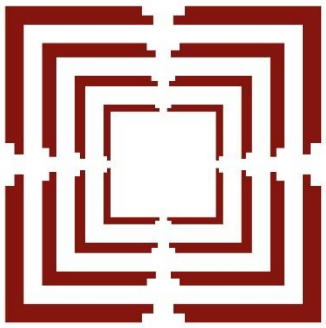


Ahmedabad University

DBMS Project Report

Project: Hospital Management System

Group Members:

Name	Enrolment Number
Anshul Mehta	AU1940275
Harsh Patel	AU1940114
Nihal Aggarwal	AU1940217

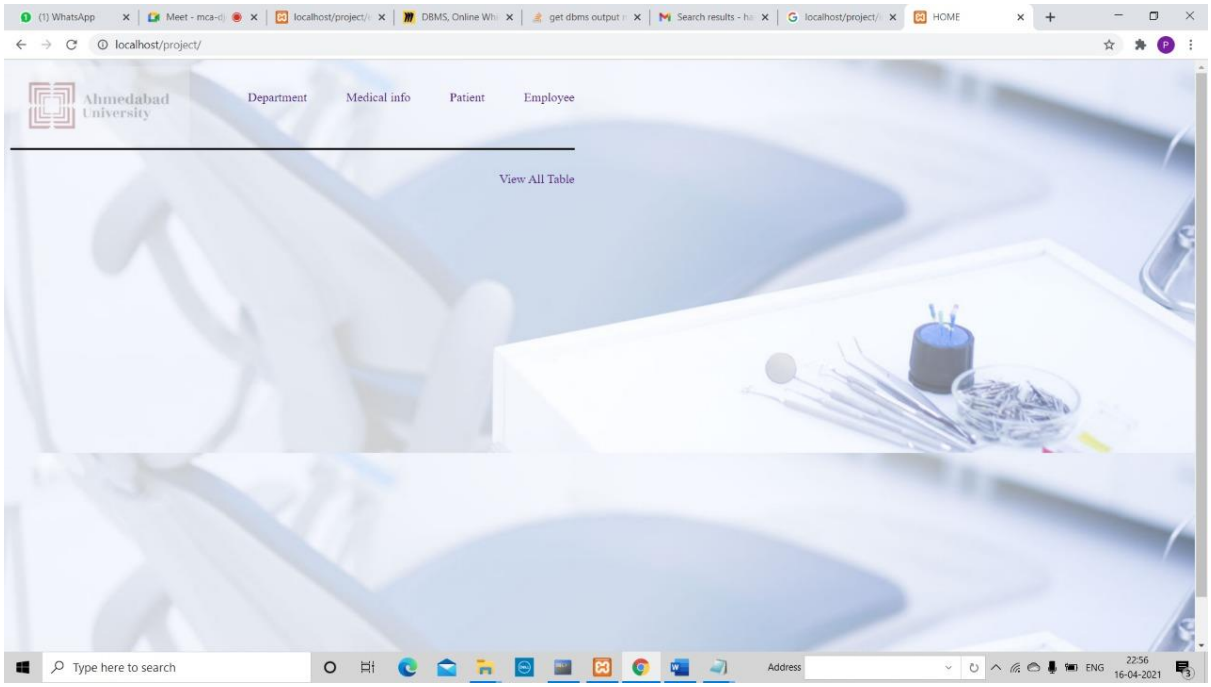
Description:

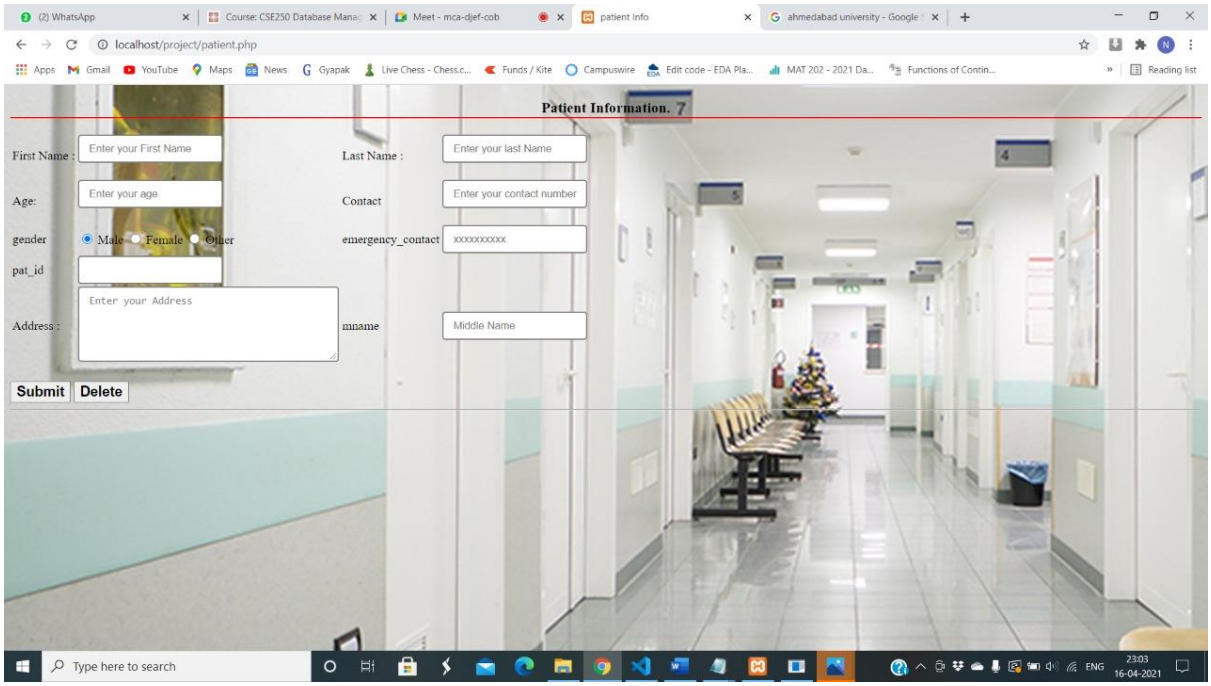
Our Project Definition is Hospital Management System. There is a ton of data that needs to be processed and retrieved on a daily basis for a massive organization like a hospital. There is details of patients and their data. Employees and their data. They also have to keep a track of inventory, departments, room allocation to patients. They also need to keep a track of all of this data and maintaining a database becomes a very integral part for their functioning. Thus we have tried to model at a small scale the Hospital Database Management System.

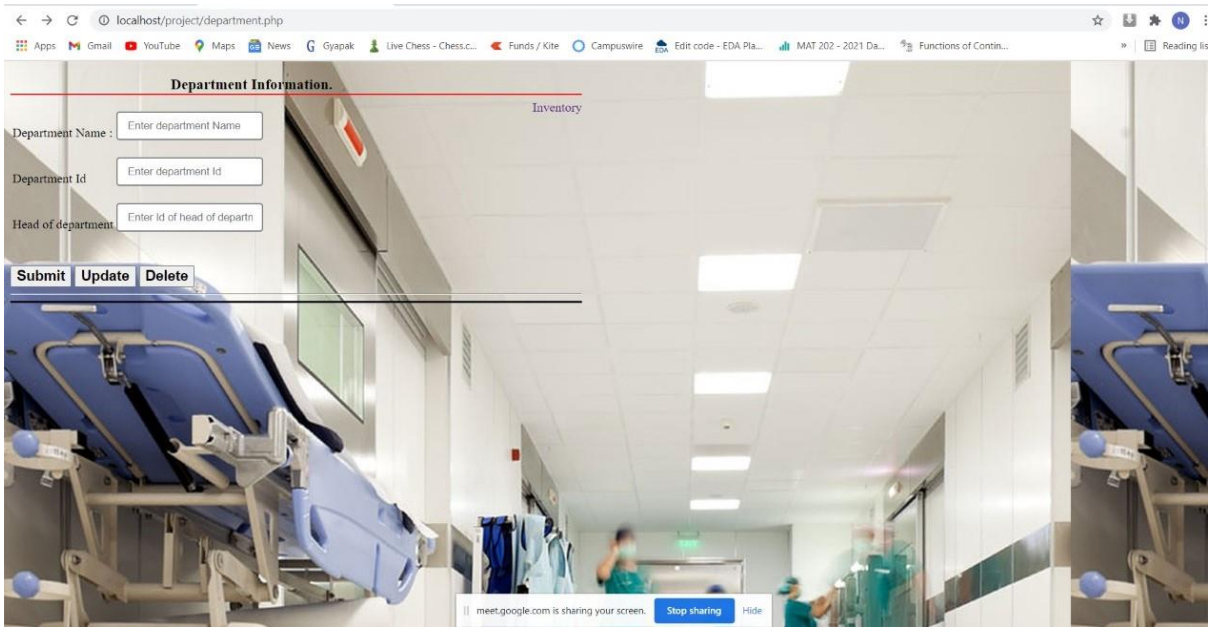
Features:

Our Project revolves around making a Hospital Management System. Our primary focus has been to comprehensively cover all the facets of a Hospital Management System. We have the function to Register a Patient, admit a Patient in a Room, Manage the Inventory of Hospital, Maintain Employee Records, Segregate the doctors into departments. We also have the feature of assigning doctors to patients, assigning treatments, maintaining pharmacy and medical cost records. We have also Implemented feature of Calculating Bill taking into consideration all the expenses of doctors' fees of diagnosis, treatment charges etc.

