISTING_AVAILABILITY_PERIOD values (LISTING_AVAIL_PERIOD_SEQ.nextval,11, date '2018-10-21', date '2018-11-25', 30);

insert into LISTING_AVAILABILITY_PERIOD values (LISTING_AVAIL_PERIOD_SEQ.nextval,11, date '2018-11-26', date '2018-12-20', 20);

insert into LISTING_AVAILABILITY_PERIOD values (LISTING_AVAIL_PERIOD_SEQ.nextval,10, date '2018-01-01', date '2018-01-30', 80);

insert into LISTING_AVAILABILITY_PERIOD values (LISTING_AVAIL_PERIOD_SEQ.nextval,8, date '2018-01-02', date '2018-01-31', 35);

insert into LISTING_AVAILABILITY_PERIOD values (LISTING_AVAIL_PERIOD_SEQ.nextval,9, date '2018-06-01', date '2018-06-30', 65);

insert into LISTING_AVAILABILITY_PERIOD values (LISTING_AVAIL_PERIOD_SEQ.nextval,9, date '2018-07-10', date '2018-07-25', 25);

**--Insert into booking information**

insert into BOOKING_INFORMATION values (BOOKING_SEQ.nextval,1, date '2018-10-22', date '2018-12-19',2,3,'Requested',0,11,450);

insert into BOOKING_INFORMATION values (BOOKING_SEQ.nextval,3, date '2018-01-02', date '2018-01-21',2,2,'Requested',0,10,600);

insert into BOOKING_INFORMATION values (BOOKING_SEQ.nextval,5, date '2018-01-05', date '2018-01-10',2,3,'Requested',0,10,400);

insert into BOOKING_INFORMATION values (BOOKING_SEQ.nextval,7, date '2018-11-27', date '2018-12-05',2,0,'Requested',0,11,500);

insert into BOOKING_INFORMATION values (BOOKING_SEQ.nextval,8, date '2018-01-04', date '2018-01-11',2,1,'Paid',0,8,80);

insert into BOOKING_INFORMATION values (BOOKING_SEQ.nextval,9, date '2018-07-11', date '2018-07-21',2,1,'Paid',0,9,80);

insert into BOOKING_INFORMATION values (BOOKING_SEQ.nextval,11, date '2018-01-12', date '2018-01-15',2,1,'Paid',1,8,80);

insert into BOOKING_INFORMATION values (BOOKING_SEQ.nextval,13, date '2018-06-20', date '2018-06-25',2,1,'Paid',1,9,80);

insert into BOOKING_INFORMATION values (BOOKING_SEQ.nextval,13, date '2018-01-12', date '2018-01-15',2,1,'Paid',1,10,400);

insert into BOOKING_INFORMATION values (BOOKING_SEQ.nextval,11, date '2018-12-06', date '2018-12-13',2,1,'Paid',1,11,500);

**--review**

insert into REVIEW values (12,8,'Very well maintained apartment',5,1);

insert into REVIEW values (15,9,'Friendly Host',4,1);

```
insert into REVIEW values (12,8,'Well Mannered guest',3.5,2);
insert into REVIEW values (15,9,'Friendly Guest',3,2);
```

**---payout**

```
insert into PAYOUT values (PAYOUT_SEQ.nextval,12,700,date '2017-03-25');
insert into PAYOUT values (PAYOUT_SEQ.nextval,15,800,date '2017-02-12');
insert into PAYOUT values (PAYOUT_SEQ.nextval,16,450,date '2017-01-02');
insert into PAYOUT values (PAYOUT_SEQ.nextval,17,280,date '2017-10-30');
```

**----select statements**

```
select * from Booking_Information;
select * from Review;
select * from Messages;
select * from Host;
select * from Listing_Availability_Period;
select * from Listing;
select * from List_of_amenities;
select * from Guest;
select * from System_User;
select * from Payout;
```

**-- FEATURE 1:  Register a user with the system**

**--FUNCTION to create CHECK_EXISTING_EMAIL**

```
create or replace function CHECK_EXISTING_EMAIL(EMAIL_ID in varchar)
return NUMBER
IS
CHECK_EMAIL VARCHAR(50);
BEGIN
select EMAIL into CHECK_EMAIL from SYSTEM_USER where EMAIL_ID = EMAIL;
return 1;
exception
when no_data_found then
return -1;
```

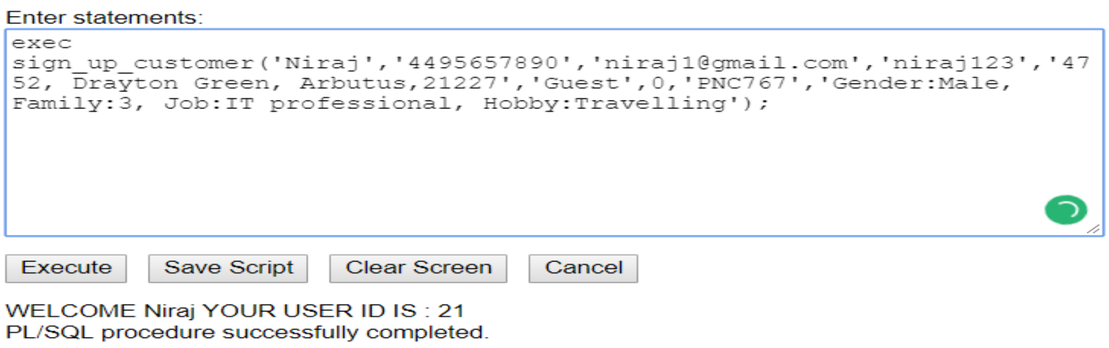
```
END;
```

**-- create procedure sign_up_customer**

```
CREATE OR REPLACE procedure sign_up_customer(name_user in varchar, phone_no_user in varchar, Email_ID in varchar,

password_user in varchar, mailing_address_user in varchar, user_type_user in varchar, average_rating_user in float, pay_method in varchar, Profile_user in varchar) IS

check_email number;

new_user_id system_user.user_id%type;

BEGIN

check_email := CHECK_EXISTING_EMAIL(Email_ID);

IF check_email = 1 THEN  dbms_output.put_line('User already exist');

ELSE

INSERT INTO system_user values(SYSTEM_USER_SEQ.nextval,name_user,phone_no_user,Email_ID,password_user,mailing_address_user,user_type_user);

select user_id into new_user_id from system_user where EMAIL = Email_ID;

        dbms_output.put_line('WELCOME '||name_user||' YOUR USER ID IS : '||new_user_id);

if user_type_user = 'Host' Then

INSERT INTO host values(SYSTEM_USER_SEQ.currval,average_rating_user,pay_method);

elsif user_type_user = 'Guest' Then

INSERT INTO guest values(SYSTEM_USER_SEQ.currval,average_rating_user,Profile_user);

elsif user_type_user = 'Both' then

INSERT INTO host values(SYSTEM_USER_SEQ.currval,average_rating_user,pay_method);

INSERT INTO guest values(SYSTEM_USER_SEQ.currval,average_rating_user,Profile_user);

end if;

END IF;

END;
```

**--Execution**

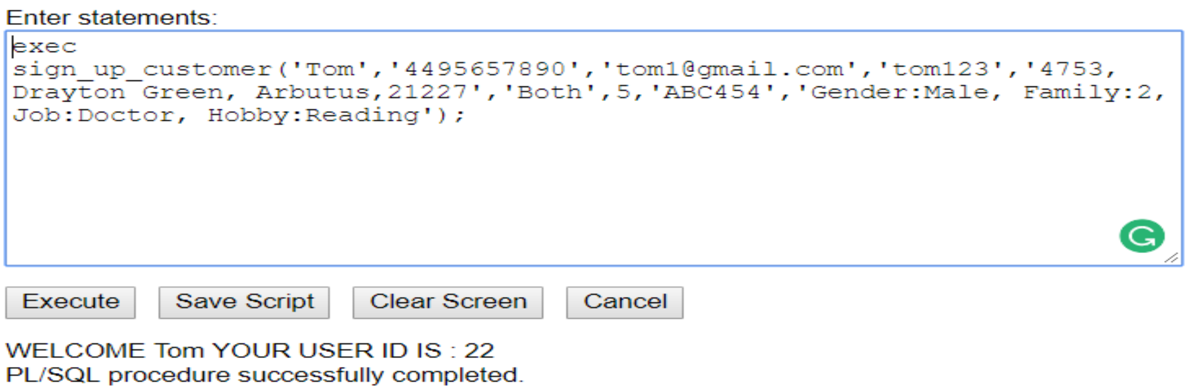
--New Guest


Enter statements:

```
exec
sign_up_customer('Niraj','4495657890','niraj1@gmail.com','niraj123','47
52, Drayton Green, Arbutus,21227','Guest',0,'PNC767','Gender:Male,
Family:3, Job:IT professional, Hobby:Travelling');
```

```
WELCOME Niraj YOUR USER ID IS : 21
PL/SQL procedure successfully completed.
```

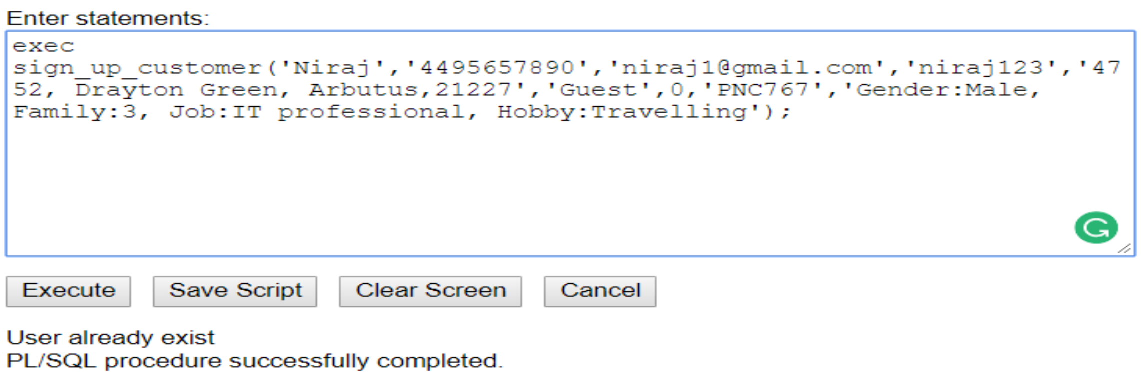
-- new both


Enter statements:

```
exec
sign_up_customer('Tom','4495657890','tom1@gmail.com','tom123','4753,
Drayton Green, Arbutus,21227','Both',5,'ABC454','Gender:Male, Family:2,
Job:Doctor, Hobby:Reading');
```

```
WELCOME Tom YOUR USER ID IS : 22
PL/SQL procedure successfully completed.
```

--Check for existing


Enter statements:

```
exec
sign_up_customer('Niraj','4495657890','niraj1@gmail.com','niraj123','47
52, Drayton Green, Arbutus,21227','Guest',0,'PNC767','Gender:Male,
Family:3, Job:IT professional, Hobby:Travelling');
```

```
User already exist
PL/SQL procedure successfully completed.
```

**-- FEATURE 2 : Allow a user to login**

**-- CREATE FUNCTION TO CHECK LOG IN CREDENTIALS**

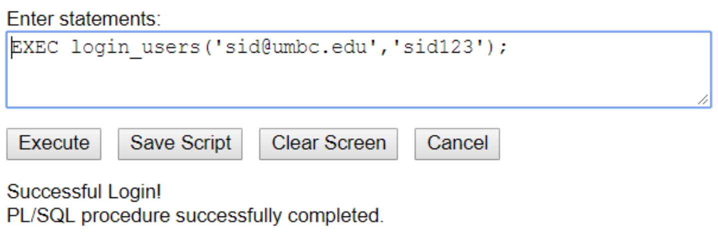
```
Create or replace function return_values(email_user in varchar2,password_user in varchar2)
return number
IS
pass varchar(10);
BEGIN
        select password into pass from system_user where email = email_user;
        if pass=password_user then
  return 1;
  else
  return 0;
  end if;
exception
        when no_data_found then

        return 0;
End;
```

**-- CREATE PROCEDURE**

```
Create or replace procedure login_users(email_user in VARCHAR2, password_user in VARCHAR2) IS
value number;
Begin
        value := return_values(email_user,password_user);
        if value=1 then
                dbms_output.put_line('Successful Login!');
                        else
                dbms_output.put_line('Invalid username and/or password!');
                dbms_output.put_line('Login Unsuccesful!');
        end if;
End;
```

**--Execution**

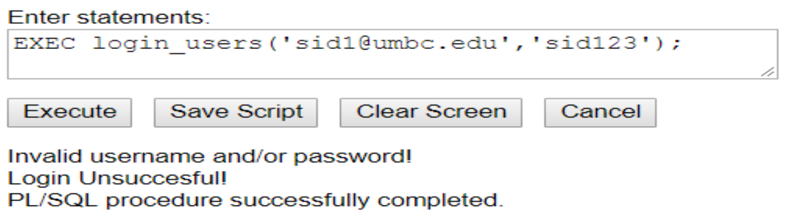
-- SUCCESSFUL LOGIN



```
EXEC login_users('sid@umbc.edu','sid123');
```

```
Successful Login!
PL/SQL procedure successfully completed.
```

-- UNSUCCESSFUL LOGIN


Enter statements:

```
EXEC login_users('sid1@umbc.edu','sid123');
```

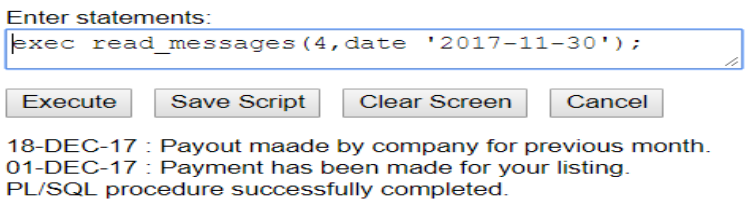
```
Invalid username and/or password!
Login Unsuccesful!
PL/SQL procedure successfully completed.
```

**-- FEATURE 3: Allow a user to read messages**

```
create or replace PROCEDURE read_messages(usr_id in integer, msg_dt in date) IS

Cursor msg_cursor is select Message_date, Message from messages where USER_ID=usr_id AND Message_date >= msg_dt and Message_date < sysdate;

message_text varchar(500);

message_dt date;

cursor_count integer:=0;

u1 int;

BEGIN

select count(*) into u1 from system_user where user_ID = usr_id;

if u1 = 0 then

dbms_output.put_line('User ID does not exist');

elsif msg_dt > sysdate then

dbms_output.put_line('Invalid date');

else Open msg_cursor;

Loop

        fetch msg_cursor into message_dt,message_text;

        exit when msg_cursor%notfound;

        dbms_output.put_line(message_dt||' : '||message_text);

        cursor_count:=cursor_count+1;

End loop;

        IF cursor_count=0 THEN  dbms_output.put_line('No messages found');

        END IF;

close msg_cursor;

end if;

END;
```

**--Execution**

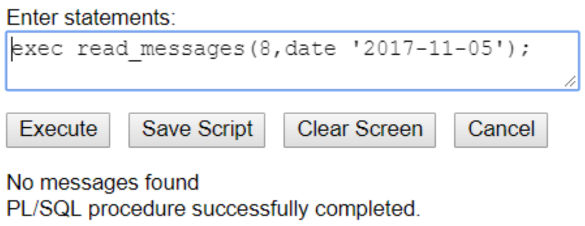
-- shows message


Enter statements:

```
exec read_messages(4,date '2017-11-30');
```

```
18-DEC-17 : Payout maade by company for previous month.
01-DEC-17 : Payment has been made for your listing.
PL/SQL procedure successfully completed.
```

-- no message


Enter statements:

```
exec read_messages(8,date '2017-11-05');
```

```
No messages found
PL/SQL procedure successfully completed.
```

**-- FEATURE 4: Allow a host to add a listing.**

**---Function to check if host exists**

```
create or replace function CHECK_EXISTING_HOST(HostID in int)
return int
IS
Check_Host number;
BEGIN
Check_Host := 0;
        select count(*) into Check_Host from Host where HostID = Host_ID;
IF CHECK_HOST=0 THEN DBMS_OUTPUT.PUT_LINE('NO SUCH HOST') ;
RETURN CHECK_HOST;
ELSE return Check_Host;
END IF;
End;
```

**-----Function to check if Listing exists**

```
create or replace function CHECK_EXISTING_LISTING (HostID in int,h_no in int,s_treet in varchar,z_code in int)
return int
IS
Check_Listing number;
hno int;
strt varchar(25);
```

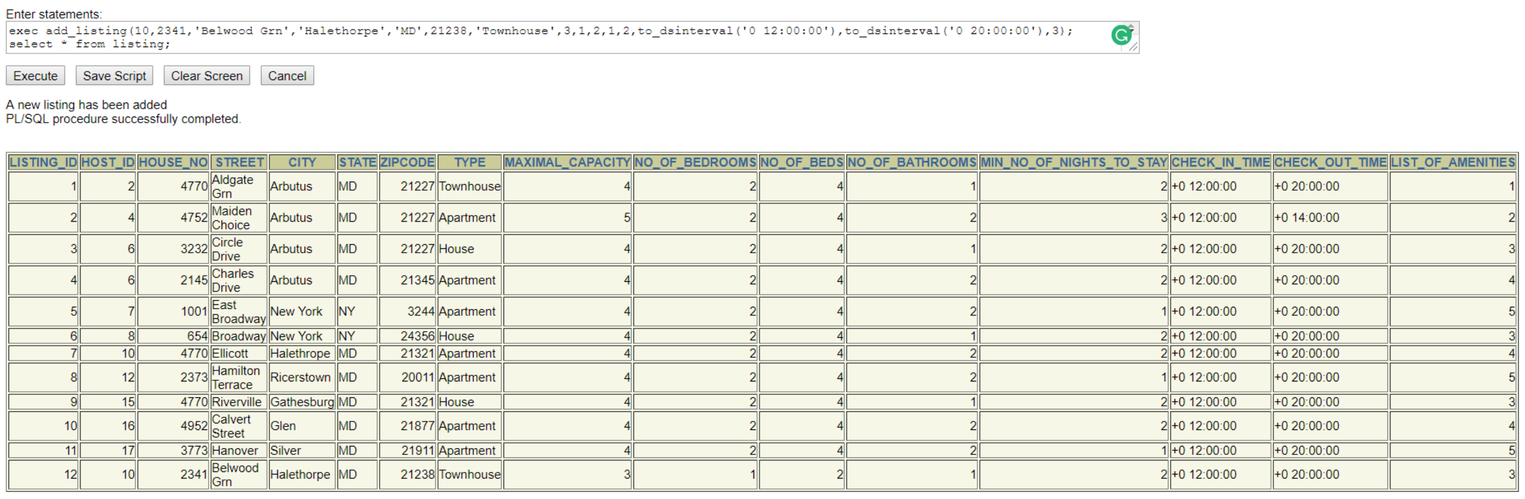
```
zcode int;

BEGIN

Check_Listing := 0;

select count(*) into Check_Listing from Listing l inner join host h on h.Host_ID = HostID and l.house_no= h_no ;

if Check_Listing=0 then

return Check_Listing;

else

return Check_Listing;

end if;

End;
```

**---main procedure**

```
CREATE OR REPLACE procedure add_listing(hostID in int,h_no in int, s_treet in varchar, c_ity in varchar,

s_tate in varchar, z_ipcode in int, t_ype in varchar, M_aximal_Capacity in int, Num_of_Bedrooms in int, Num_of_Beds in int,Num_of_Bathrooms in int,Min_Stay in int,In_Time interval day to second,Out_Time interval day to second,amenities int) IS

check_host number;

check_listing number;

BEGIN

check_host := CHECK_EXISTING_HOST(hostID);

IF check_host > 0 THEN

            check_listing :=CHECK_EXISTING_LISTING(hostID,h_no,s_treet,z_ipcode);

    IF check_listing = 0 THEN

             INSERT INTO listing values(LISTING_SEQ.nextval,hostID,h_no,s_treet,c_ity,s_tate,z_ipcode,t_ype,M_aximal_Capacity,Num_of_Bedrooms,Num_of_Beds,Num_of_Bathrooms,Min_Stay,In_Time,Out_Time,amenities);

            dbms_output.put_line('A new listing has been added');

    ELSE

            dbms_output.put_line('Listing already exists');

    end if;

else

    dbms_output.put_line('No such Host exists');

END IF; END;
```

**--Execution**

--new listing

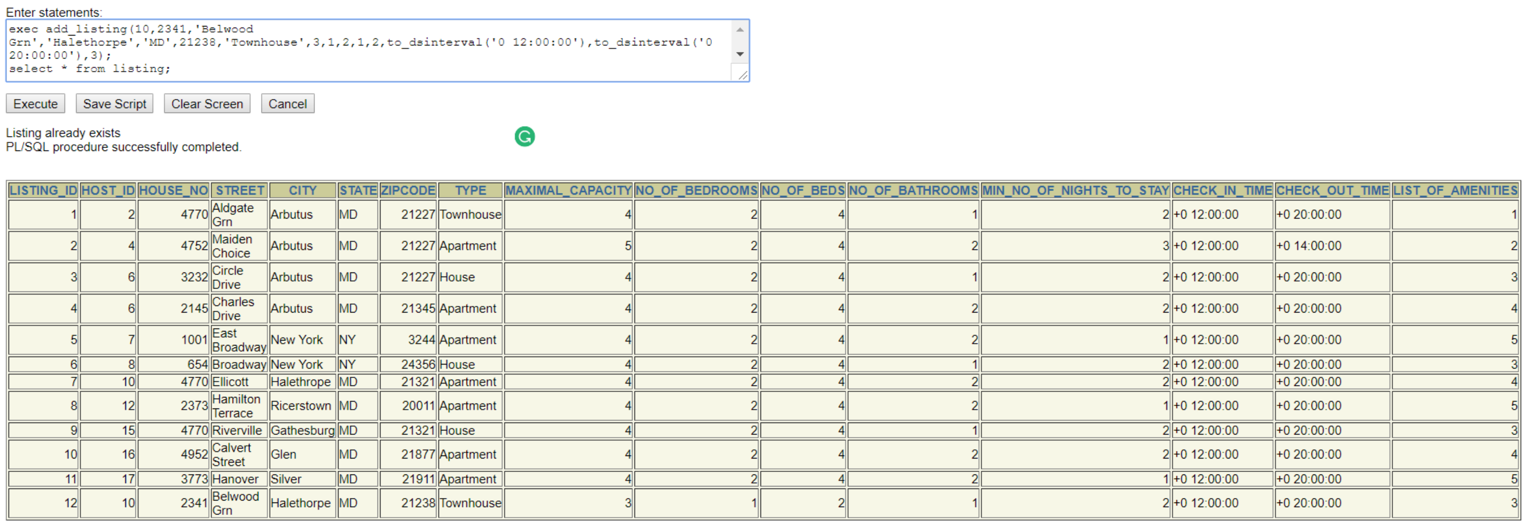

Enter statements:

```
exec add_listing(10,2341,'Belwood Grn','Halethorpe','MD',21238,'Townhouse',3,1,2,1,2,to_dsinterval('0 12:00:00'),to_dsinterval('0 20:00:00'),3);
select * from listing;
```

```
A new listing has been added
PL/SQL procedure successfully completed.
```

| LISTING_ID | HOST_ID | HOUSE_NO | STREET | CITY | STATE | ZIPCODE | TYPE | MAXIMAL_CAPACITY | NO_OF_BEDROOMS | NO_OF_BEDS | NO_OF_BATHROOMS | MIN_NO_OF_NIGHTS_TO_STAY | CHECK_IN_TIME | CHECK_OUT_TIME | LIST_OF_AMENITIES |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 2 | 4770 | Aldgate Grn | Arbutus | MD | 21227 | Townhouse | 4 | 2 | 4 | 1 | 2 | +0 12:00:00 | +0 20:00:00 | 1 |
| 2 | 4 | 4752 | Maiden Choice | Arbutus | MD | 21227 | Apartment | 5 | 2 | 4 | 2 | 3 | +0 12:00:00 | +0 14:00:00 | 2 |
| 3 | 6 | 3232 | Circle Drive | Arbutus | MD | 21227 | House | 4 | 2 | 4 | 1 | 2 | +0 12:00:00 | +0 20:00:00 | 3 |
| 4 | 6 | 2145 | Charles Drive | Arbutus | MD | 21345 | Apartment | 4 | 2 | 4 | 2 | 2 | +0 12:00:00 | +0 20:00:00 | 4 |
| 5 | 7 | 1001 | East Broadway | New York | NY | 3244 | Apartment | 4 | 2 | 4 | 2 | 1 | +0 12:00:00 | +0 20:00:00 | 5 |
| 6 | 8 | 654 | Broadway | New York | NY | 24356 | House | 4 | 2 | 4 | 1 | 2 | +0 12:00:00 | +0 20:00:00 | 3 |
| 7 | 10 | 4770 | Ellicott | Halethrope | MD | 21321 | Apartment | 4 | 2 | 4 | 2 | 2 | +0 12:00:00 | +0 20:00:00 | 4 |
| 8 | 12 | 2373 | Hamilton Terrace | Ricerstown | MD | 20011 | Apartment | 4 | 2 | 4 | 2 | 1 | +0 12:00:00 | +0 20:00:00 | 5 |
| 9 | 15 | 4770 | Riverville | Gathesburg | MD | 21321 | House | 4 | 2 | 4 | 1 | 2 | +0 12:00:00 | +0 20:00:00 | 3 |
| 10 | 16 | 4952 | Calvert Street | Glen | MD | 21877 | Apartment | 4 | 2 | 4 | 2 | 2 | +0 12:00:00 | +0 20:00:00 | 4 |
| 11 | 17 | 3773 | Hanover | Silver | MD | 21911 | Apartment | 4 | 2 | 4 | 2 | 1 | +0 12:00:00 | +0 20:00:00 | 5 |
| 12 | 10 | 2341 | Belwood Grn | Halethorpe | MD | 21238 | Townhouse | 3 | 1 | 2 | 1 | 2 | +0 12:00:00 | +0 20:00:00 | 3 |

--existing listing


Enter statements:

```
exec add_listing(10,2341,'Belwood
Grn','Halethorpe','MD',21238,'Townhouse',3,1,2,1,2,to_dsinterval('0 12:00:00'),to_dsinterval('0
20:00:00'),3);
select * from listing;
```

```
Listing already exists
PL/SQL procedure successfully completed.
```

| LISTING_ID | HOST_ID | HOUSE_NO | STREET | CITY | STATE | ZIPCODE | TYPE | MAXIMAL_CAPACITY | NO_OF_BEDROOMS | NO_OF_BEDS | NO_OF_BATHROOMS | MIN_NO_OF_NIGHTS_TO_STAY | CHECK_IN_TIME | CHECK_OUT_TIME | LIST_OF_AMENITIES |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 2 | 4770 | Aldgate Grn | Arbutus | MD | 21227 | Townhouse | 4 | 2 | 4 | 1 | 2 | +0 12:00:00 | +0 20:00:00 | 1 |
| 2 | 4 | 4752 | Maiden Choice | Arbutus | MD | 21227 | Apartment | 5 | 2 | 4 | 2 | 3 | +0 12:00:00 | +0 14:00:00 | 2 |
| 3 | 6 | 3232 | Circle Drive | Arbutus | MD | 21227 | House | 4 | 2 | 4 | 1 | 2 | +0 12:00:00 | +0 20:00:00 | 3 |
| 4 | 6 | 2145 | Charles Drive | Arbutus | MD | 21345 | Apartment | 4 | 2 | 4 | 2 | 2 | +0 12:00:00 | +0 20:00:00 | 4 |
| 5 | 7 | 1001 | East Broadway | New York | NY | 3244 | Apartment | 4 | 2 | 4 | 2 | 1 | +0 12:00:00 | +0 20:00:00 | 5 |
| 6 | 8 | 654 | Broadway | New York | NY | 24356 | House | 4 | 2 | 4 | 1 | 2 | +0 12:00:00 | +0 20:00:00 | 3 |
| 7 | 10 | 4770 | Ellicott | Halethrope | MD | 21321 | Apartment | 4 | 2 | 4 | 2 | 2 | +0 12:00:00 | +0 20:00:00 | 4 |
| 8 | 12 | 2373 | Hamilton Terrace | Ricerstown | MD | 20011 | Apartment | 4 | 2 | 4 | 2 | 1 | +0 12:00:00 | +0 20:00:00 | 5 |
| 9 | 15 | 4770 | Riverville | Gathesburg | MD | 21321 | House | 4 | 2 | 4 | 1 | 2 | +0 12:00:00 | +0 20:00:00 | 3 |
| 10 | 16 | 4952 | Calvert Street | Glen | MD | 21877 | Apartment | 4 | 2 | 4 | 2 | 2 | +0 12:00:00 | +0 20:00:00 | 4 |
| 11 | 17 | 3773 | Hanover | Silver | MD | 21911 | Apartment | 4 | 2 | 4 | 2 | 1 | +0 12:00:00 | +0 20:00:00 | 5 |
| 12 | 10 | 2341 | Belwood Grn | Halethorpe | MD | 21238 | Townhouse | 3 | 1 | 2 | 1 | 2 | +0 12:00:00 | +0 20:00:00 | 3 |

**--Feature 5: Allow a host to enter an availability period for a listing**

**--Function to check if listing_ID exists**

```
create or replace function CHECK_EXISTING_LISTING_ID(ListingID in int)
return int
IS
Check_ListingID number;
BEGIN
Check_ListingID := 0;
        select count(*) into Check_ListingID from Listing where Listing_ID = ListingID;
return Check_ListingID;
exception
        when no_data_found then
  dbms_output.put_line('No such Listing exists');
  return 0;
End;

-----Function to  check if period falls in existing period
create or replace function CHECK_EXISTING_PERIOD(ListingID in int,s_date in date, e_date in date)
return int
IS
cursor c1 is select Listing_ID,Start_Date,End_Date from listing_availability_period where Listing_ID=ListingID;
Check_Period number;
data c1%rowtype;
BEGIN
Check_Period := 0;
open c1;
loop
   fetch c1 into data;
      if (s_date<data.Start_Date and e_date<data.Start_Date) or (s_date>data.End_Date and e_date>data.End_Date)
then
         Check_Period :=0;
                        return Check_Period;
      elsif(c1%notfound) then
         Check_Period :=0;
                        return Check_Period;
         exit;


      else
                        Check_Period := 1;
                        return Check_Period;
         exit;


               end if;



exit when c1%notfound;
```

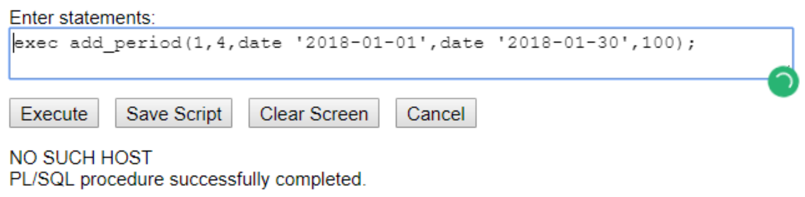
```
end loop;

close c1;

End;
```

**----main procedure**

```
CREATE OR REPLACE procedure add_period(HostID in int,ListingID in int,s_date in date, e_date in date,price int)IS

check_host number;

check_listingID number;

check_period number;

Begin

check_host := CHECK_EXISTING_HOST(hostID);

IF check_host = 1 THEN

            check_listingID :=CHECK_EXISTING_LISTING_ID(ListingID);

    IF check_listingID =0 THEN

    dbms_output.put_line('Listing does not exist');

    ELSE

        if e_date<=s_date then

         dbms_output.put_line('Enter valid dates');

        else

            check_period:=CHECK_EXISTING_PERIOD(ListingID,s_date,e_date);

            if check_period=0 then

                 INSERT INTO Listing_Availability_Period
values(LISTING_AVAIL_PERIOD_SEQ.nextval,ListingID,s_date,e_date,price);

                                                  dbms_output.put_line('A new listing period has been added');

            else dbms_output.put_line('Listing period already exists');

        end if;  end if;

    end if;

END IF; END;
```

**--Execution**

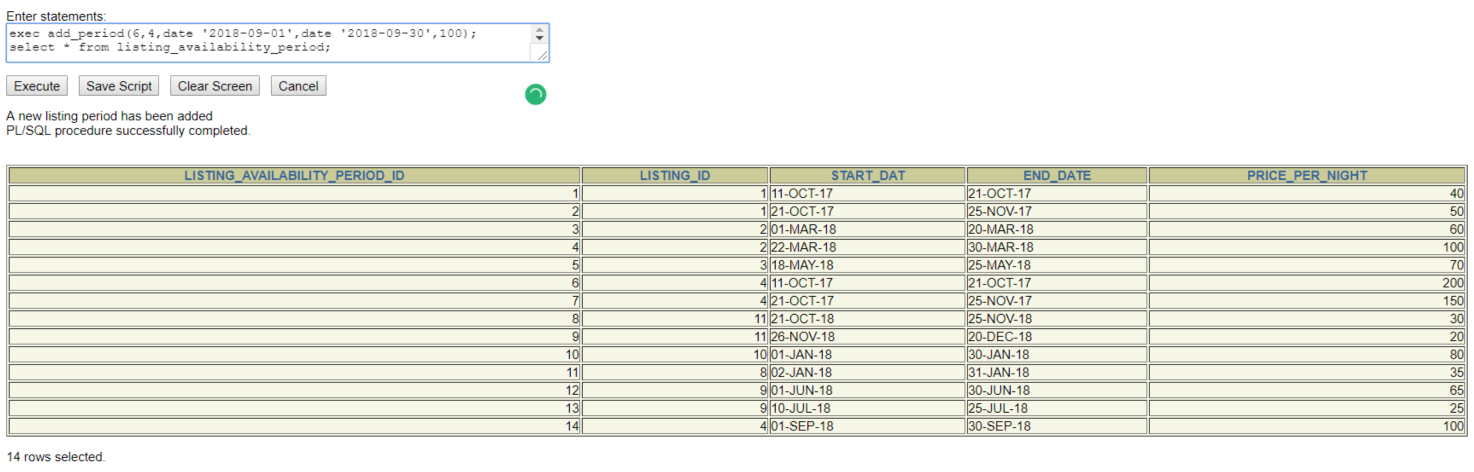
```
--no such host
```



```
exec add_period(1,4,date '2018-01-01',date '2018-01-30',100);
```

```
NO SUCH HOST
PL/SQL procedure successfully completed.
```