Overview Images:







System Requirements and Implemented Functions:

Backend:

We have created the database as backend in Oracle Database.

Oracle Versions-11c, 12c,18c Community Editions are all Compatible.

Frontend:

The Frontend is done on PHP with XAMMP as the server and Oracle is Connected and Integrated with the Frontend Forms.

We have created CRUD functions for most of the tables.

Tables:

We have a made a CRUD page for almost all pages where we can perform operations on Tables.

We have made 17 tables in the Database namely:

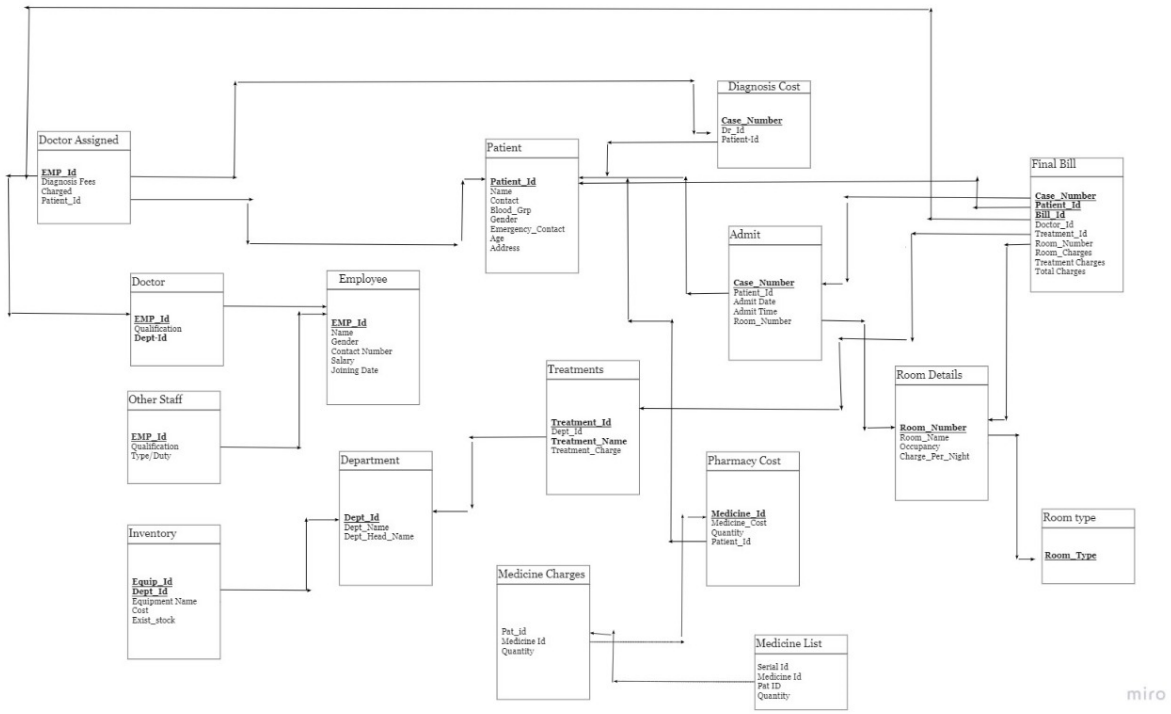
Table Name	The function
Patient	Patient Registration and Details
Employee	Employee Details
Type	Room Types like ICUS, OT etc.
Room Details	Room Details of each Room Number
Treatment Given	The treatment that patient undergoes
Department	Information of Different Departments
Doctor	Doctor Info
Other Staff	General Staff Info
Inventory	Inventory keeps track of med. Equipment
Treatments	List of treatments available dept. wise
Doctors Assigned	List of Doctors and Patients Assigned to them
Diagnosis	The diagnosis table
Admit	Table to admit patients into a room
Pharmacy	List of Medicines available
Medical Charge	The Medical Charge of Patient
Medicine List	List of all the medicines taken by a Patient
Bill	Final billing taking into account all of above expenses

Database Design:

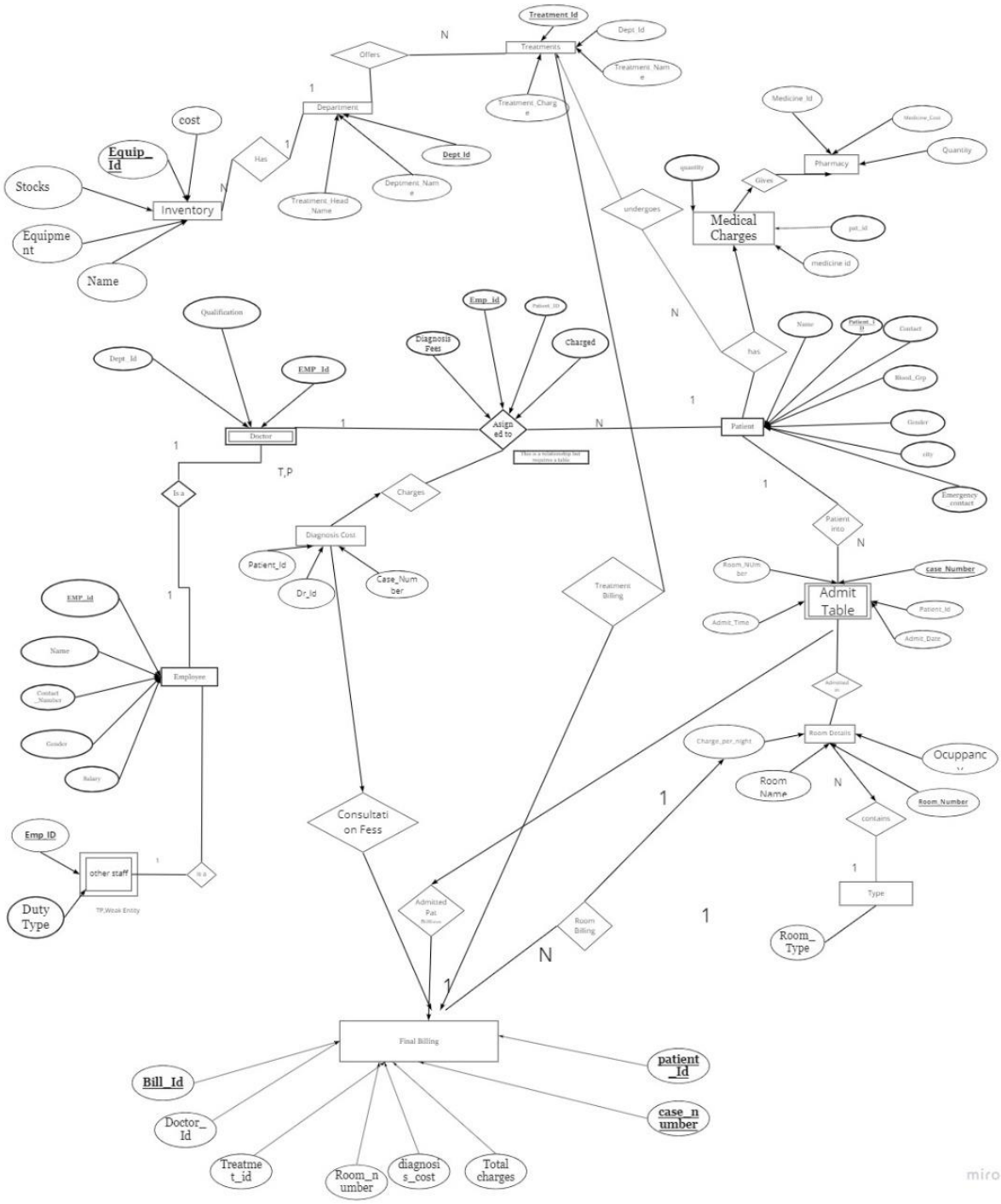
Here are the Schema and ER Diagram.