--new listing period



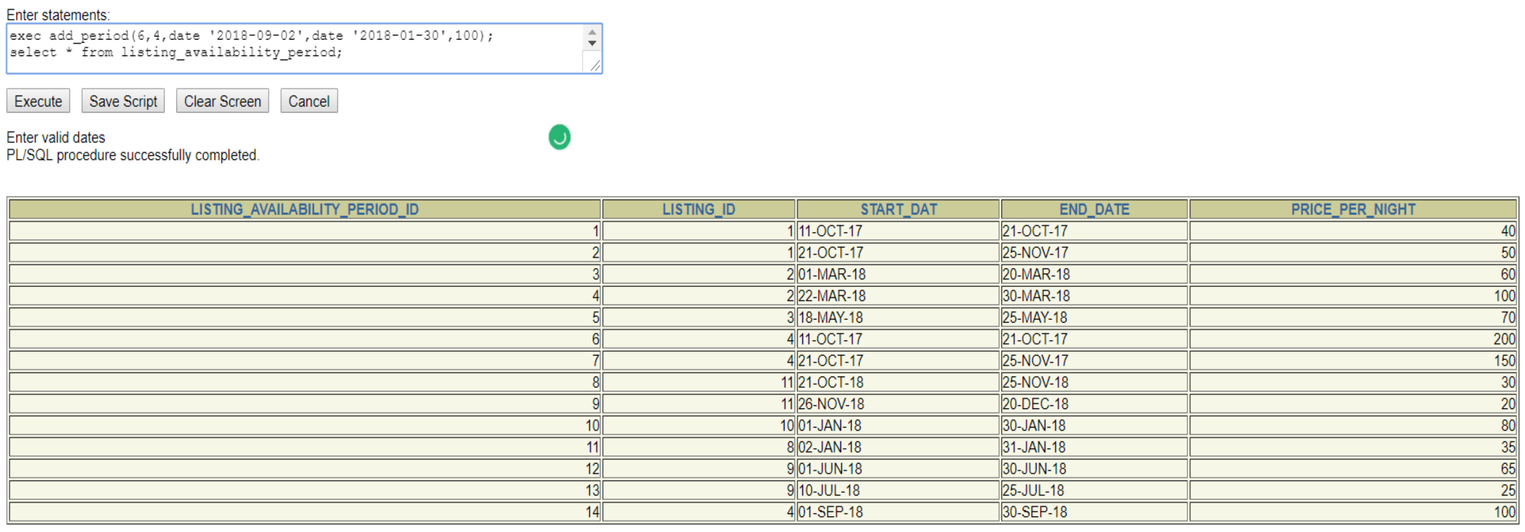
```
exec add_period(6,4,date '2018-09-01',date '2018-09-30',100);
select * from listing_availability_period;
```

```
A new listing period has been added
PL/SQL procedure successfully completed.
```

| LISTING_AVAILABILITY_PERIOD_ID | LISTING_ID | START_DAT | END_DATE | PRICE_PER_NIGHT |
|---|---|---|---|---|
| 1 | 1 | 11-OCT-17 | 21-OCT-17 | 40 |
| 2 | 1 | 21-OCT-17 | 25-NOV-17 | 50 |
| 3 | 2 | 01-MAR-18 | 20-MAR-18 | 60 |
| 4 | 2 | 22-MAR-18 | 30-MAR-18 | 100 |
| 5 | 3 | 18-MAY-18 | 25-MAY-18 | 70 |
| 6 | 4 | 11-OCT-17 | 21-OCT-17 | 200 |
| 7 | 4 | 21-OCT-17 | 25-NOV-17 | 150 |
| 8 | 11 | 21-OCT-18 | 25-NOV-18 | 30 |
| 9 | 11 | 26-NOV-18 | 20-DEC-18 | 20 |
| 10 | 10 | 01-JAN-18 | 30-JAN-18 | 80 |
| 11 | 8 | 02-JAN-18 | 31-JAN-18 | 35 |
| 12 | 9 | 01-JUN-18 | 30-JUN-18 | 65 |
| 13 | 9 | 10-JUL-18 | 25-JUL-18 | 25 |
| 14 | 4 | 01-SEP-18 | 30-SEP-18 | 100 |

```
14 rows selected.
```

--Listing period already exists



```
exec add_period(6,4,date '2018-09-02',date '2018-01-30',100);
select * from listing_availability_period;
```

```
Enter valid dates
PL/SQL procedure successfully completed.
```

| LISTING_AVAILABILITY_PERIOD_ID | LISTING_ID | START_DAT | END_DATE | PRICE_PER_NIGHT |
|---|---|---|---|---|
| 1 | 1 | 11-OCT-17 | 21-OCT-17 | 40 |
| 2 | 1 | 21-OCT-17 | 25-NOV-17 | 50 |
| 3 | 2 | 01-MAR-18 | 20-MAR-18 | 60 |
| 4 | 2 | 22-MAR-18 | 30-MAR-18 | 100 |
| 5 | 3 | 18-MAY-18 | 25-MAY-18 | 70 |
| 6 | 4 | 11-OCT-17 | 21-OCT-17 | 200 |
| 7 | 4 | 21-OCT-17 | 25-NOV-17 | 150 |
| 8 | 11 | 21-OCT-18 | 25-NOV-18 | 30 |
| 9 | 11 | 26-NOV-18 | 20-DEC-18 | 20 |
| 10 | 10 | 01-JAN-18 | 30-JAN-18 | 80 |
| 11 | 8 | 02-JAN-18 | 31-JAN-18 | 35 |
| 12 | 9 | 01-JUN-18 | 30-JUN-18 | 65 |
| 13 | 9 | 10-JUL-18 | 25-JUL-18 | 25 |
| 14 | 4 | 01-SEP-18 | 30-SEP-18 | 100 |

**-----feature 6: Look up available houses at a given city and state and in a given period.**

**-----Function to check if city and state combo exists**

create or replace function CHECK_EXISTING_CITY_STATE(c_ity varchar,s_tate varchar)

return int

IS

Check_Combination number;

BEGIN

Check_Combination := 0;

```
select count(*) into Check_Combination from Listing where city = c_ity and state=s_tate;

return Check_Combination;

exception

        when no_data_found then

  dbms_output.put_line('Enter correct city and state');

  return 0;

End;
```

**----------Function to check if check in and check out date fall in any listing availability period**

```
create or replace function CHECK_EXISTING_CINCOUT(checkin date,checkout date,lid int)

return int

IS

count_cin number;

count_cout number;

BEGIN

count_cin := 0;

count_cout:=0;

select count(listing_availability_period_id)  into count_cin from listing_availability_period where(checkin between start_date and end_date) and listing_id=lid;

select count(listing_availability_period_id)  into count_cout from listing_availability_period where(checkout between start_date and end_date) and listing_id=lid;

if(count_cin!=0 and count_cout!=0) then

return 1;

--else

--

else

--dbms_output.put_line('No listing Found');

return 0;

--dbms_output.put_line('hi');

end if;

End;
```

**-------Function to check if period is covered and compute total cost**

```
create or replace procedure CHECK_EXISTING_DATES(checkin in date,checkout in date,lid in int,l_id out int,tc out
int)

IS

cursor c1 is select listing_availability_period_id,listing_id, start_date, end_date from listing_availability_period where
((checkin between start_date and end_date) or (checkout between start_date and end_date)) and listing_id=lid ;


Check_date number;

cin date;

cout date;

--for start_date

s date;

--for end_date

e date;

--for listing

l int;

--for listing_availability_period_ID

lp int;

total_stay int;

cost int;

--for price

p int;

partial_stay int;

total_cost int;

r int;

BEGIN

total_cost:=0;

cin:=checkin;

cout:=checkout;

total_stay:=cout-cin+1;

cost:=0;

-- this works

--dbms_output.put_line(checkin || ',' || checkout || ',' || lid );

r:=CHECK_EXISTING_CINCOUT(checkin,checkout,lid);

--this works
```

```
--dbms_output.put_line(checkin || ',' || checkout || ',' || lid );
-- this works
--dbms_output.put_line(r || 'is its value');
if (r=1) then
open c1;
loop
--dbms_output.put_line('Hi');
    fetch c1 into lp,l,s,e;
            --dbms_output.put_line(lp || ',' ||l||','||s||','||e);
    if ((cin between s and e) and(cout between s and e))then
            --dbms_output.put_line('1a');
         select price_per_night into p from listing_availability_period where listing_id=l and start_date=s and end_date=e;
         cost:=cost+(p*total_stay);
                        --dbms_output.put_line('Hi');
          exit;
    elsif((cin between s and e) and(cout not between s and e))then
            --dbms_output.put_line('1b');
        partial_stay:=e-cin;
        select price_per_night into p from listing_availability_period where listing_id=l and start_date=s and end_date=e;
        cost:=cost+(p*partial_stay);
                        --dbms_output.put_line('Hi1');
         cin:=e;
         total_stay:=cout-cin+1;
--dbms_output.put_line('Hi2');
   elsif((cout between s and e) and(cin not between s and e))then
  -- dbms_output.put_line('1c');
       if(cin!=s) then
               cost:=0;
               exit;
       end if;
    elsif(cin!=s) then
                dbms_output.put_line('Entire period does not fit');
                cost:=0;
            exit;
    end if;
     exit when c1%notfound;
end loop;
total_cost:=cost*1.05;
if total_cost>0 then
    tc:=total_cost;
    l_id:=lid;
end if;
close c1;
end if;
End;
```

**-----------------------------main procedure**

```
create or replace procedure RETRIEVE_LISTINGID_NOTBOOKED(c_ity varchar,s_tate varchar,checkin in date,
checkout in date)
as
lst number;
lap number;
lid number;
tc number;
--counter int;
h number;
s varchar2(25);
c varchar2(10);
st varchar2(5);
z number;
combo number;
cnt number;
check_val number;
ccnt number;
cursor c1 is select b.listing_id from booking_information b, listing l,listing_availability_period la where
BOOKING_STATUS='Requested' and ((checkin between b.check_in_date and b.check_out_date)or (checkout between
b.check_in_date and b.check_out_date)) and city=c_ity and state=s_tate and b.listing_id=la.listing_id union (select
l.listing_id from listing l, listing_availability_period la where city=c_ity and state=s_tate and la.listing_id=l.listing_id);

BEGIN

check_val:=0;

ccnt:=0;

combo:=CHECK_EXISTING_CITY_STATE(c_ity,s_tate);

if combo=0 then dbms_output.put_line('Enter correct City and State ');

elsif (checkout<=checkin) then dbms_output.put_line('Enter valid Checkin and Checkout dates');

else

--counter:=0;

open c1;

   loop


      fetch c1 into lst;

                  --dbms_output.put_line('hieee');


                  --dbms_output.put_line(lst|| ',' || lid);

                  CHECK_EXISTING_DATES(checkin,checkout,lst,lid,tc);

                  --dbms_output.put_line(lst|| ',' || lid);









                  if (ccnt != lid) then


                  if (tc!=0 and lid=lst) then

                  check_val:=1;


                                                                                    select count(*)into cnt from listing where
listing_id=lid and city=c_ity and state=s_tate;



                                                                                    if cnt>0 then
```

```
                        select house_no,street,city,state,zipcode into h,s,c,st,z from listing where listing_id=lst and
city=c_ity and state=s_tate;

                                                            ccnt:=lst;



                        dbms_output.put_line('Listing ID: '|| lst||'  Address:'||h||' '||s||' '||c||' '||st||' '||z||'  Total cost:'||tc);
                        --counter:=counter-1;



                                      else dbms_output.put_line('No listing found');
                                      end if;


                 elsif(c1%notfound) then
                 exit;
                 else
                 check_val:=1;
        --continue;
             end if;




                 if (c1%notfound and check_val<1) then dbms_output.put_line('No listing found');
                 end if;


       if check_val=0 then
        exit;
       end if;


             --:=counter+1;


        end if ;
exit when c1%notfound;
```

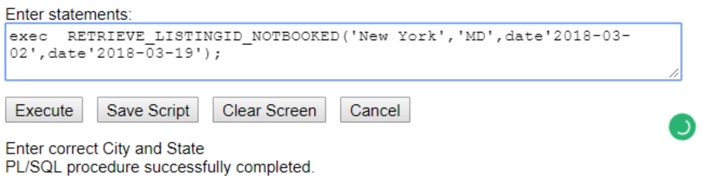
end loop;

close c1;

end if; End;

**--Execution**

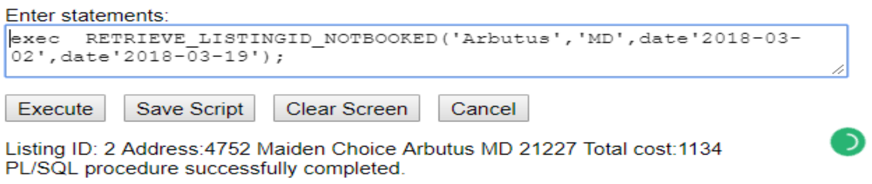
-- Enter correct City and State



```
exec  RETRIEVE_LISTINGID_NOTBOOKED('New York','MD',date'2018-03-02',date'2018-03-19');
```

```
Enter correct City and State
PL/SQL procedure successfully completed.
```

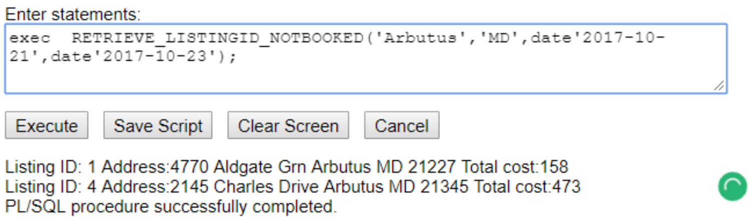
-- Shows one listing for which the check in and check out date falls into



```
exec  RETRIEVE_LISTINGID_NOTBOOKED('Arbutus','MD',date'2018-03-02',date'2018-03-19');
```

```
Listing ID: 2 Address:4752 Maiden Choice Arbutus MD 21227 Total cost:1134
PL/SQL procedure successfully completed.
```

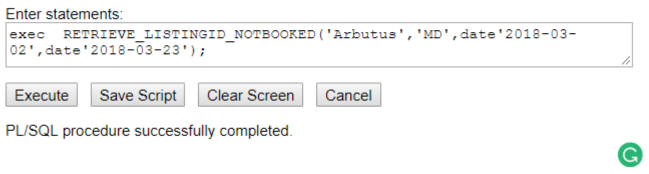
-- shows two outputs ( two listing) from 4 outputs where the checkin and check out dates fall into two different listing periods



```
exec  RETRIEVE_LISTINGID_NOTBOOKED('Arbutus','MD',date'2017-10-21',date'2017-10-23');
```

```
Listing ID: 1 Address:4770 Aldgate Grn Arbutus MD 21227 Total cost:158
Listing ID: 4 Address:2145 Charles Drive Arbutus MD 21345 Total cost:473
PL/SQL procedure successfully completed.
```

-- shows no listing as there is a gap between two listing periods , but the checkin and check out are falling into two listing periods



```
exec  RETRIEVE_LISTINGID_NOTBOOKED('Arbutus','MD',date'2018-03-02',date'2018-03-23');
```

```
PL/SQL procedure successfully completed.
```

**------feature 7: Booking request**

**-----Function to check if guest exists**

create or replace function CHECK_EXISTING_GUEST(GuestID in int)

return int

IS

Check_Guest number;

BEGIN

```
Check_Guest := 0;
        select count(*) into Check_Guest from Guest where GuestID = Guest_ID;
return Check_Guest;
exception
        when no_data_found then
  dbms_output.put_line('No such Guest exists');
  return 0;
End;
```

**------Function to check max capacity is met**

```
create or replace function CHECK_MAX_CAPACITY(listing int,adutls int,kids int)
return int
IS
total number;
max_cap int;
BEGIN
total:=0;
total:=adutls+kids;
select maximal_capacity into max_cap from listing where listing_id=listing;
if(max_cap>=total) then return 0;
else return 1;
end if;
End;
```

**------Function to check min stay requirement**

```
create or replace function CHECK_MIN_STAY(listing int,checkin date,checkout date)
return int
IS
stay number;
min_stay int;
BEGIN
stay:=0;
stay:=checkout-checkin+1;
select MIN_NO_OF_NIGHTS_TO_STAY into min_stay from listing where listing_id=listing;
```

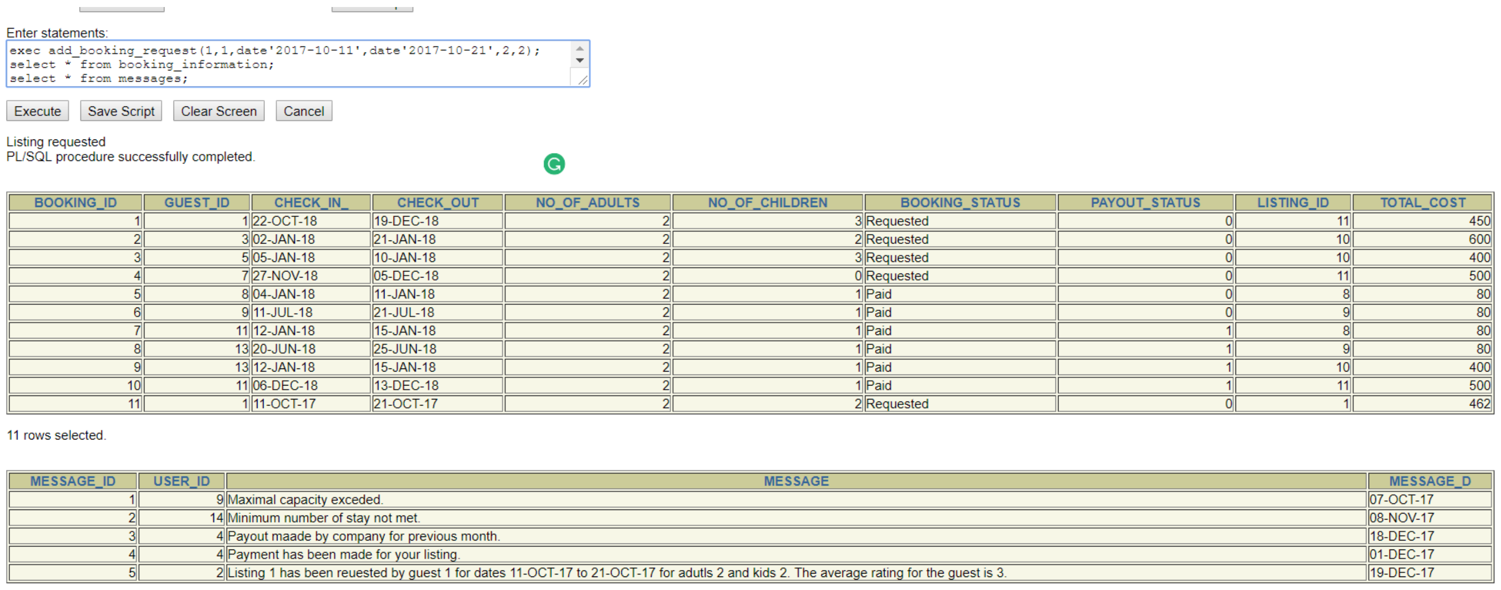
```
if(min_stay<=stay) then return 0;
else return 1;
end if;
End;
```

**------Procedure to insert booking request**

```
CREATE OR REPLACE procedure add_booking_request(l_id int,guest int,checkin date,checkout date,adults int,kids
int) IS
check_guest number;
check_listing number;
check_cap number;
check_stay number;
host number;
lid number;
tc number;
rating float;

begin

check_guest:=CHECK_EXISTING_GUEST(guest);
check_listing := CHECK_EXISTING_LISTING_ID(l_id);
if check_guest = 0 THEN dbms_output.put_line('Guest does not exist');
elsif check_listing = 0 THEN dbms_output.put_line('Listing does not exist');
elsif (checkin>=checkout) THEN dbms_output.put_line('Enter correct Checkin and Checkout dates');
else
   select host_id into host from listing where listing_id=l_id;
   CHECK_EXISTING_DATES(checkin,checkout,l_id,lid,tc);



   if (tc!=0 and l_id=lid)  then
                    check_cap:=CHECK_MAX_CAPACITY(l_id,adults,kids);
                    check_stay:=CHECK_MIN_STAY(l_id,checkin,checkout);

         if check_cap!=0 then dbms_output.put_line('Maximal capacity exceeded');

            Insert into Messages values (MESSAGES_SEQ.nextval, guest,'Maximal capacity exceeded',sysdate);


          elsif check_stay!=0 then dbms_output.put_line('Minimum number of stay requirements not met');

             Insert into Messages values (MESSAGES_SEQ.nextval, guest,'Minimum number of stay requirements not
met',sysdate);

          else

             INSERT INTO booking_information
values(BOOKING_SEQ.nextval,guest,checkin,checkout,adults,kids,'Requested',0,l_id,tc);

                              dbms_output.put_line('Listing requested');

             select average_rating into rating from guest where guest_id=guest;

             Insert into Messages values (MESSAGES_SEQ.nextval, host,'Listing '||l_id||' has been reuested by guest
'||guest||' for dates '||checkin||' to '||checkout||' for adutls '||adults||' and kids '||kids||'. The average rating for the guest is
'||rating||'.',sysdate);


          end if;

            else dbms_output.put_line('Listing cannot be requested');

      end if;

end if;

end;
```

**-Execution**

---successfully requested



Enter statements:

```
exec add_booking_request(1,1,date'2017-10-11',date'2017-10-21',2,2);
select * from booking_information;
select * from messages;
```

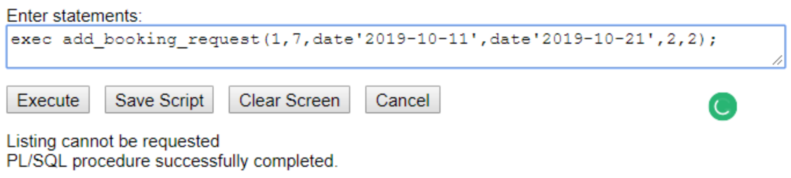
Listing requested
PL/SQL procedure successfully completed.

| BOOKING_ID | GUEST_ID | CHECK_IN_ | CHECK_OUT | NO_OF_ADULTS | NO_OF_CHILDREN | BOOKING_STATUS | PAYOUT_STATUS | LISTING_ID | TOTAL_COST |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 22-OCT-18 | 19-DEC-18 | 2 | 3 | Requested | 0 | 11 | 450 |
| 2 | 3 | 02-JAN-18 | 21-JAN-18 | 2 | 2 | Requested | 0 | 10 | 600 |
| 3 | 5 | 05-JAN-18 | 10-JAN-18 | 2 | 3 | Requested | 0 | 10 | 400 |
| 4 | 7 | 27-NOV-18 | 05-DEC-18 | 2 | 0 | Requested | 0 | 11 | 500 |
| 5 | 8 | 04-JAN-18 | 11-JAN-18 | 2 | 1 | Paid | 0 | 8 | 80 |
| 6 | 9 | 11-JUL-18 | 21-JUL-18 | 2 | 1 | Paid | 0 | 9 | 80 |
| 7 | 11 | 12-JAN-18 | 15-JAN-18 | 2 | 1 | Paid | 1 | 8 | 80 |
| 8 | 13 | 20-JUN-18 | 25-JUN-18 | 2 | 1 | Paid | 1 | 9 | 80 |
| 9 | 13 | 12-JAN-18 | 15-JAN-18 | 2 | 1 | Paid | 1 | 10 | 400 |
| 10 | 11 | 06-DEC-18 | 13-DEC-18 | 2 | 1 | Paid | 1 | 11 | 500 |
| 11 | 1 | 11-OCT-17 | 21-OCT-17 | 2 | 2 | Requested | 0 | 1 | 462 |

11 rows selected.

| MESSAGE_ID | USER_ID | MESSAGE | MESSAGE_D |
|---|---|---|---|
| 1 | 9 | Maximal capacity exceded. | 07-OCT-17 |
| 2 | 14 | Minimum number of stay not met. | 08-NOV-17 |
| 3 | 4 | Payout maade by company for previous month. | 18-DEC-17 |
| 4 | 4 | Payment has been made for your listing. | 01-DEC-17 |
| 5 | 2 | Listing 1 has been reuested by guest 1 for dates 11-OCT-17 to 21-OCT-17 for adutls 2 and kids 2. The average rating for the guest is 3. | 19-DEC-17 |

--cannot be requested



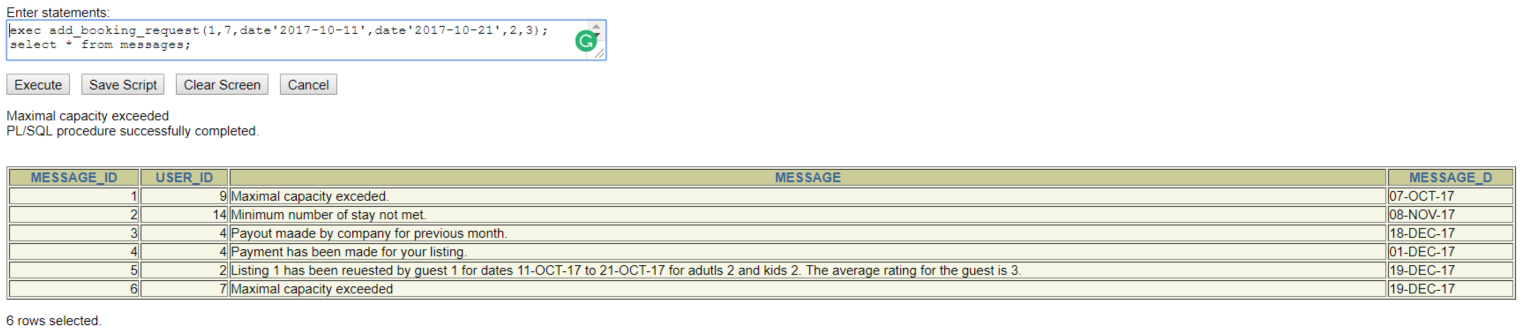
Enter statements:

```
exec add_booking_request(1,7,date'2019-10-11',date'2019-10-21',2,2);
```

Execute | Save Script | Clear Screen | Cancel

```
Listing cannot be requested
PL/SQL procedure successfully completed.
```

---maximum capacity exceeded



Enter statements:

```
exec add_booking_request(1,7,date'2017-10-11',date'2017-10-21',2,3);
select * from messages;
```

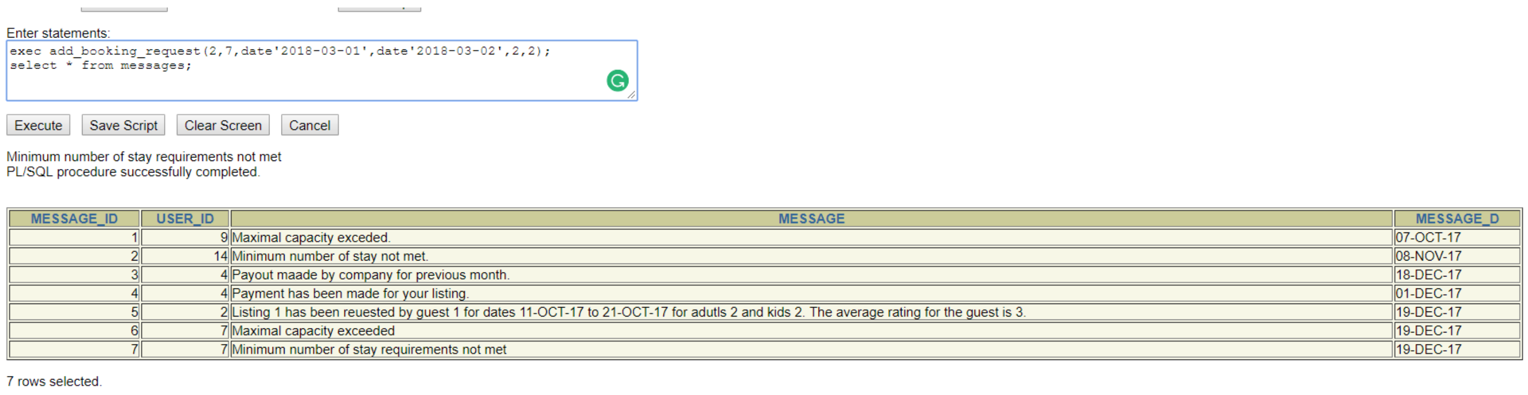
Execute | Save Script | Clear Screen | Cancel

```
Maximal capacity exceeded
PL/SQL procedure successfully completed.
```

| MESSAGE_ID | USER_ID | MESSAGE | MESSAGE_D |
|---|---|---|---|
| 1 | 9 | Maximal capacity exceded. | 07-OCT-17 |
| 2 | 14 | Minimum number of stay not met. | 08-NOV-17 |
| 3 | 4 | Payout maade by company for previous month. | 18-DEC-17 |
| 4 | 4 | Payment has been made for your listing. | 01-DEC-17 |
| 5 | 2 | Listing 1 has been reuested by guest 1 for dates 11-OCT-17 to 21-OCT-17 for adutls 2 and kids 2. The average rating for the guest is 3. | 19-DEC-17 |
| 6 | 7 | Maximal capacity exceeded | 19-DEC-17 |

6 rows selected.

----minimum stay requirement not met



Enter statements:

```
exec add_booking_request(2,7,date'2018-03-01',date'2018-03-02',2,2);
select * from messages;
```

Execute | Save Script | Clear Screen | Cancel

```
Minimum number of stay requirements not met
PL/SQL procedure successfully completed.
```

| MESSAGE_ID | USER_ID | MESSAGE | MESSAGE_D |
|---|---|---|---|
| 1 | 9 | Maximal capacity exceded. | 07-OCT-17 |
| 2 | 14 | Minimum number of stay not met. | 08-NOV-17 |
| 3 | 4 | Payout maade by company for previous month. | 18-DEC-17 |
| 4 | 4 | Payment has been made for your listing. | 01-DEC-17 |
| 5 | 2 | Listing 1 has been reuested by guest 1 for dates 11-OCT-17 to 21-OCT-17 for adutls 2 and kids 2. The average rating for the guest is 3. | 19-DEC-17 |
| 6 | 7 | Maximal capacity exceeded | 19-DEC-17 |
| 7 | 7 | Minimum number of stay requirements not met | 19-DEC-17 |

7 rows selected.