The List of Normalizations taken into account are mentioned in a separate section,

Schema:



E-R Diagram:

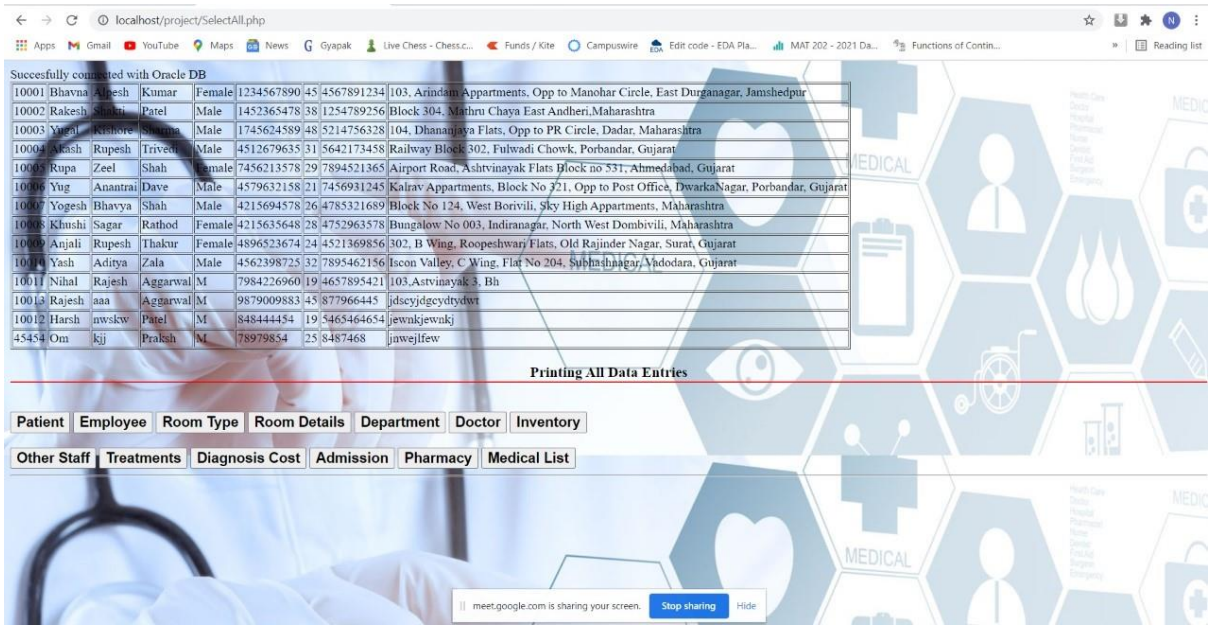


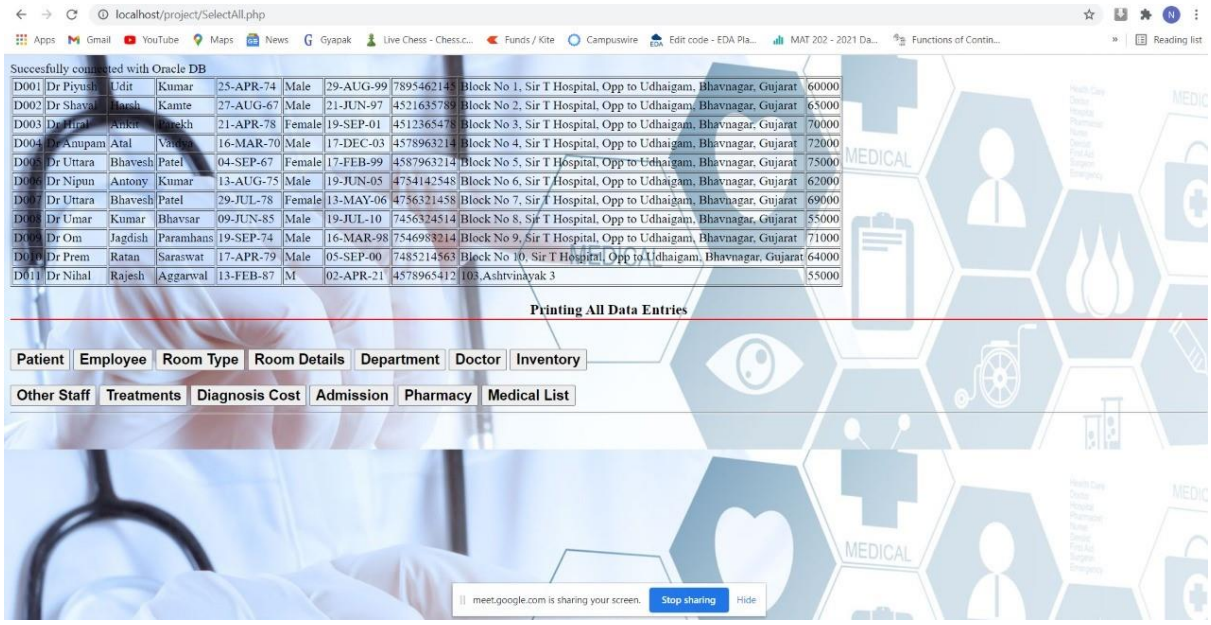
Normalizations and additional tables taken into account:

1) All Composite – are split into separate Attributes while implementing create commands
2) Tables like Room Details has attribute Room Type that has Partial Dependency on Room Number so We created a separate table that enumerates all the Room Types and then it can be directly referenced
3) Table Medical Charges has lot of MVAs (Multi Value Attributes) as a Patient can take lot of medicines and it is a variable number. We created a table Medical List that maps the Patient Id from Medical Charge and then that table can have a list of n number of medicines that are given to patient. There is also an accompanying trigger that inserts the data in Medical List after insertion in Medical Charge.
4) We have also checked for Partial and Transitive Dependencies and we found one in Doctor Assigned table which we solve by changing the primary key.
5) There is a relation between doctor and patient of assigned to which needs a table to store which doctor has been assigned to which patient.
6) Patient Undergoes Treatment is also a relationship that is put in a table.

Queries:

For execution of simple Query execution, we have put a separate page on the Frontend. The Page acts in a manner that it has buttons for named after all the tables. Whenever a Table button is pressed the Data of the Table is Displayed. This helps in retrieving data from all the tables at one place. The data of any table can be read from the page:





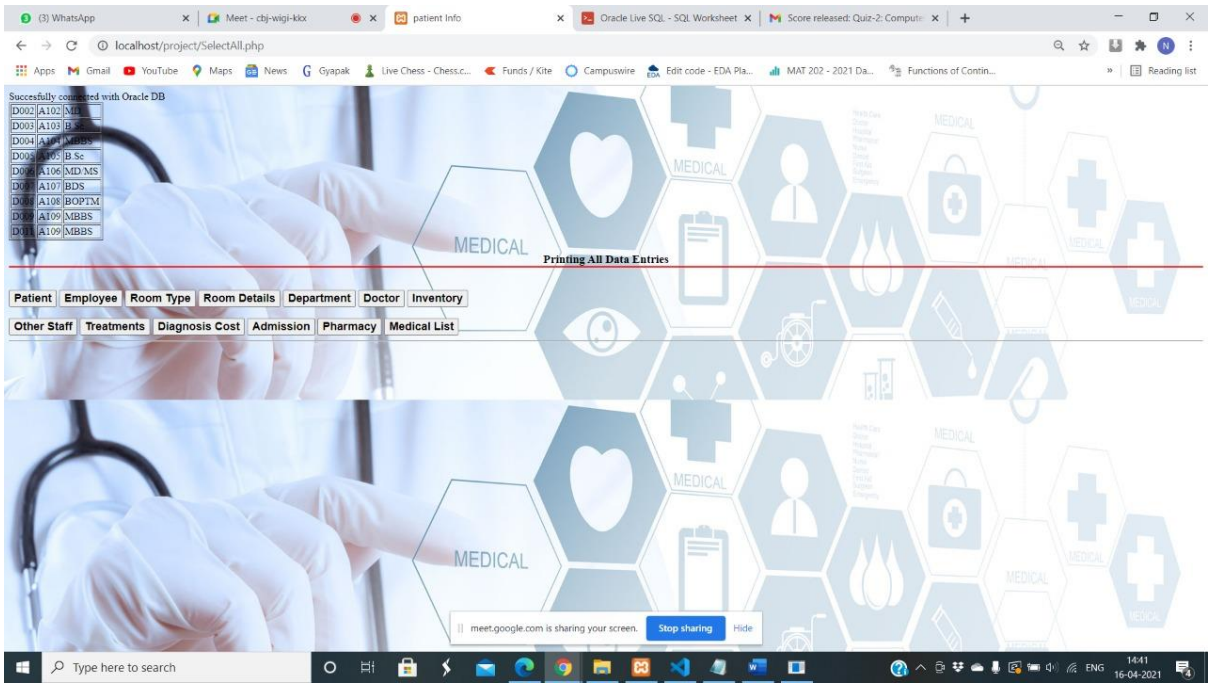
List of Triggers and Screenshots:

1)Trigger for Handling all the Tables to which Employee is a foreign key

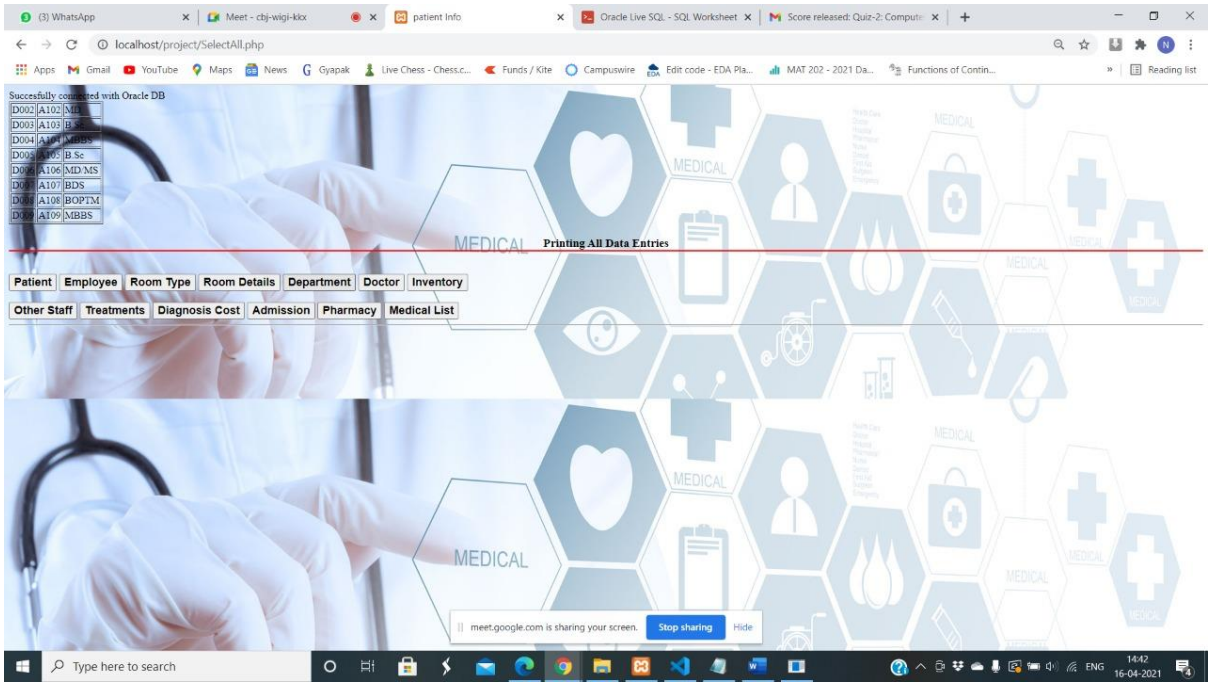
```
create or replace trigger del_rec before delete on employee
for each row
begin

    delete from doctor where doctor.emp_id=:old.emp_id;
    delete from other_staff where other_staff.emp_id=:old.emp_id;
    update doctor_assigned set dr_id=null where dr_id=:old.emp_id;
    update diagnosis_cost set dr_id=null where dr_id=:old.emp_id;



end;
/
```

Screenshots:

The view of Doctor Table before delete on Employee

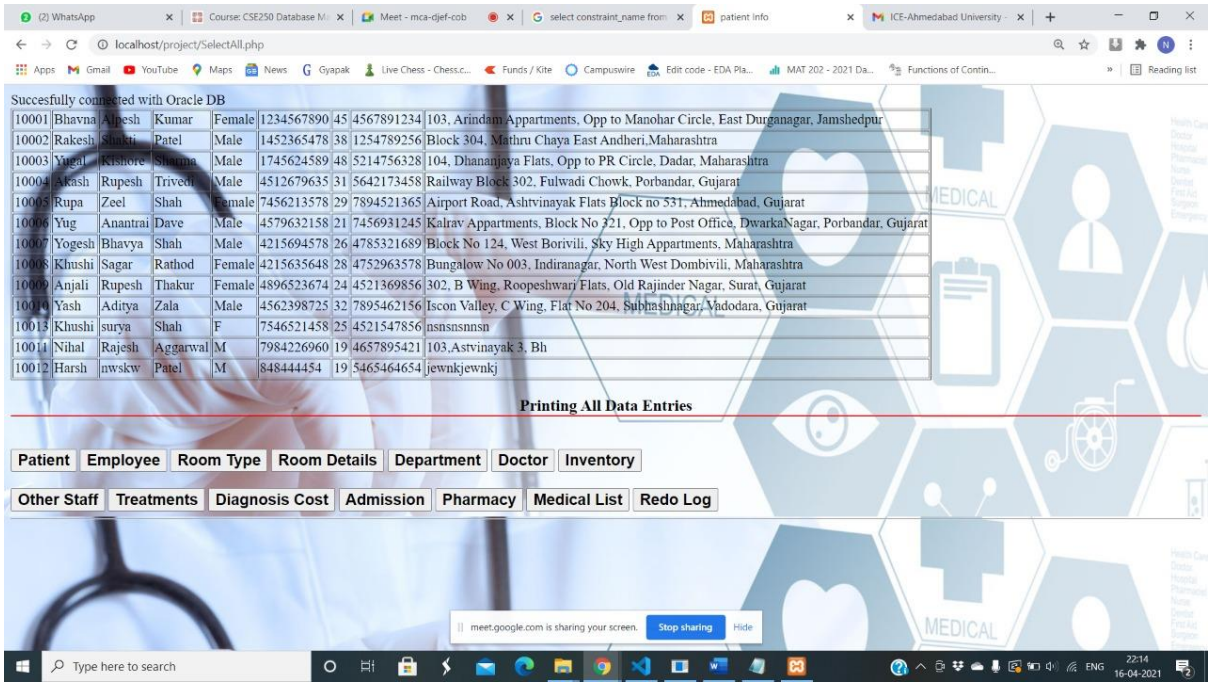


The view of doctor Table after delete on Employee



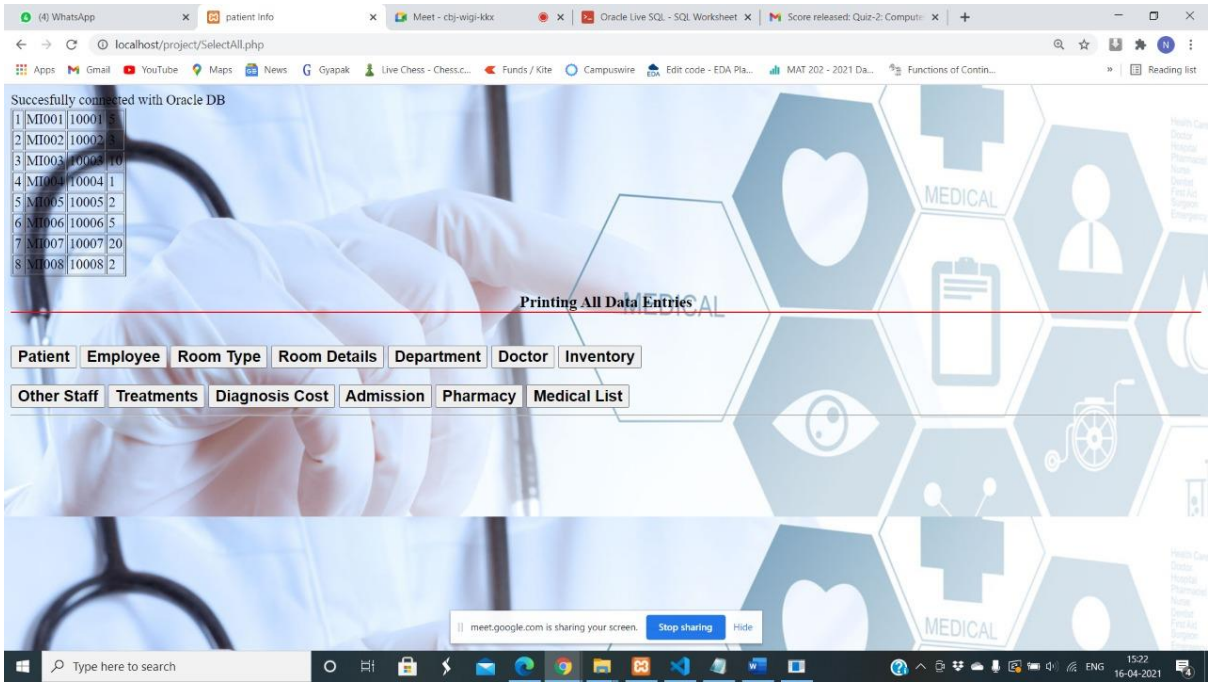
2) Trigger for Handling all child Tables where Patient is a Reference

```
create or replace trigger del_admitted_pat before delete on patient
for each row
begin
    delete from admit where admit.patient_id=:old.pat_id;
    update doctor_assigned set pat_id=null where pat_id=:old.pat_id;
    update diagnosis_cost set pat_id=null where pat_id=:old.pat_id;

end;
/
```