**-- Feature 8: Allow a host to approve or deny a booking request**

**-- function for checking existing booking id**

create or replace function CHECK_EXISTING_BOOK_R(BookingID in int)

return int

IS

Check_BookingID number;

BEGIN

Check_BookingID := 0;

select count(*) into Check_BookingID from Booking_Information where Booking_ID = BookingID and Booking_status = 'Requested';

if Check_BookingID=0 then return Check_BookingID;

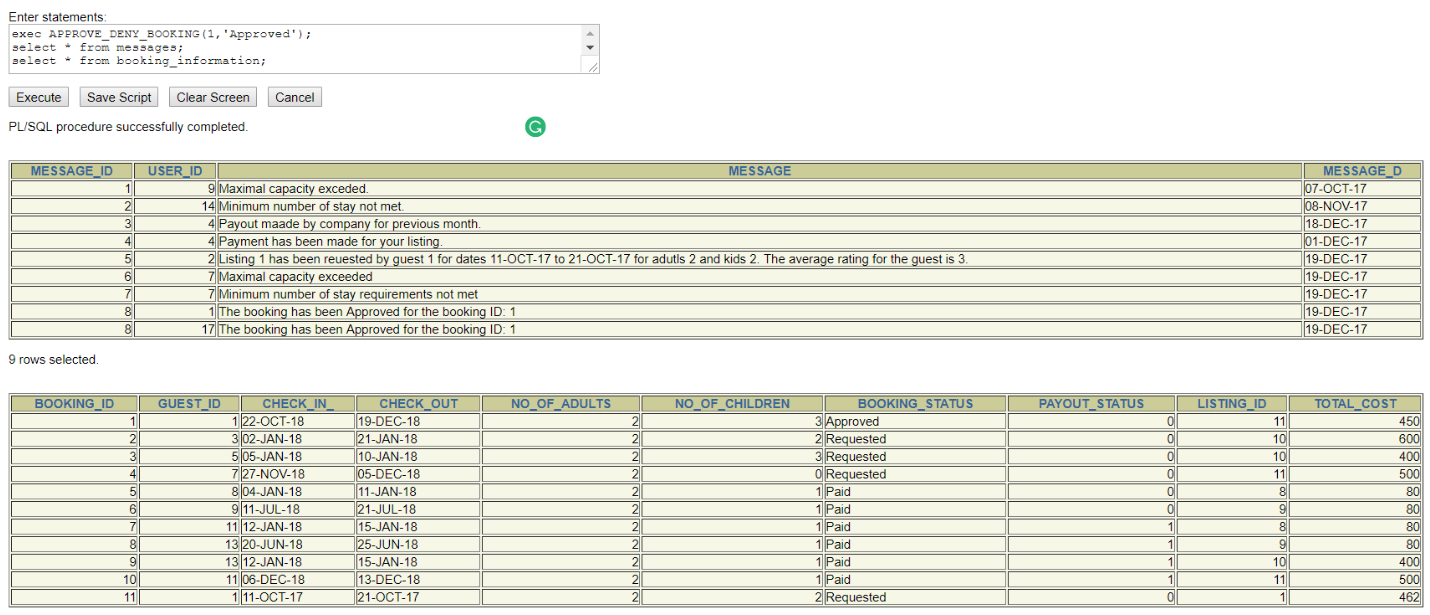
```
else return Check_BookingID;

end if;

End;
```

**-- Main procedure**

```
CREATE OR REPLACE procedure Approve_deny_Booking(BookingID in int, Decision in varchar) IS

Check_BookingID number;

gid number;

hid number;

valuee number;

BEGIN

valuee:=MESSAGES_SEQ.nextval;

    Check_BookingID :=CHECK_EXISTING_BOOK_R(BookingID);

        IF Check_BookingID = 1 THEN

                                    update Booking_information set Booking_Status = Decision where booking_ID=BookingID;

                                    select Guest_ID into gid from BOOKING_INFORMATION where booking_ID=BookingID;

                                    select host_ID into hid from Listing l,BOOKING_INFORMATION b where
b.booking_ID=BookingID and l.listing_ID = b.listing_ID;

                                    Insert into Messages values (valuee, gid,'The booking has been ' ||Decision|| ' for the booking ID:
' || BookingID ,sysdate);

                                    Insert into Messages values (valuee, hid,'The booking has been '||Decision|| ' for the booking ID: '
|| BookingID ,sysdate);

Else dbms_output.put_line('Booking does not exist for this host or has been Approved or Denied');

   end if; END;
```

**-Execution**

```
--booking approved
```



```
exec APPROVE_DENY_BOOKING(1,'Approved');
select * from messages;
select * from booking_information;
```

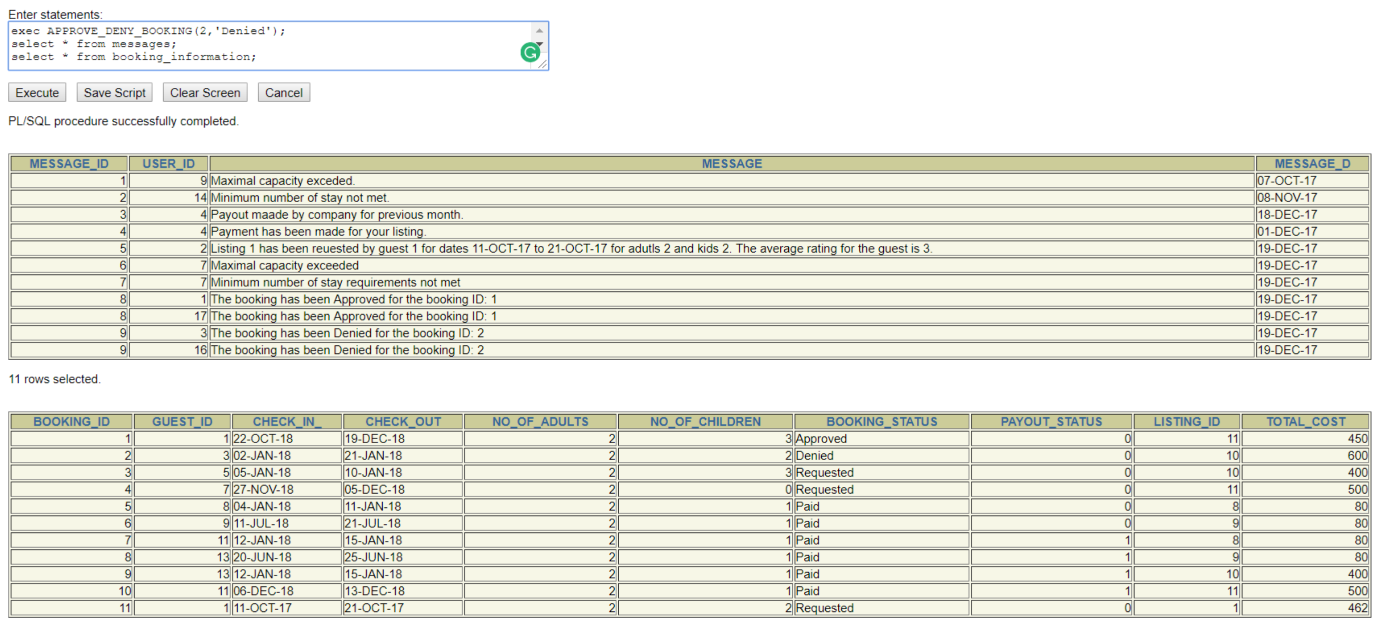
PL/SQL procedure successfully completed.

| MESSAGE_ID | USER_ID | MESSAGE | MESSAGE_D |
|---|---|---|---|
| 1 | 9 | Maximal capacity exceded. | 07-OCT-17 |
| 2 | 14 | Minimum number of stay not met. | 08-NOV-17 |
| 3 | 4 | Payout maade by company for previous month. | 18-DEC-17 |
| 4 | 4 | Payment has been made for your listing. | 01-DEC-17 |
| 5 | 2 | Listing 1 has been reuested by guest 1 for dates 11-OCT-17 to 21-OCT-17 for adutls 2 and kids 2. The average rating for the guest is 3. | 19-DEC-17 |
| 6 | 7 | Maximal capacity exceeded | 19-DEC-17 |
| 7 | 7 | Minimum number of stay requirements not met | 19-DEC-17 |
| 8 | 1 | The booking has been Approved for the booking ID: 1 | 19-DEC-17 |
| 8 | 17 | The booking has been Approved for the booking ID: 1 | 19-DEC-17 |

9 rows selected.

| BOOKING_ID | GUEST_ID | CHECK_IN_ | CHECK_OUT | NO_OF_ADULTS | NO_OF_CHILDREN | BOOKING_STATUS | PAYOUT_STATUS | LISTING_ID | TOTAL_COST |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 22-OCT-18 | 19-DEC-18 | 2 | 3 | Approved | 0 | 11 | 450 |
| 2 | 3 | 02-JAN-18 | 21-JAN-18 | 2 | 2 | Requested | 0 | 10 | 600 |
| 3 | 5 | 05-JAN-18 | 10-JAN-18 | 2 | 3 | Requested | 0 | 10 | 400 |
| 4 | 7 | 27-NOV-18 | 05-DEC-18 | 2 | 0 | Requested | 0 | 11 | 500 |
| 5 | 8 | 04-JAN-18 | 11-JAN-18 | 2 | 1 | Paid | 0 | 8 | 80 |
| 6 | 9 | 11-JUL-18 | 21-JUL-18 | 2 | 1 | Paid | 0 | 9 | 80 |
| 7 | 11 | 12-JAN-18 | 15-JAN-18 | 2 | 1 | Paid | 1 | 8 | 80 |
| 8 | 13 | 20-JUN-18 | 25-JUN-18 | 2 | 1 | Paid | 1 | 9 | 80 |
| 9 | 13 | 12-JAN-18 | 15-JAN-18 | 2 | 1 | Paid | 1 | 10 | 400 |
| 10 | 11 | 06-DEC-18 | 13-DEC-18 | 2 | 1 | Paid | 1 | 11 | 500 |
| 11 | 1 | 11-OCT-17 | 21-OCT-17 | 2 | 2 | Requested | 0 | 1 | 462 |

--booking denied



```
exec APPROVE_DENY_BOOKING(2,'Denied');
select * from messages;
select * from booking_information;
```

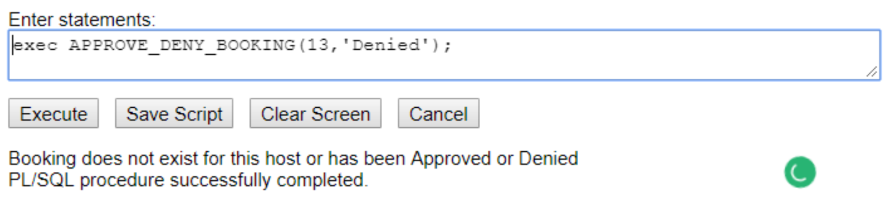
PL/SQL procedure successfully completed.

| MESSAGE_ID | USER_ID | MESSAGE | MESSAGE_D |
|---|---|---|---|
| 1 | 9 | Maximal capacity exceded. | 07-OCT-17 |
| 2 | 14 | Minimum number of stay not met. | 08-NOV-17 |
| 3 | 4 | Payout maade by company for previous month. | 18-DEC-17 |
| 4 | 4 | Payment has been made for your listing. | 01-DEC-17 |
| 5 | 2 | Listing 1 has been reuested by guest 1 for dates 11-OCT-17 to 21-OCT-17 for adutls 2 and kids 2. The average rating for the guest is 3. | 19-DEC-17 |
| 6 | 7 | Maximal capacity exceeded | 19-DEC-17 |
| 7 | 7 | Minimum number of stay requirements not met | 19-DEC-17 |
| 8 | 1 | The booking has been Approved for the booking ID: 1 | 19-DEC-17 |
| 8 | 17 | The booking has been Approved for the booking ID: 1 | 19-DEC-17 |
| 9 | 3 | The booking has been Denied for the booking ID: 2 | 19-DEC-17 |
| 9 | 16 | The booking has been Denied for the booking ID: 2 | 19-DEC-17 |

11 rows selected.

| BOOKING_ID | GUEST_ID | CHECK_IN_ | CHECK_OUT | NO_OF_ADULTS | NO_OF_CHILDREN | BOOKING_STATUS | PAYOUT_STATUS | LISTING_ID | TOTAL_COST |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 22-OCT-18 | 19-DEC-18 | 2 | 3 | Approved | 0 | 11 | 450 |
| 2 | 3 | 02-JAN-18 | 21-JAN-18 | 2 | 2 | Denied | 0 | 10 | 600 |
| 3 | 5 | 05-JAN-18 | 10-JAN-18 | 2 | 3 | Requested | 0 | 10 | 400 |
| 4 | 7 | 27-NOV-18 | 05-DEC-18 | 2 | 0 | Requested | 0 | 11 | 500 |
| 5 | 8 | 04-JAN-18 | 11-JAN-18 | 2 | 1 | Paid | 0 | 8 | 80 |
| 6 | 9 | 11-JUL-18 | 21-JUL-18 | 2 | 1 | Paid | 0 | 9 | 80 |
| 7 | 11 | 12-JAN-18 | 15-JAN-18 | 2 | 1 | Paid | 1 | 8 | 80 |
| 8 | 13 | 20-JUN-18 | 25-JUN-18 | 2 | 1 | Paid | 1 | 9 | 80 |
| 9 | 13 | 12-JAN-18 | 15-JAN-18 | 2 | 1 | Paid | 1 | 10 | 400 |
| 10 | 11 | 06-DEC-18 | 13-DEC-18 | 2 | 1 | Paid | 1 | 11 | 500 |
| 11 | 1 | 11-OCT-17 | 21-OCT-17 | 2 | 2 | Requested | 0 | 1 | 462 |

--no booking id



```
exec APPROVE_DENY_BOOKING(13,'Denied');
```

Booking does not exist for this host or has been Approved or Denied
PL/SQL procedure successfully completed.

**-- Feature 9: Look up booking request for a host**

**--- main procedure**

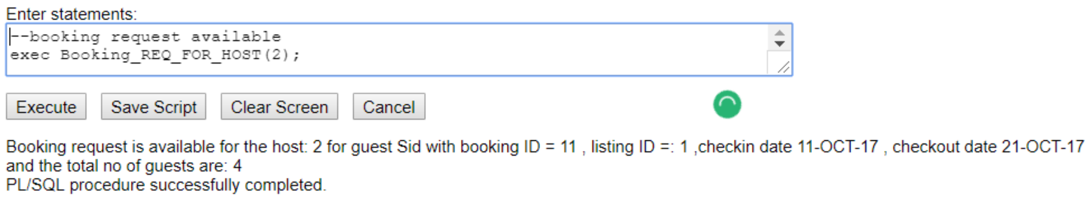
create or replace PROCEDURE Booking_REQ_FOR_HOST(hostid in int) IS

```
Cursor c1 is select b.booking_ID,s.name,b.listing_ID,b.check_in_date,b.check_out_date,(b.No_of_Adults + b.No_of_Children) from BOOKING_INFORMATION b, System_USER s, listing l where l.host_ID= hostid and l.listing_ID = b.listing_ID and

            b.guest_ID = s.user_ID and Booking_Status = 'Requested';

bookid number;

gname varchar(50);

lid number;

checkin date;

checkout date;

total number;

Check_HOST number;

check_val number;

BEGIN

check_val:=0;

Check_HOST :=CHECK_EXISTING_HOST(hostid);

        IF Check_Host =1 THEN

Open c1;

Loop

fetch c1 into bookid,gname,lid,checkin,checkout,total ;

if c1%found then

check_val:=1;

dbms_output.put_line('Booking request is available for the host: ' ||hostid|| '   for guest ' ||gname||' with booking ID = ' ||bookid|| ' , listing ID =: ' ||lid|| ' ,checkin date ' ||checkin|| ' , checkout date ' || checkout|| ' and the total no of guests are: ' ||total);

end if;


if check_val = 0 then

dbms_output.put_line('NO REQUEST FOUND');

end if;

exit when c1%notfound;


End loop;

close c1;

End if;
```

END;

**-- Execution**

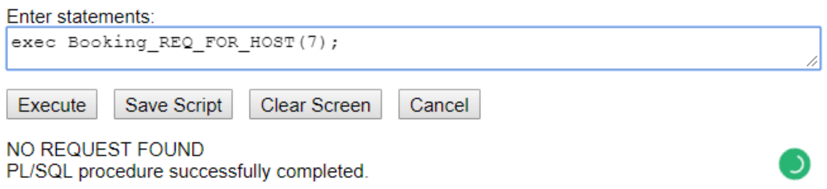
--booking request available



```
--booking request available
exec Booking_REQ_FOR_HOST(2);
```

```
Booking request is available for the host: 2 for guest Sid with booking ID = 11 , listing ID =: 1 ,checkin date 11-OCT-17 , checkout date 21-OCT-17
and the total no of guests are: 4
PL/SQL procedure successfully completed.
```

--booking request not available



```
exec Booking_REQ_FOR_HOST(7);
```

```
NO REQUEST FOUND
PL/SQL procedure successfully completed.
```

**--FEATURE 10: Allow a guest to make payment**

**-- function for checking existing booking id and booking status approved**

create or replace function CHECK_EXISTING_BOOK_A(BookingID in int)

return int

IS

Check_BookingID number;

BEGIN

Check_BookingID := 0;

select count(*) into Check_BookingID from Booking_Information where  Booking_ID = BookingID and Booking_status = 'Approved';

if Check_BookingID=0 then return Check_BookingID;

else return Check_BookingID;

end if;

End;

**--main procedure**

CREATE OR REPLACE procedure Make_Payment(BookingID in int, paymethod in varchar, pay_date in date) IS

Check_BookingID number;

check_date date;

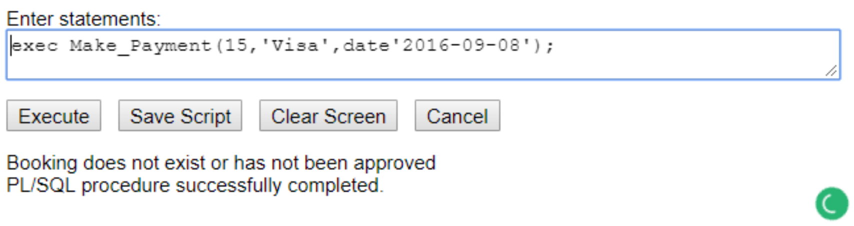
gid number;

hid number;

```
valuee number;

BEGIN

valuee:=MESSAGES_SEQ.nextval;

   Check_BookingID :=CHECK_EXISTING_BOOK_A(BookingID);

      IF Check_BookingID = 1  THEN

                              select check_in_date into check_date from booking_information where booking_ID=BookingID;

                              If check_date > pay_date then

                                       update Booking_information set booking_status='Paid' where booking_ID=BookingID;

                                       dbms_output.put_line('The payment has been made');

                                                select Guest_ID into gid from BOOKING_INFORMATION where
booking_ID=BookingID;

                                                select host_ID into hid from Listing l,BOOKING_INFORMATION b where
b.booking_ID=BookingID and l.listing_ID = b.listing_ID;

                                                Insert into Messages values (valuee, gid,'The payment has been made for booking
ID '||BookingID|| ' using ' ||paymethod|| ' as payment method',sysdate);

                                                Insert into Messages values (valuee, hid,'The payment has been made for booking
ID '||BookingID|| ' using ' ||paymethod|| ' as payment method',sysdate);

                              ELSE

                                       dbms_output.put_line('Please Enter a valid date');

                              end if;

                     ELSE

                              dbms_output.put_line('Booking does not exist or has not been approved');

   end if;

END;
```

**-Execution**

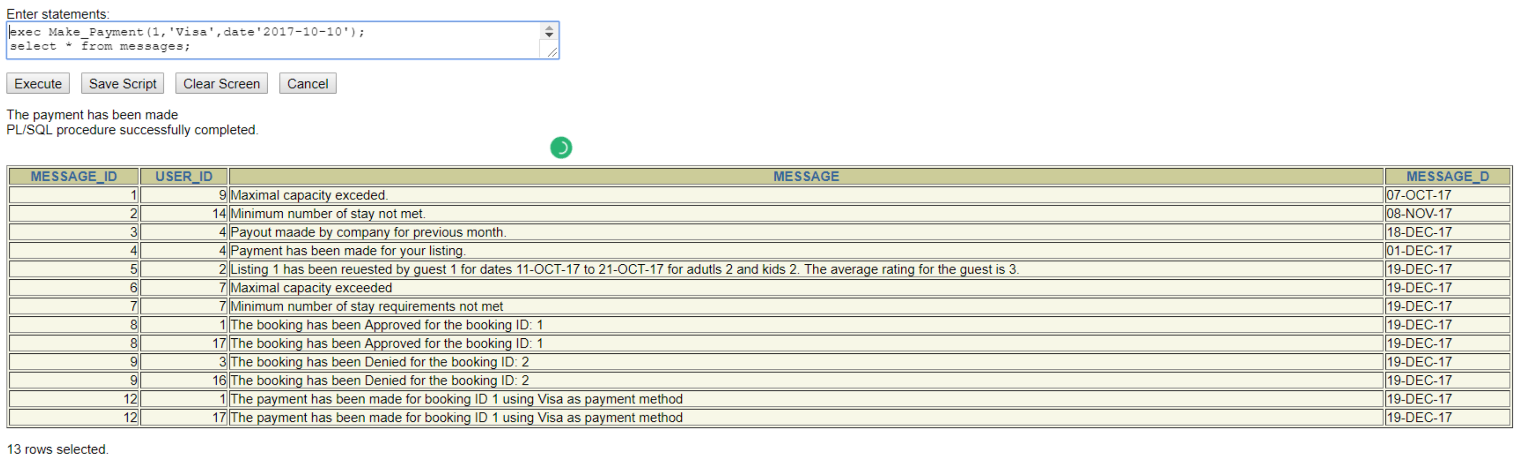
--Booking does not exist or has not been approved


Enter statements:

```
exec Make_Payment(15,'Visa',date'2016-09-08');
```

```
Booking does not exist or has not been approved
PL/SQL procedure successfully completed.
```

--The payment has been made


Enter statements:

```
exec Make_Payment(1,'Visa',date'2017-10-10');
select * from messages;
```

```
The payment has been made
PL/SQL procedure successfully completed.
```

| MESSAGE_ID | USER_ID | MESSAGE | MESSAGE_D |
|---|---|---|---|
| 1 | 9 | Maximal capacity exceded. | 07-OCT-17 |
| 2 | 14 | Minimum number of stay not met. | 08-NOV-17 |
| 3 | 4 | Payout maade by company for previous month. | 18-DEC-17 |
| 4 | 4 | Payment has been made for your listing. | 01-DEC-17 |
| 5 | 2 | Listing 1 has been reuested by guest 1 for dates 11-OCT-17 to 21-OCT-17 for adutls 2 and kids 2. The average rating for the guest is 3. | 19-DEC-17 |
| 6 | 7 | Maximal capacity exceeded | 19-DEC-17 |
| 7 | 7 | Minimum number of stay requirements not met | 19-DEC-17 |
| 8 | 1 | The booking has been Approved for the booking ID: 1 | 19-DEC-17 |
| 8 | 17 | The booking has been Approved for the booking ID: 1 | 19-DEC-17 |
| 9 | 3 | The booking has been Denied for the booking ID: 2 | 19-DEC-17 |
| 9 | 16 | The booking has been Denied for the booking ID: 2 | 19-DEC-17 |
| 12 | 1 | The payment has been made for booking ID 1 using Visa as payment method | 19-DEC-17 |
| 12 | 17 | The payment has been made for booking ID 1 using Visa as payment method | 19-DEC-17 |

13 rows selected.

**--Feature 11: Allow a guest to cancel a booking if not paid yet**

**--Function to check existing booking**

create or replace function CHECK_EXISTING_BOOKING_I(BookingID in int)

return int

IS

Check_BookingID number;

BEGIN

Check_BookingID := 0;

    select count(*) into Check_BookingID from Booking_Information where  Booking_ID = BookingID;

if Check_BookingID=0 then return Check_BookingID;

else return Check_BookingID;

end if;

End;

**--Main procedure**

CREATE OR REPLACE procedure cancel_Booking(BookingID in int) IS

Check_BookingID number;

countvalue number;

valuee number;

gid number;

hid number;

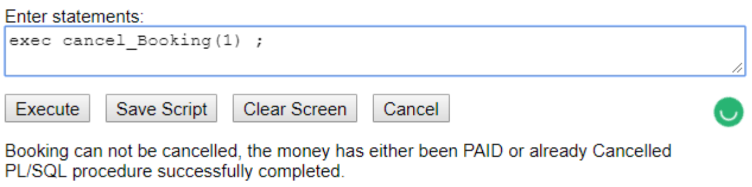
BEGIN

valuee:=MESSAGES_SEQ.nextval;

```
Check_BookingID :=CHECK_EXISTING_BOOKING_I(BookingID);
        IF Check_BookingID = 1 THEN


                            select count(*) into countvalue from booking_information where booking_id=BookingID and
(booking_status='Paid' or booking_status='Canceled');


                                                if countvalue=0 then



                                           update Booking_information set Booking_Status = 'Canceled'  where
booking_ID=BookingID and (Booking_Status!='Paid' or Booking_status='Canceled');
                                                   dbms_output.put_line('The Booking has been cancelled by the Guest');
                                           select Guest_ID into gid from BOOKING_INFORMATION where
booking_ID=BookingID;
                   select host_ID into hid from Listing l,BOOKING_INFORMATION b where b.booking_ID=BookingID
and l.listing_ID = b.listing_ID;
                   Insert into Messages values (valuee, gid,'The booking has been Cancelled by the Guest',sysdate);
                   Insert into Messages values (valuee, hid,'The booking has been Cancelled by the guest',sysdate);


                                elsif countvalue=1 then
                   dbms_output.put_line('Booking can not be cancelled, the money has either been PAID or already
Cancelled' );
                                                   select Guest_ID into gid from BOOKING_INFORMATION where
booking_ID=BookingID;
                                                   Insert into Messages values (valuee, gid,'Booking can not be cancelled, the money has
either been PAID or already Cancelled',sysdate);
                 end if;
                        Elsif check_bookingid=0 then
                    dbms_output.put_line('Booking ID does not exist for any Guest');
        end if;
END;
```

**-- Execution**

--paid

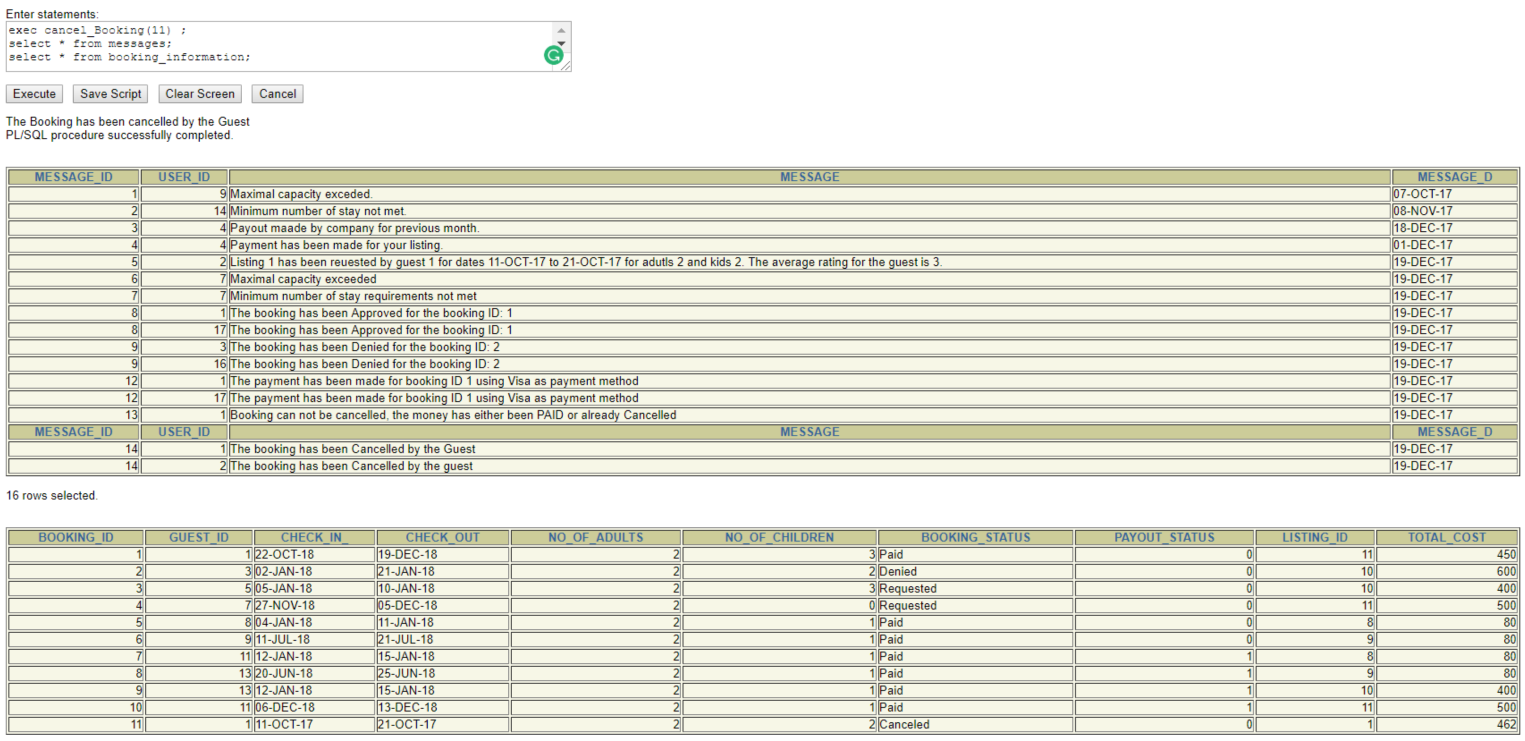


Enter statements:

```
exec cancel_Booking(1) ;
```

```
Booking can not be cancelled, the money has either been PAID or already Cancelled
PL/SQL procedure successfully completed.
```

--requested



Enter statements:

```
exec cancel_Booking(11) ;
select * from messages;
select * from booking_information;
```

```
The Booking has been cancelled by the Guest
PL/SQL procedure successfully completed.
```

| MESSAGE_ID | USER_ID | MESSAGE | MESSAGE_D |
|---|---|---|---|
| 1 | 9 | Maximal capacity exceded. | 07-OCT-17 |
| 2 | 14 | Minimum number of stay not met. | 08-NOV-17 |
| 3 | 4 | Payout maade by company for previous month. | 18-DEC-17 |
| 4 | 4 | Payment has been made for your listing. | 01-DEC-17 |
| 5 | 2 | Listing 1 has been reuested by guest 1 for dates 11-OCT-17 to 21-OCT-17 for adutls 2 and kids 2. The average rating for the guest is 3. | 19-DEC-17 |
| 6 | 7 | Maximal capacity exceeded | 19-DEC-17 |
| 7 | 7 | Minimum number of stay requirements not met | 19-DEC-17 |
| 8 | 1 | The booking has been Approved for the booking ID: 1 | 19-DEC-17 |
| 8 | 17 | The booking has been Approved for the booking ID: 1 | 19-DEC-17 |
| 9 | 3 | The booking has been Denied for the booking ID: 2 | 19-DEC-17 |
| 9 | 16 | The booking has been Denied for the booking ID: 2 | 19-DEC-17 |
| 12 | 1 | The payment has been made for booking ID 1 using Visa as payment method | 19-DEC-17 |
| 12 | 17 | The payment has been made for booking ID 1 using Visa as payment method | 19-DEC-17 |
| 13 | 1 | Booking can not be cancelled, the money has either been PAID or already Cancelled | 19-DEC-17 |
| MESSAGE_ID | USER_ID | MESSAGE | MESSAGE_D |
| 14 | 1 | The booking has been Cancelled by the Guest | 19-DEC-17 |
| 14 | 2 | The booking has been Cancelled by the guest | 19-DEC-17 |

16 rows selected.

| BOOKING_ID | GUEST_ID | CHECK_IN_ | CHECK_OUT | NO_OF_ADULTS | NO_OF_CHILDREN | BOOKING_STATUS | PAYOUT_STATUS | LISTING_ID | TOTAL_COST |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 22-OCT-18 | 19-DEC-18 | 2 | 3 | Paid | 0 | 11 | 450 |
| 2 | 3 | 02-JAN-18 | 21-JAN-18 | 2 | 2 | Denied | 0 | 10 | 600 |
| 3 | 5 | 05-JAN-18 | 10-JAN-18 | 2 | 3 | Requested | 0 | 10 | 400 |
| 4 | 7 | 27-NOV-18 | 05-DEC-18 | 2 | 0 | Requested | 0 | 11 | 500 |
| 5 | 8 | 04-JAN-18 | 11-JAN-18 | 2 | 1 | Paid | 0 | 8 | 80 |
| 6 | 9 | 11-JUL-18 | 21-JUL-18 | 2 | 1 | Paid | 0 | 9 | 80 |
| 7 | 11 | 12-JAN-18 | 15-JAN-18 | 2 | 1 | Paid | 1 | 8 | 80 |
| 8 | 13 | 20-JUN-18 | 25-JUN-18 | 2 | 1 | Paid | 1 | 9 | 80 |
| 9 | 13 | 12-JAN-18 | 15-JAN-18 | 2 | 1 | Paid | 1 | 10 | 400 |
| 10 | 11 | 06-DEC-18 | 13-DEC-18 | 2 | 1 | Paid | 1 | 11 | 500 |
| 11 | 1 | 11-OCT-17 | 21-OCT-17 | 2 | 2 | Canceled | 0 | 1 | 462 |

**--Feature 12: Allow the system to generate payout to host**

**-----function to compute total payment for host**

```
create or replace function COMPUTE_PAYMENT_HOST(HostID int)

return int

IS

cursor c1 is select b.booking_id,b.total_cost,b.Listing_ID,l.host_id from booking_information b,listing l where l.host_id=HostID and l.listing_ID=b.listing_ID and b.booking_status='Paid' and b.payout_status=0;

booking number;

listing number;

total number;

data c1%rowtype;

BEGIN

total:=0;
```

```
open c1;
loop
   fetch c1 into data;
   exit when c1%notfound;
      update booking_information set payout_status=1;
 --    dbms_output.put_line(data.total_cost);
      total:=total+data.total_cost;



end loop;
close c1;
return total;
End;
```

**-----procedure to generate payout to host**

```
show errors;
CREATE OR REPLACE procedure GENERATE_PAYOUT(HostID int, Payout date) IS
compute number;
total_payment int;
service_fee float;
check_host number;
begin
check_host:=check_existing_host(HostID);
if  check_host>0 then
compute:=COMPUTE_PAYMENT_HOST(HostID);
--dbms_output.put_line('Cost without service tax deduction: '||compute);
service_fee:=(1.05*0.97);
---dbms_output.put_line('Service fee: '||service_fee);

total_payment:=compute/service_fee;

if(total_payment!=0) then
   dbms_output.put_line('Total payment with service tax deducted: '||total_payment||' dispatched on: '||Payout);
```

Insert into Payout values (PAYOUT_SEQ.nextval,HostID,total_payment,Payout);

Insert into Messages values (MESSAGES_SEQ.nextval, HostID,'Amount dispatched from company for the month on '||Payout,Payout);

else

dbms_output.put_line('Payout already done for this Host');

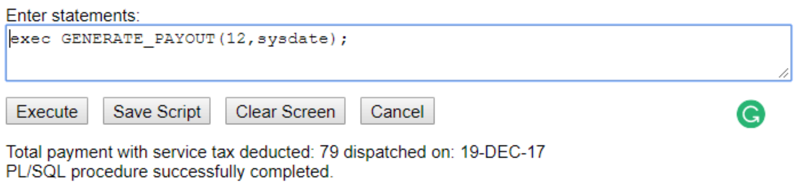
end if;

end if;

end;