3) Trigger to Insert values into Medical List after Inserting in Medical Charges

```
create or replace trigger test1 after insert on medical_charge
for each row
declare
        cursor count is select count(serial_no) from medicallist;
        sid int;
begin
        open count;
        fetch count into sid;
        close count;
        sid:=sid+1;
        insert into medicallist values(sid,:new.medicine_id1,:new.pat_id,:new.quantity);
end;
/
```



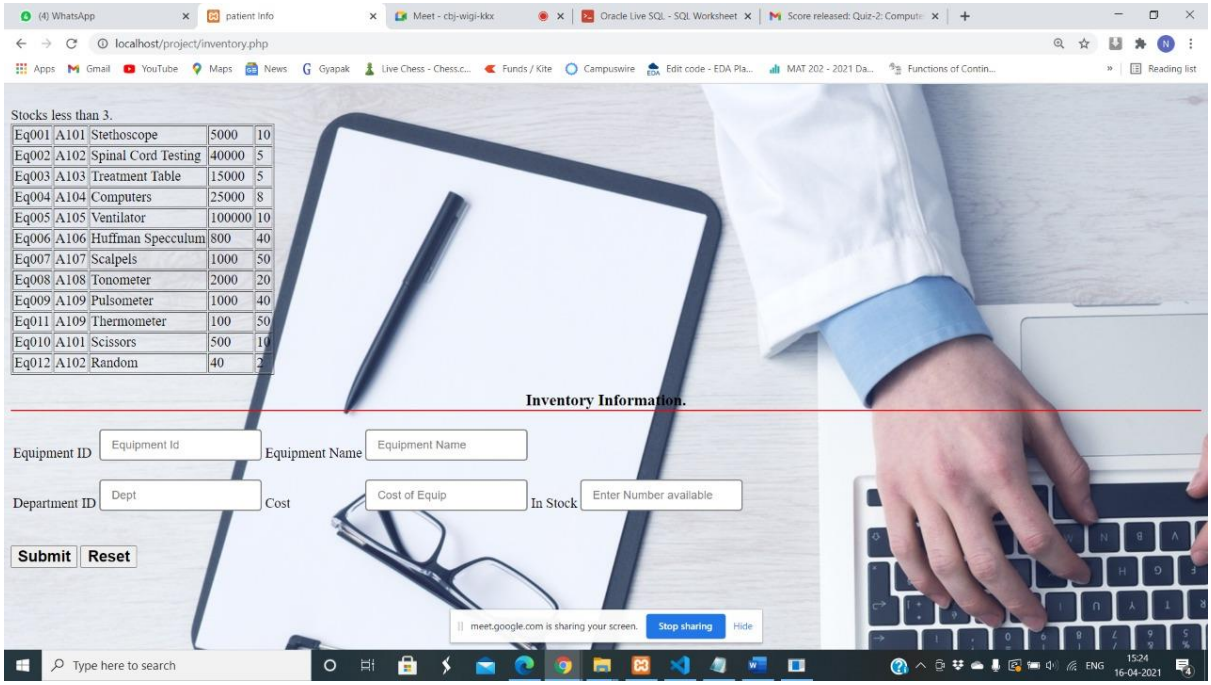
4)Trigger for Intimation on Low Stock Prices

```
create or replace trigger stockalert after update or insert on inventory
for each row
begin
        if (inserting) then
                if (:new.stock <3) then
                        raise_application_error(-20199,'Stocks Less Than 3');
        end if;
        elsif (updating) then
                if (:old.stock<3) then
                        raise_application_error(-20012,'Stocks Less Than 3');
        else
                dbms_output.put_line('Stocks are fine');
                end if;
        end if;
```

end;

/

Screenshot



5)Trigger for keeping a Log of the Audit and Before After Stocks on Inventory

create table redolog_values(equip_id varchar(10) , equip_name varchar(30), before_stock int,after_stock int);

create or replace trigger chk_redolog after update on inventory

for each row

begin

if(:new.stock<>:old.stock) then

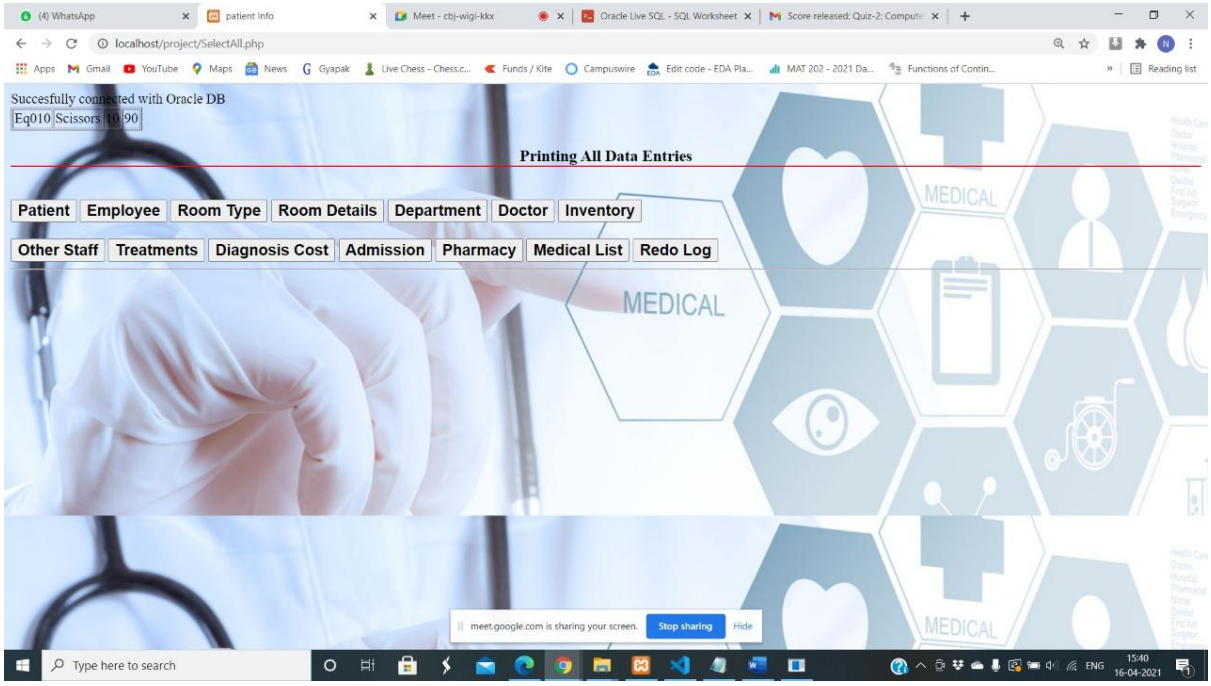
insert into redolog_values values(:old.equip_id,:old.equip_name,:old.stock,:new.stock);

end if;

end;

/

Output : Alert that Stocks less than 3



6)Trigger for Security Breach and Data Entry in Doctor Table

create table securtiy(user_name varchar2(20),current_date varchar(20), time varchar2(20));

create or replace trigger chk_trap after insert or update or delete on doctor

for each row

begin

if(to_char(sysdate,'dy')='sat' or to_char(sysdate,'dy')='sun' or to_number(to_char(sysdate,'HH24'))<6 or to_number(to_char(sysdate,'HH24'))>22) then

insert into security values(user,to_char(sysdate),SYSTIMESTAMP);

end if;

end;

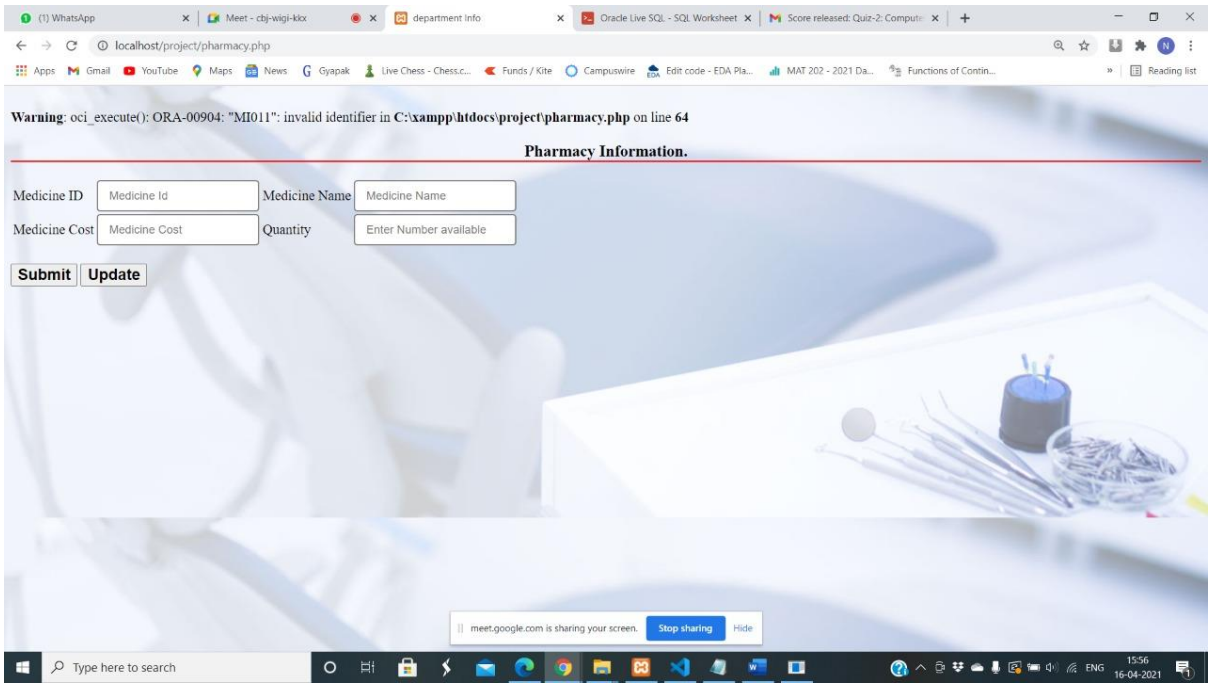
/

Is Applicable Only When Data is inserted or updates after 10 PM or Saturdays Sundays

7)Checking Valid Medicine Names

```
create or replace trigger chk_cost before insert or update on pharmacy
for each row
begin
    if (:new.medicine_name is null) then
        raise_application_error(-200016,'Enter a Proper Medicine Name ');
    end if;
end;
/
```

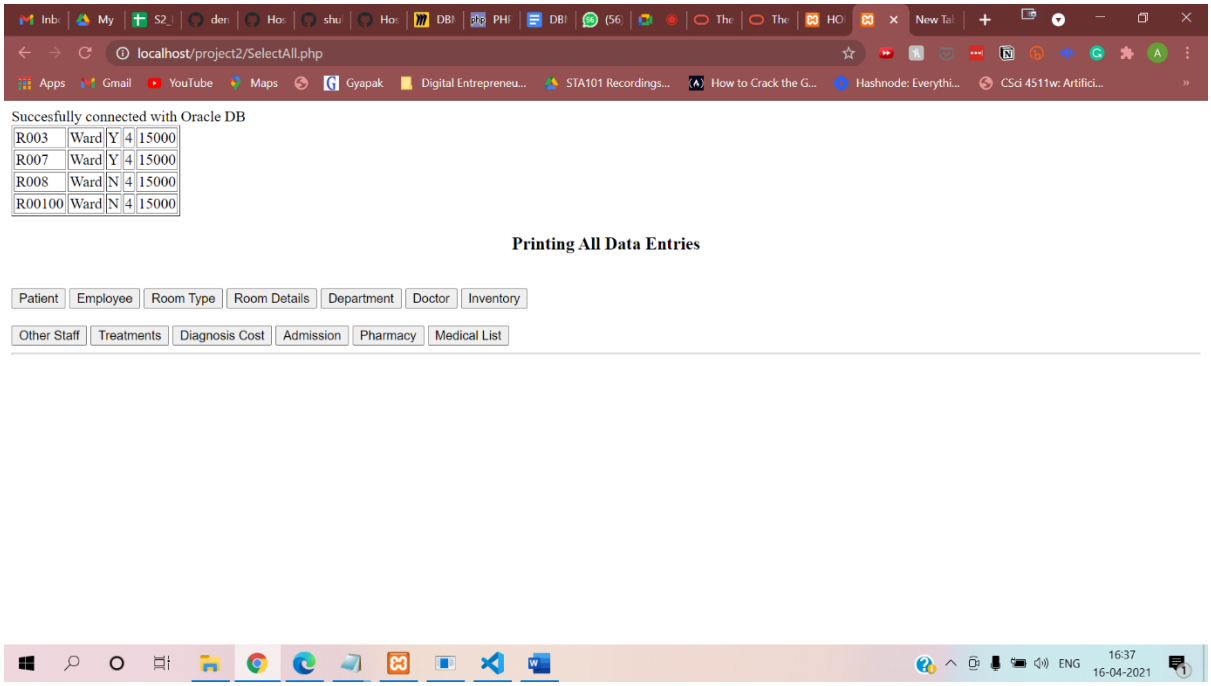
Screenshot:



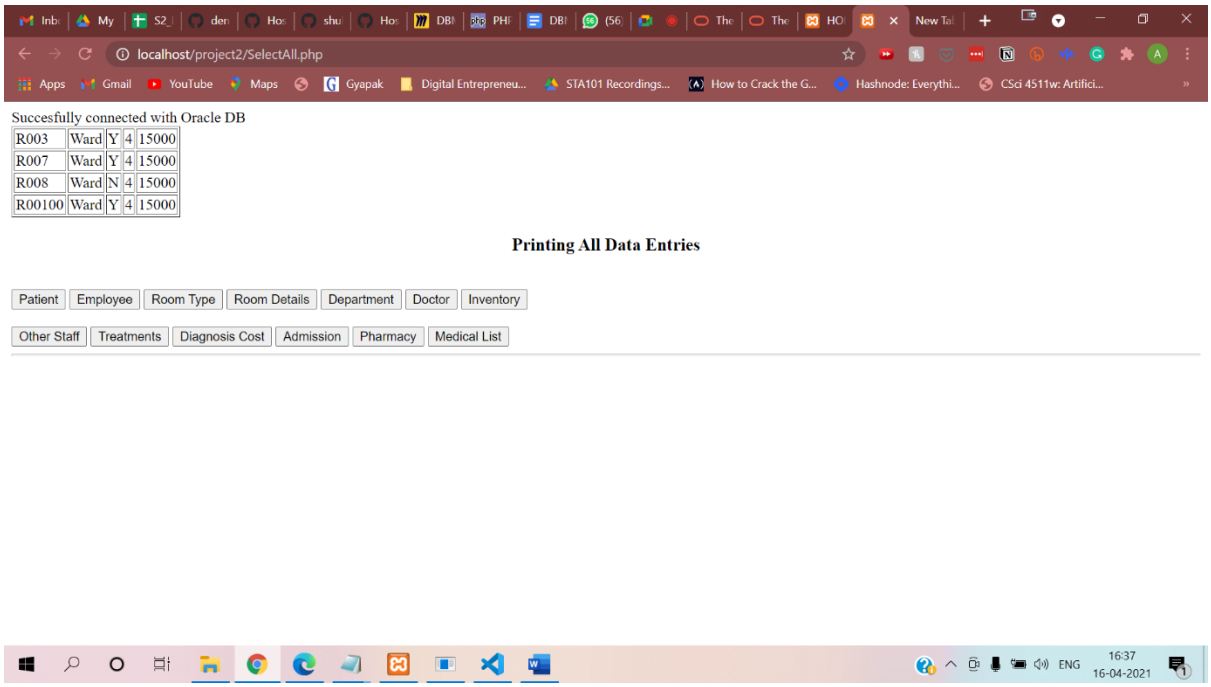
9) Trigger for Changing Occupancy Status of a Room when allocated to a Particular Patient

```
create or replace trigger changestatus after insert or update on admit
for each row
declare
        cursor c_status is select * from room_details;
        r_status  c_status%rowtype;
begin
        open c_status ;
        loop
        fetch c_status into r_status;
                if c_status%NOTFOUND then
                exit;
                end if;
        if (r_status.room_no=:new.room_no) then
                update room_details set occupancy='Y' where room_no=:new.room_no;
        end if;
        end loop;
        close c_status;
end;
/
```

Screen Shot:



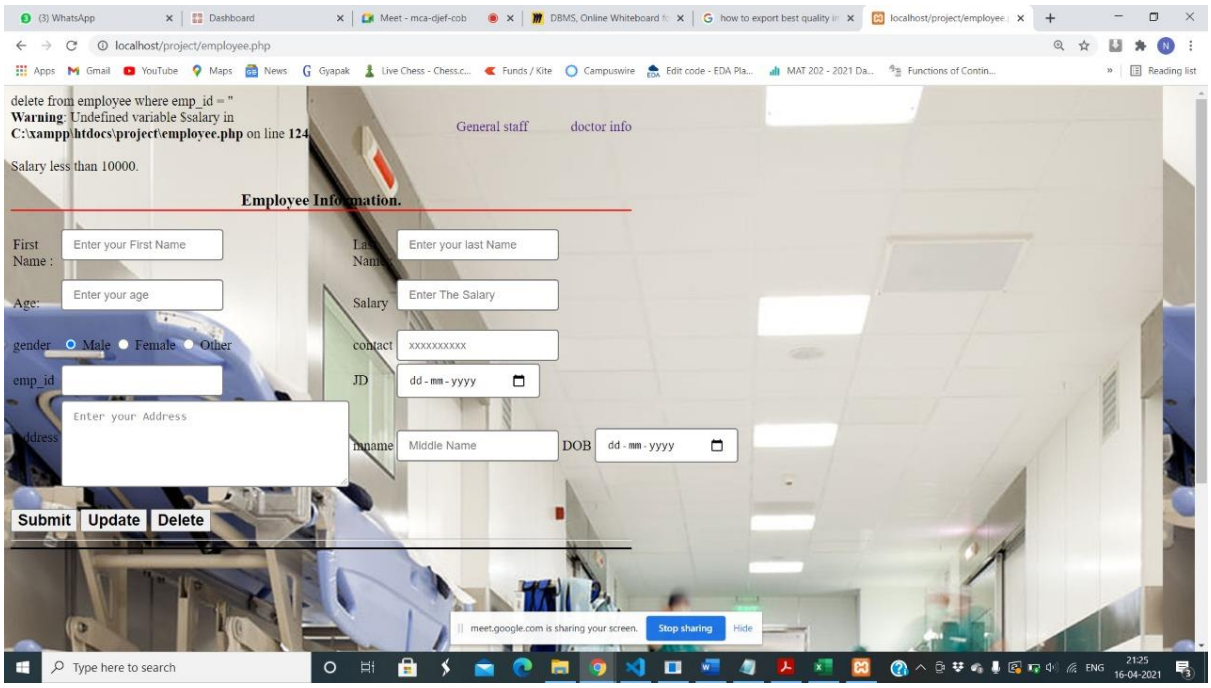
After inserting into admit the Occupancy Status is Changed to Y:



10) To Set Salary minimum cap of 10000

```
create or replace trigger check_employee_salary
  before insert or update on employee
  for each row
```

```
begin
  if (:NEW.salary < 10000 ) then
     raise_application_error(-20005,'The entered value in salary is less than 10000!!');
end if;
end;
/
```

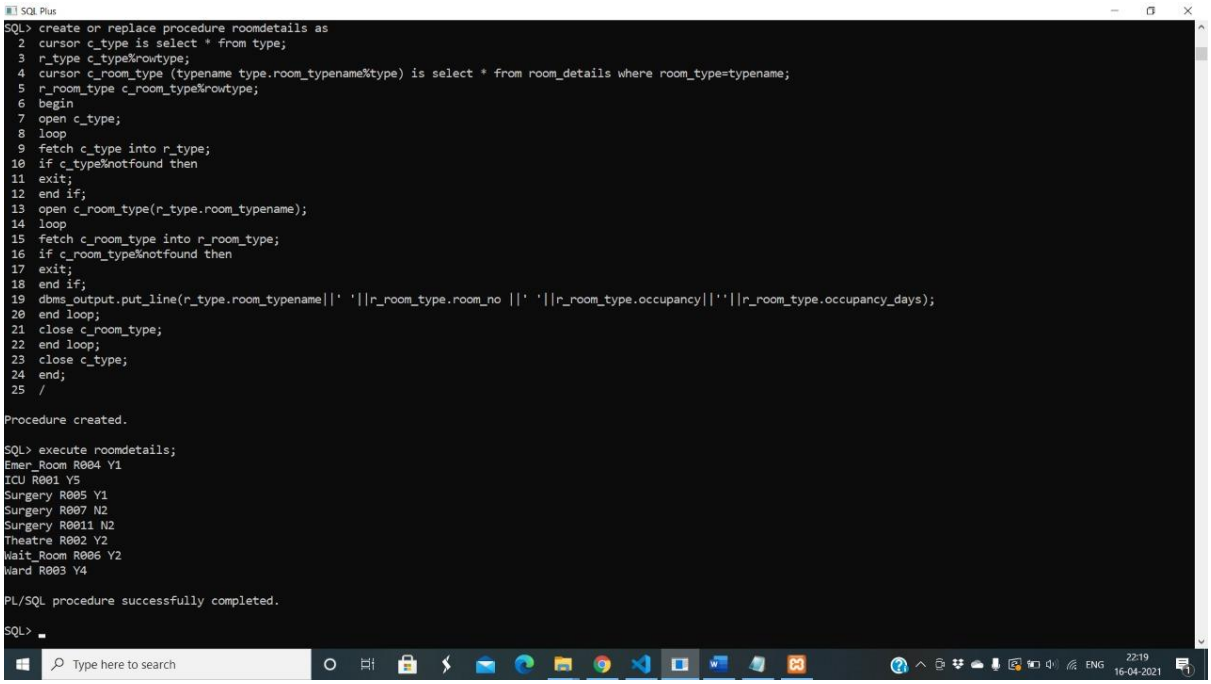


List of Stored Procedures:

1)Procedure to Input Room Type and Get List of Rooms in that Types and their Details

```
create or replace procedure roomdetails as
cursor c_type is select * from type;
```

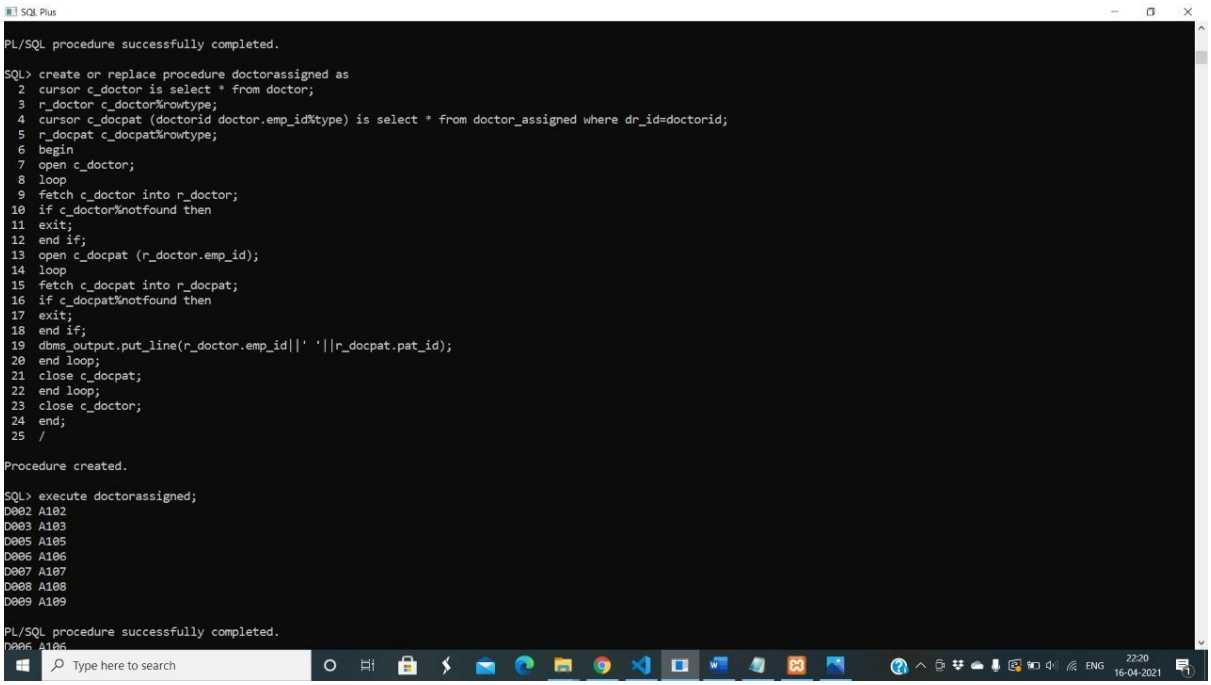
```
r_type c_type%rowtype;
cursor c_room_type (typename type.room_typename%type) is select * from room_details
where room_type=typename;
r_room_type c_room_type%rowtype;
begin
open c_type;
loop
fetch c_type into r_type;
if c_type%notfound then
exit;
end if;
open c_room_type(r_type.room_typename);
loop
fetch c_room_type into r_room_type;
if c_room_type%notfound then
exit;
end if;
dbms_output.put_line(r_type.room_typename||' '||r_room_type.room_no ||'
'||r_room_type.occupancy||''||r_room_type.occupancy_days);
end loop;
close c_room_type;
end loop;
close c_type;
end;
/
```