**--Execution**

--payout is made

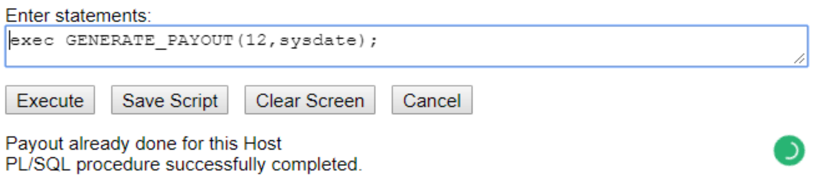


Enter statements:

```
exec GENERATE_PAYOUT(12,sysdate);
```

```
Total payment with service tax deducted: 79 dispatched on: 19-DEC-17
PL/SQL procedure successfully completed.
```

--Payout already done for this Host



Enter statements:

```
exec GENERATE_PAYOUT(12,sysdate);
```

```
Payout already done for this Host
PL/SQL procedure successfully completed.
```

**-- Feature 13 : Allow a guest to enter a review for host and update average rating for the host as well.**

**--Function to compute avg rating for host**

create or replace function COMPUTE_AVG_RATING(ID_HOST in int,ID_GUEST in int)

return int

IS

init_avg float;

new_avg float;

star float;

BEGIN

```
new_avg := 0;


select average_rating into init_avg from Host h, Review r where h.host_ID = r.host_ID and r.guest_ID=ID_GUEST
and r.flag = 1   and r.host_id=id_host;

if init_avg = 0 then

        select stars into new_avg from review where host_ID = ID_HOST and guest_id=id_guest and flag = 1;

        dbms_output.put_line('This is  the first rating= '||  new_avg);

        update host set average_rating=new_avg where host_ID = ID_HOST;

        return new_avg;


     else

        select stars into star from review where host_ID = ID_HOST and guest_id=id_guest and flag = 1;


        new_avg := (init_avg + star) /2 ;

        dbms_output.put_line('The rating='||  new_avg);

        update host set average_rating=new_avg where host_ID = ID_HOST;

        return new_avg;

     end if;

End;
```

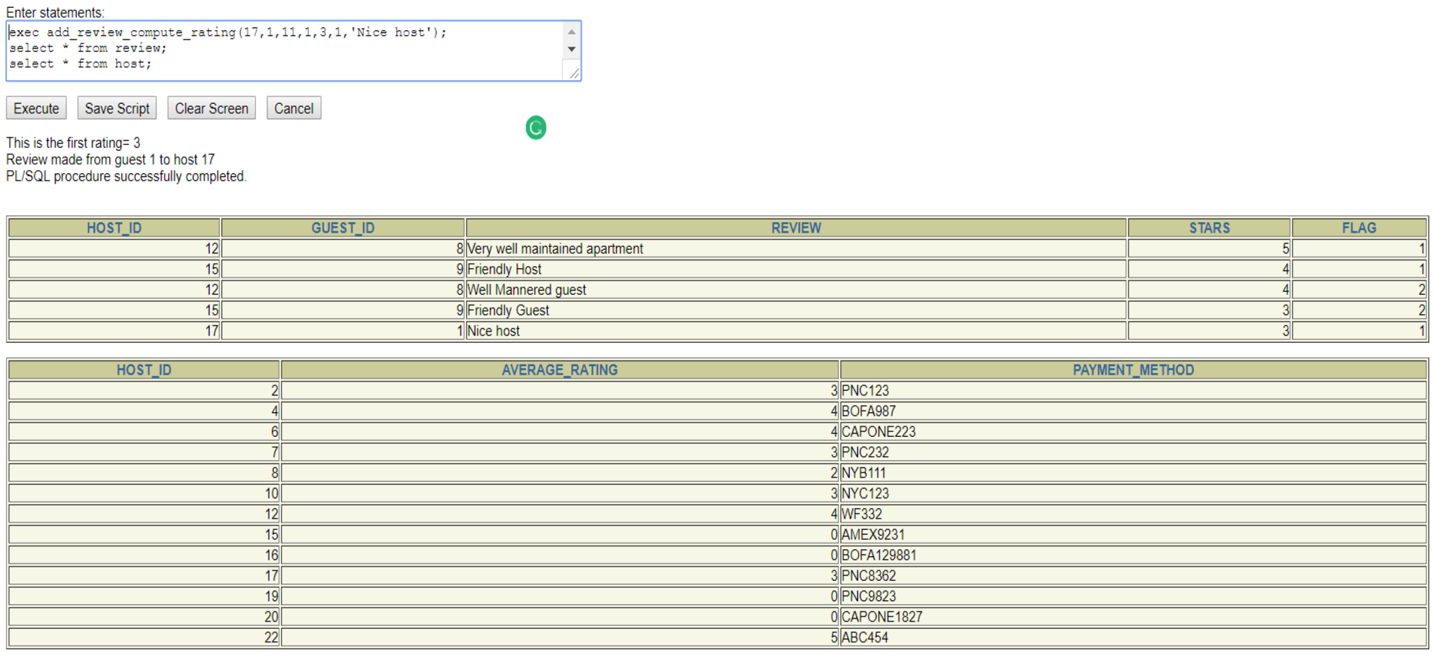
**--Main Procedure**

```
CREATE OR REPLACE procedure add_review_compute_rating(HostID in int,GuestID in int,ListingID in
int,BookingID in int,str in float,flagg in int, revieww in varchar)IS

check_host number;

check_listingID number;

check_guest number;

check_rating float;

bs number;

valuee number;

check_booking number;

Begin

check_host := CHECK_EXISTING_HOST(HostID);

check_guest:=CHECK_EXISTING_GUEST(GuestID);
```

```
if check_host=0  then dbms_output.put_line('No HOST found'); end if;

if check_GUEST=0  then dbms_output.put_line('No GUEST found'); end if;

if flagg=2 then dbms_output.put_line('Review cannot be entered'); end if;


IF check_host = 1 and check_guest=1 and flagg=1 THEN

            check_listingID :=CHECK_EXISTING_LISTING_ID(ListingID);

               if check_listingID= 0 then dbms_output.put_line('No Listing found');

               else

                                   check_booking:=CHECK_EXISTING_BOOKING_ID(BookingID);

                        if check_booking= 0 then

                           dbms_output.put_line('No Booking found');


                                          elsif check_booking=1 then

                          select count(*) into bs from booking_information where Guest_ID=GuestID and
Booking_ID=BookingID and booking_status='Paid';

                          if bs=0 then

                            dbms_output.put_line('Booking either canceled or Unpaid , Review can''t be given');

                          elsif bs>0 then

                            select count(*) into valuee from review where host_id=HostID and Guest_id=guestID  and
flag=flagg;

                                 if valuee=0 then

                                     insert into review values(HostID,GuestID,revieww ,str,flagg);

                                     check_rating:=COMPUTE_AVG_RATING(HostID,GuestID);

                                                                                                            dbms_output.put_line('Review made from guest
'||GuestID|| ' to host ' ||hostID);

                                 else

                                 dbms_output.put_line('Review already made');

                                                                                                  end if;


                          end if;

                        end if;

                end if;

end if;

end;
```

## --Execution

--review made from guest to host first time



Enter statements:

```
exec add_review_compute_rating(17,1,11,1,3,1,'Nice host');
select * from review;
select * from host;
```

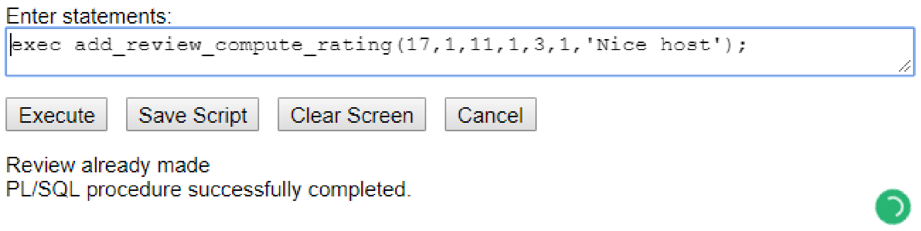
Execute | Save Script | Clear Screen | Cancel

```
This is the first rating= 3
Review made from guest 1 to host 17
PL/SQL procedure successfully completed.
```

| HOST_ID | GUEST_ID | REVIEW | STARS | FLAG |
|---|---|---|---|---|
| 12 | 8 | Very well maintained apartment | 5 | 1 |
| 15 | 9 | Friendly Host | 4 | 1 |
| 12 | 8 | Well Mannered guest | 4 | 2 |
| 15 | 9 | Friendly Guest | 3 | 2 |
| 17 | 1 | Nice host | 3 | 1 |

| HOST_ID | AVERAGE_RATING | PAYMENT_METHOD |
|---|---|---|
| 2 | 3 | PNC123 |
| 4 | 4 | BOFA987 |
| 6 | 4 | CAPONE223 |
| 7 | 3 | PNC232 |
| 8 | 2 | NYB111 |
| 10 | 3 | NYC123 |
| 12 | 4 | WF332 |
| 15 | 0 | AMEX9231 |
| 16 | 0 | BOFA129881 |
| 17 | 3 | PNC8362 |
| 19 | 0 | PNC9823 |
| 20 | 0 | CAPONE1827 |
| 22 | 5 | ABC454 |

--review already made from guest to host



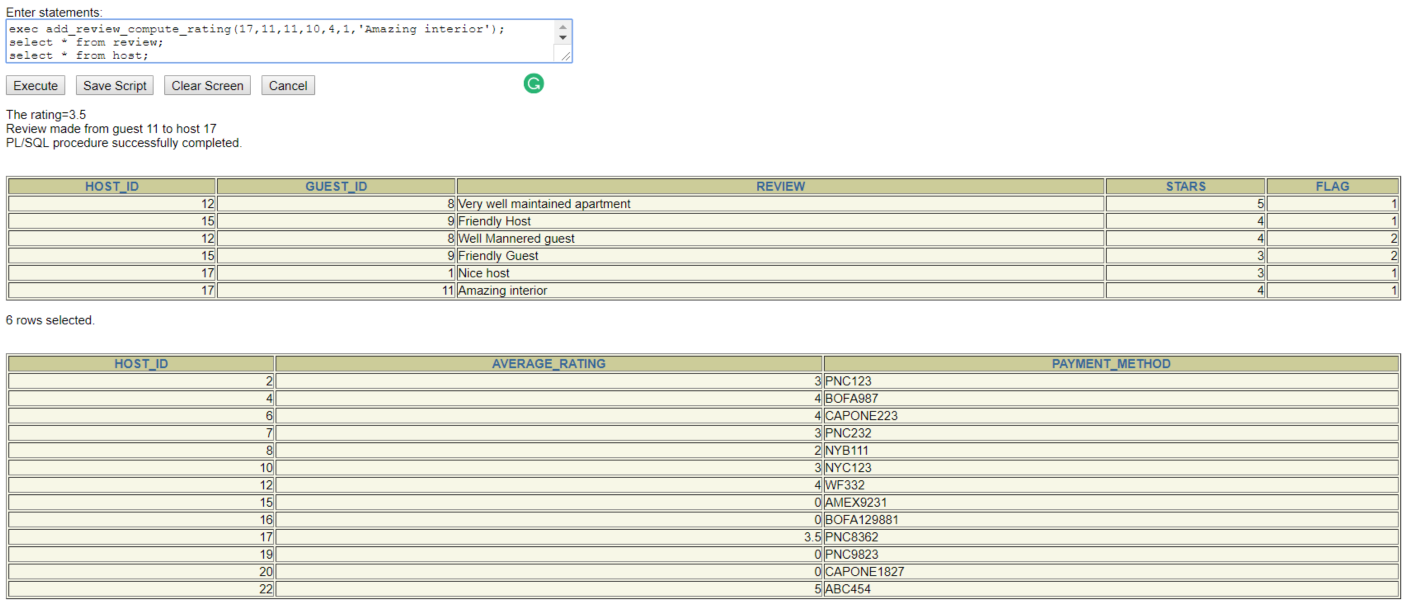
Enter statements:

```
exec add_review_compute_rating(17,1,11,1,3,1,'Nice host');
```

Execute | Save Script | Clear Screen | Cancel

```
Review already made
PL/SQL procedure successfully completed.
```

--review made from guest to host not for the 1st time



Enter statements:

```
exec add_review_compute_rating(17,11,11,10,4,1,'Amazing interior');
select * from review;
select * from host;
```

Execute | Save Script | Clear Screen | Cancel

```
The rating=3.5
Review made from guest 11 to host 17
PL/SQL procedure successfully completed.
```

| HOST_ID | GUEST_ID | REVIEW | STARS | FLAG |
|---|---|---|---|---|
| 12 | 8 | Very well maintained apartment | 5 | 1 |
| 15 | 9 | Friendly Host | 4 | 1 |
| 12 | 8 | Well Mannered guest | 4 | 2 |
| 15 | 9 | Friendly Guest | 3 | 2 |
| 17 | 1 | Nice host | 3 | 1 |
| 17 | 11 | Amazing interior | 4 | 1 |

6 rows selected.

| HOST_ID | AVERAGE_RATING | PAYMENT_METHOD |
|---|---|---|
| 2 | 3 | PNC123 |
| 4 | 4 | BOFA987 |
| 6 | 4 | CAPONE223 |
| 7 | 3 | PNC232 |
| 8 | 2 | NYB111 |
| 10 | 3 | NYC123 |
| 12 | 4 | WF332 |
| 15 | 0 | AMEX9231 |
| 16 | 0 | BOFA129881 |
| 17 | 3.5 | PNC8362 |
| 19 | 0 | PNC9823 |
| 20 | 0 | CAPONE1827 |
| 22 | 5 | ABC454 |

**--Feature 14: Allow a host to enter a review to guest and update average rating for the guest as well.**

**--Function to compute avg rating for guest**

```
create or replace function COMPUTE_AVG_RATING_G(HostID in int,GuestID in int)
return int
IS
init_avg float;
new_avg float;
star float;
BEGIN
new_avg := 0;

select average_rating into init_avg from guest g, Review r where g.guest_ID = r.guest_ID and r.host_ID=HostID and
r.flag = 2  and r.guest_id=GuestID;
if init_avg = 0 then
      select stars into new_avg from review where guest_ID = GuestID and host_id=HostID and flag = 2;
      dbms_output.put_line('This is  the first rating= '||  new_avg);
      update guest set average_rating=new_avg where guest_ID = GuestID;
      return new_avg;

    else
      select stars into star from review where guest_ID = GuestID and host_id=HostID and flag = 2;

      new_avg := (init_avg + star) /2 ;
      --dbms_output.put_line('The rating='||  new_avg);
      update guest set average_rating=new_avg where guest_ID = GuestID;
      return new_avg;
    end if;

            End;
```

**--Main procedure**

```
CREATE OR REPLACE procedure add_review_compute_rating_g(HostID in int,GuestID in int,ListingID in
int,BookingID in int,str in float,flagg in int, revieww in varchar)IS

check_host number;

check_listingID number;

check_guest number;

check_rating float;

bs number;

valuee number;

check_booking number;

Begin


check_guest:=CHECK_EXISTING_GUEST(GuestID);

check_host := CHECK_EXISTING_HOST(HostID);

if check_host=0  then dbms_output.put_line('No HOST found'); end if;

if check_GUEST=0  then dbms_output.put_line('No GUEST found'); end if;

if flagg=1 then dbms_output.put_line('Review cannot be entered'); end if;


IF check_host = 1 and check_guest=1 and flagg=2 THEN

        check_listingID :=CHECK_EXISTING_LISTING_ID(ListingID);

          if check_listingID= 0 then dbms_output.put_line('No Listing found');

          else

                        check_booking:=CHECK_EXISTING_BOOKING_ID(BookingID);

                if check_booking= 0 then

                  dbms_output.put_line('No Booking found');


                            elsif check_booking=1 then

                 select count(*) into bs from booking_information where guest_ID=guestID and
Booking_ID=BookingID and booking_status='Paid';

                 if bs=0 then

                   dbms_output.put_line('Booking either canceled or Unpaid , Review can''t be given');

                 elsif bs>0 then

                  select count(*) into valuee from review where host_id=HostID and Guest_id=guestID  and
flag=flagg;

                      if valuee=0 then
```

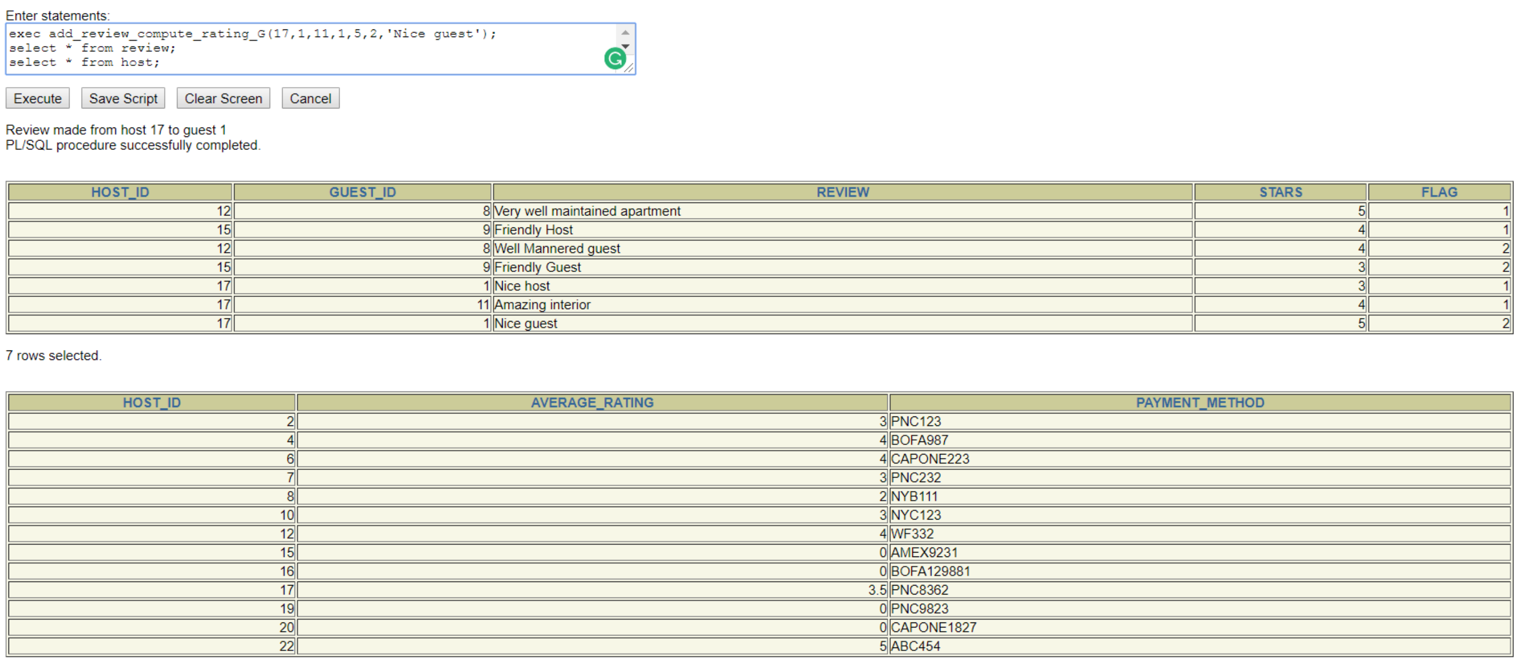
```
                    insert into review values(HostID,GuestID,revieww ,str,flagg);
                                                                  dbms_output.put_line('Review made from host
'||hostID|| ' to guest ' ||GuestID);
                    check_rating:=COMPUTE_AVG_RATING_G(HostID,GuestID);
                  else
                  dbms_output.put_line('Review already made');
                                                            end if;


               end if;
              end if;
          end if;
end if;
end;
```

**--Execution**

--review made for guest

--review made from host to guest first time

set serveroutput on;



Enter statements:

```
exec add_review_compute_rating_G(17,1,11,1,5,2,'Nice guest');
select * from review;
select * from host;
```

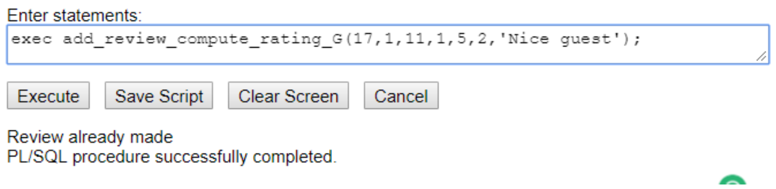
```
Review made from host 17 to guest 1
PL/SQL procedure successfully completed.
```

| HOST_ID | GUEST_ID | REVIEW | STARS | FLAG |
|---|---|---|---|---|
| 12 | 8 | Very well maintained apartment | 5 | 1 |
| 15 | 9 | Friendly Host | 4 | 1 |
| 12 | 8 | Well Mannered guest | 4 | 2 |
| 15 | 9 | Friendly Guest | 3 | 2 |
| 17 | 1 | Nice host | 3 | 1 |
| 17 | 11 | Amazing interior | 4 | 1 |
| 17 | 1 | Nice guest | 5 | 2 |

7 rows selected.

| HOST_ID | AVERAGE_RATING | PAYMENT_METHOD |
|---|---|---|
| 2 | 3 | PNC123 |
| 4 | 4 | BOFA987 |
| 6 | 4 | CAPONE223 |
| 7 | 3 | PNC232 |
| 8 | 2 | NYB111 |
| 10 | 3 | NYC123 |
| 12 | 4 | WF332 |
| 15 | 0 | AMEX9231 |
| 16 | 0 | BOFA129881 |
| 17 | 3.5 | PNC8362 |
| 19 | 0 | PNC9823 |
| 20 | 0 | CAPONE1827 |
| 22 | 5 | ABC454 |

--review already made

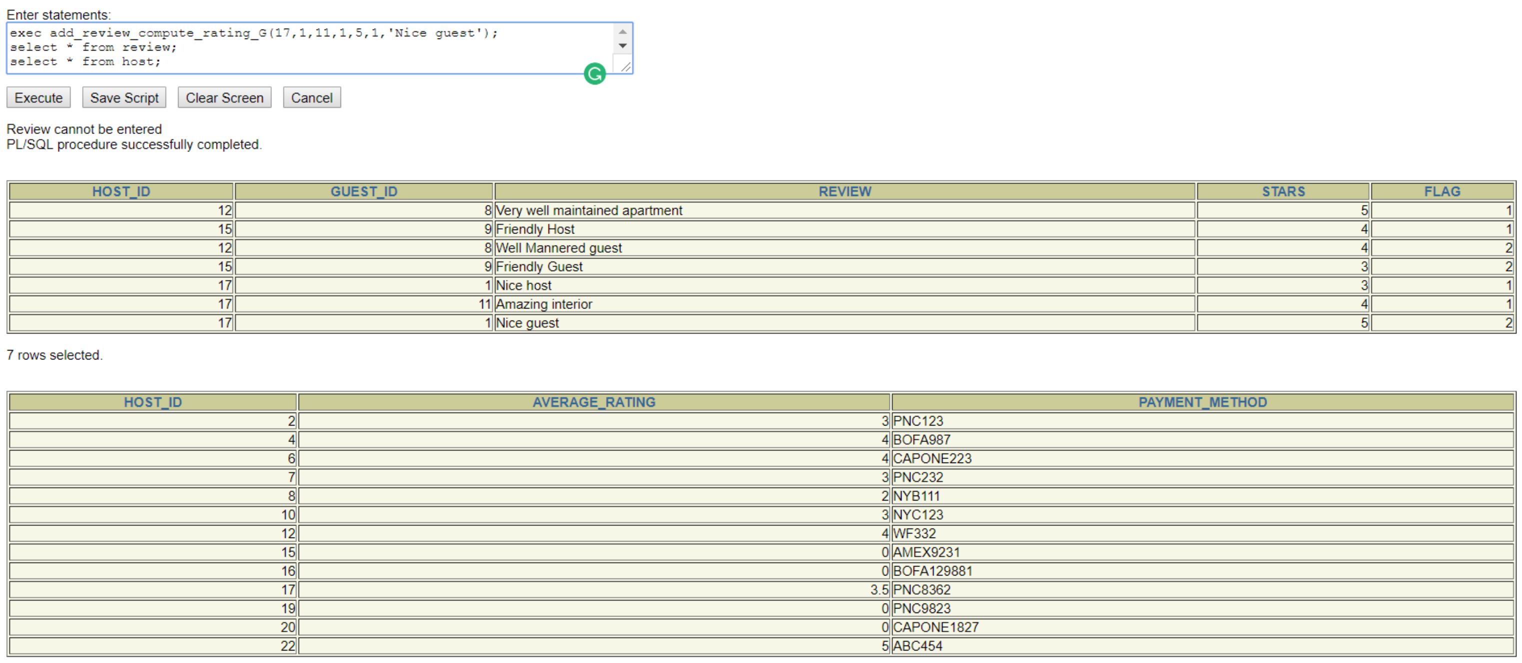


Enter statements:

```
exec add_review_compute_rating_G(17,1,11,1,5,2,'Nice guest');
```

```
Review already made
PL/SQL procedure successfully completed.
```

--review cant be entered


Enter statements:

```
exec add_review_compute_rating_G(17,1,11,1,5,1,'Nice guest');
select * from review;
select * from host;
```

```
Review cannot be entered
PL/SQL procedure successfully completed.
```

| HOST_ID | GUEST_ID | REVIEW | STARS | FLAG |
|---|---|---|---|---|
| 12 | 8 | Very well maintained apartment | 5 | 1 |
| 15 | 9 | Friendly Host | 4 | 1 |
| 12 | 8 | Well Mannered guest | 4 | 2 |
| 15 | 9 | Friendly Guest | 3 | 2 |
| 17 | 1 | Nice host | 3 | 1 |
| 17 | 11 | Amazing interior | 4 | 1 |
| 17 | 1 | Nice guest | 5 | 2 |

7 rows selected.

| HOST_ID | AVERAGE_RATING | PAYMENT_METHOD |
|---|---|---|
| 2 | 3 | PNC123 |
| 4 | 4 | BOFA987 |
| 6 | 4 | CAPONE223 |
| 7 | 3 | PNC232 |
| 8 | 2 | NYB111 |
| 10 | 3 | NYC123 |
| 12 | 4 | WF332 |
| 15 | 0 | AMEX9231 |
| 16 | 0 | BOFA129881 |
| 17 | 3.5 | PNC8362 |
| 19 | 0 | PNC9823 |
| 20 | 0 | CAPONE1827 |
| 22 | 5 | ABC454 |

**----feature 15: Report the following statistics**

create or replace procedure COMPUTE_STATS

IS

Total_number_of_users number;

Total_number_of_guests number;

Total_number_of_hosts number;

Total_number_of_bookings number;

Total_number_of_listings number;

average_length_per_booking number;

average_cost_per_booking float;

BEGIN

select count(*) into Total_number_of_users from system_user;

dbms_output.put_line('Total number of users are: ' || Total_number_of_users);

select count(*) into Total_number_of_guests from guest;

dbms_output.put_line('Total number of guests are: ' || Total_number_of_guests);

select count(*) into Total_number_of_hosts from host;

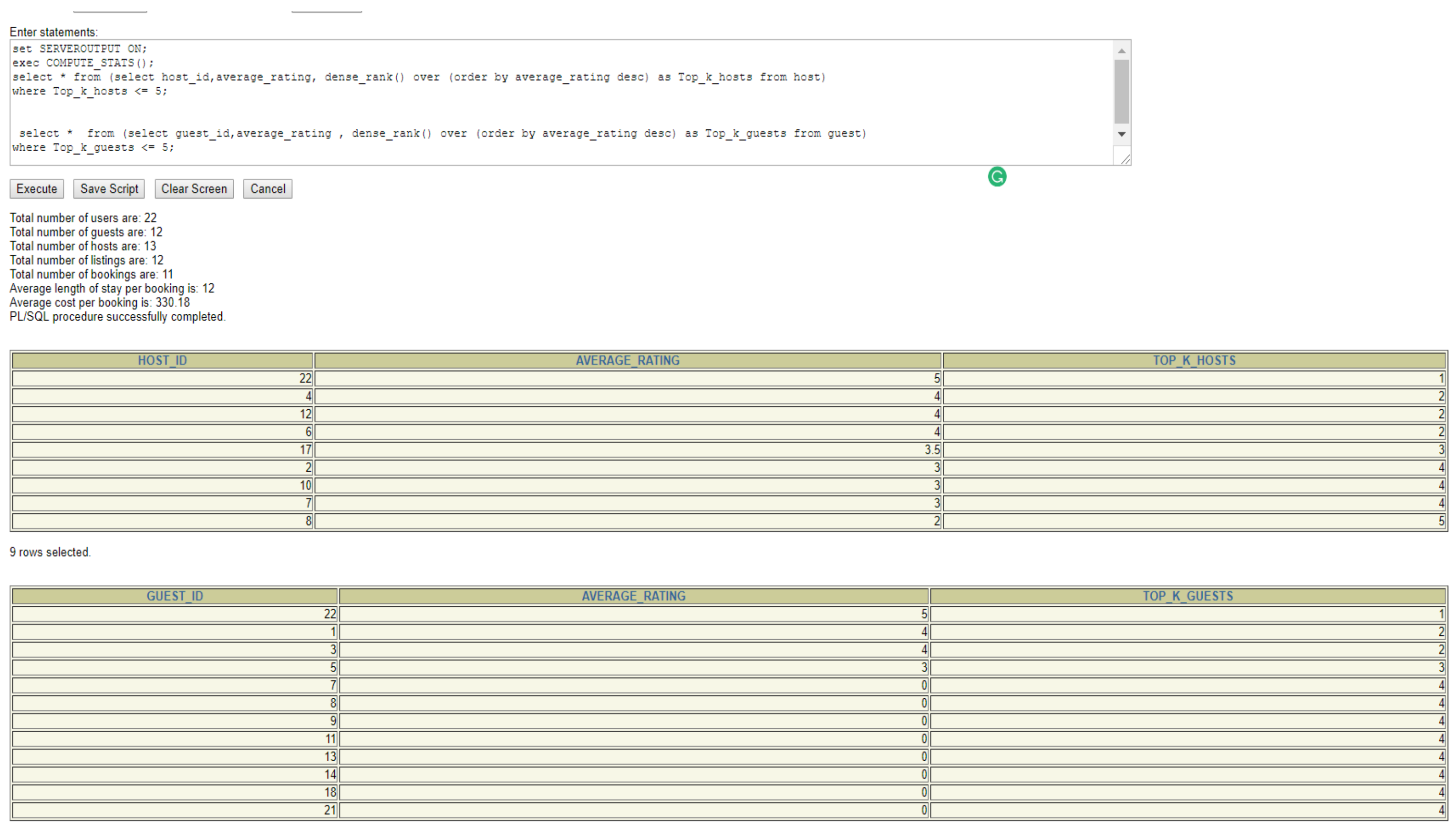
dbms_output.put_line('Total number of hosts are: ' || Total_number_of_hosts);

select count(*) into Total_number_of_listings from listing;

dbms_output.put_line('Total number of listings are: ' || Total_number_of_listings);

select count(*) into Total_number_of_bookings from booking_information;

```
dbms_output.put_line('Total number of bookings are: ' || Total_number_of_bookings);

select round(avg(check_out_date-check_in_date))into average_length_per_booking from booking_information ;

dbms_output.put_line('Average length of stay per booking is: ' || average_length_per_booking);

select round((avg(total_cost)),2)into average_cost_per_booking from booking_information ;

dbms_output.put_line('Average cost per booking is: ' || average_cost_per_booking);

End;


set SERVEROUTPUT ON;

exec COMPUTE_STATS();

select * from (select host_id,average_rating, dense_rank() over (order by average_rating desc) as Top_k_hosts from host)

where Top_k_hosts <= 5;



 select *  from (select guest_id,average_rating , dense_rank() over (order by average_rating desc) as Top_k_guests from guest)

where Top_k_guests <= 5;
```



Enter statements:

```
set SERVEROUTPUT ON;
exec COMPUTE_STATS();
select * from (select host_id,average_rating, dense_rank() over (order by average_rating desc) as Top_k_hosts from host)
where Top_k_hosts <= 5;


 select *  from (select guest_id,average_rating , dense_rank() over (order by average_rating desc) as Top_k_guests from guest)
where Top_k_guests <= 5;
```

```
Total number of users are: 22
Total number of guests are: 12
Total number of hosts are: 13
Total number of listings are: 12
Total number of bookings are: 11
Average length of stay per booking is: 12
Average cost per booking is: 330.18
PL/SQL procedure successfully completed.
```

| HOST_ID | AVERAGE_RATING | TOP_K_HOSTS |
|---|---|---|
| 22 | 5 | 1 |
| 4 | 4 | 2 |
| 12 | 4 | 2 |
| 6 | 4 | 2 |
| 17 | 3.5 | 3 |
| 2 | 3 | 4 |
| 10 | 3 | 4 |
| 7 | 3 | 4 |
| 8 | 2 | 5 |

9 rows selected.

| GUEST_ID | AVERAGE_RATING | TOP_K_GUESTS |
|---|---|---|
| 22 | 5 | 1 |
| 1 | 4 | 2 |
| 3 | 4 | 2 |
| 5 | 3 | 3 |
| 7 | 0 | 4 |
| 8 | 0 | 4 |
| 9 | 0 | 4 |
| 11 | 0 | 4 |
| 13 | 0 | 4 |
| 14 | 0 | 4 |
| 18 | 0 | 4 |
| 21 | 0 | 4 |