2)For Giving List of Doctors and Patients Assigned to Them

```
create or replace procedure doctorassigned as
cursor c_doctor is select * from doctor;
r_doctor c_doctor%rowtype;
cursor c_docpat (doctorid doctor.emp_id%type) is select * from doctor_assigned where dr_id=doctorid;
r_docpat c_docpat%rowtype;
begin
open c_doctor;
loop
fetch c_doctor into r_doctor;
if c_doctor%notfound then
exit;
end if;
open c_docpat (r_doctor.emp_id);
```

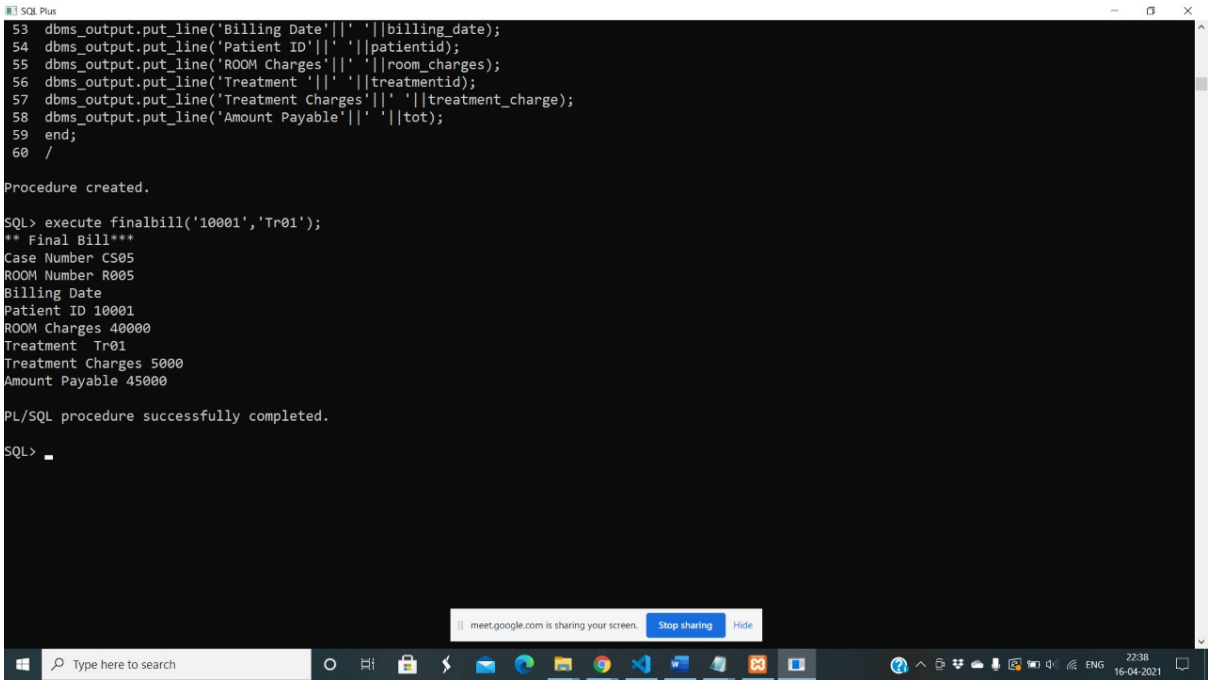
```
loop
fetch c_docpat into r_docpat;
if c_docpat%notfound then
exit;
end if;
dbms_output.put_line(r_doctor.emp_id||' '||r_docpat.pat_id);
end loop;
close c_docpat;
end loop;
close c_doctor;
end;
/
```



3) For Calculating Final Bill

```
create or replace procedure finalbill(pat_id varchar, treatment_id varchar) as
cursor c_patient is select * from patient;
r_patient c_patient%rowtype;
cursor c_treatments is select * from treatments;
r_treatments c_treatments%rowtype;
cursor c_admit(patientid patient.pat_id%type) is select * from admit where
pat_id=patientid;
r_admit c_admit%rowtype;
cursor c_room_details(roomno admit.room_no%type) is select * from room_details;
r_room_details c_room_details%rowtype;
patientid varchar(20);
treatmentid varchar(20);
case_number varchar(20);
billing_date date;
room_no varchar(20);
room_charges int;
treatment_charge int;
tot int;
begin
for r_patient in c_patient loop
if (r_patient.pat_id=pat_id) then
patientid:=r_patient.pat_id;
end if;
end loop;
for r_treatments in c_treatments loop
if(r_treatments.treatment_id=treatment_id) then
treatmentid:=r_treatments.treatment_id;
treatment_charge:=r_treatments.charge;
end if;
```

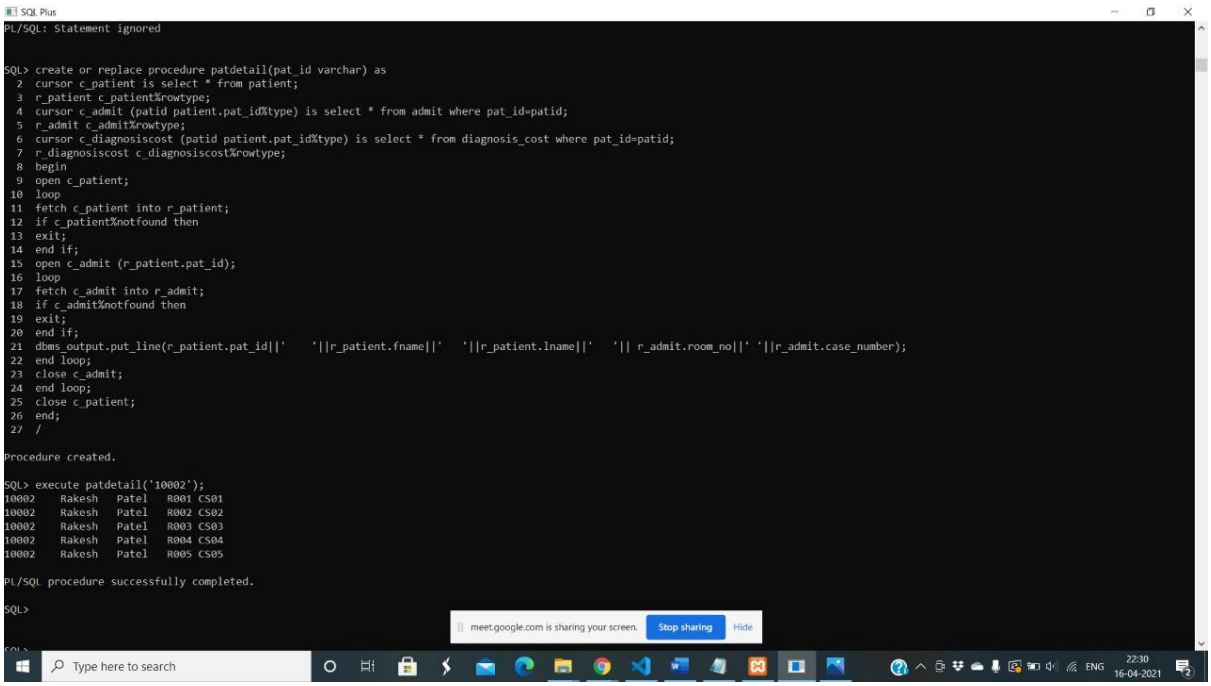
```
end loop;
open c_admit (patientid);
loop
fetch c_admit into r_admit;
if c_admit%notfound then
exit;
end if;
case_number:=r_admit.case_number;
room_no:=r_admit.room_no;
end loop;
close c_admit;
open c_room_details (room_no);
loop
fetch c_room_details into r_room_details;
if c_room_details%notfound then
exit;
end if;
room_charges:=r_room_details.occupancy_days*r_room_details.charge_of_room;
end loop;
close c_room_details;
tot:=room_charges+treatment_charge;
dbms_output.put_line(tot);
insert into final_bill values()
end;
/
```



4)Procedure to Input Patient ID and retrieve data of Room in which he is admitted, Diagnosis Cost and Other Details

```
create or replace procedure patdetail(pat_id varchar) as
cursor c_patient is select * from patient;
r_patient c_patient%rowtype;
cursor c_admit (patid patient.pat_id%type) is select * from admit where pat_id=patid;
r_admit c_admit%rowtype;
cursor c_diagnosiscost (patid patient.pat_id%type) is select * from diagnosis_cost where pat_id=patid;
r_diagnosiscost c_diagnosiscost%rowtype;
begin
open c_patient;
loop
fetch c_patient into r_patient;
if c_patient%notfound then
exit;
```

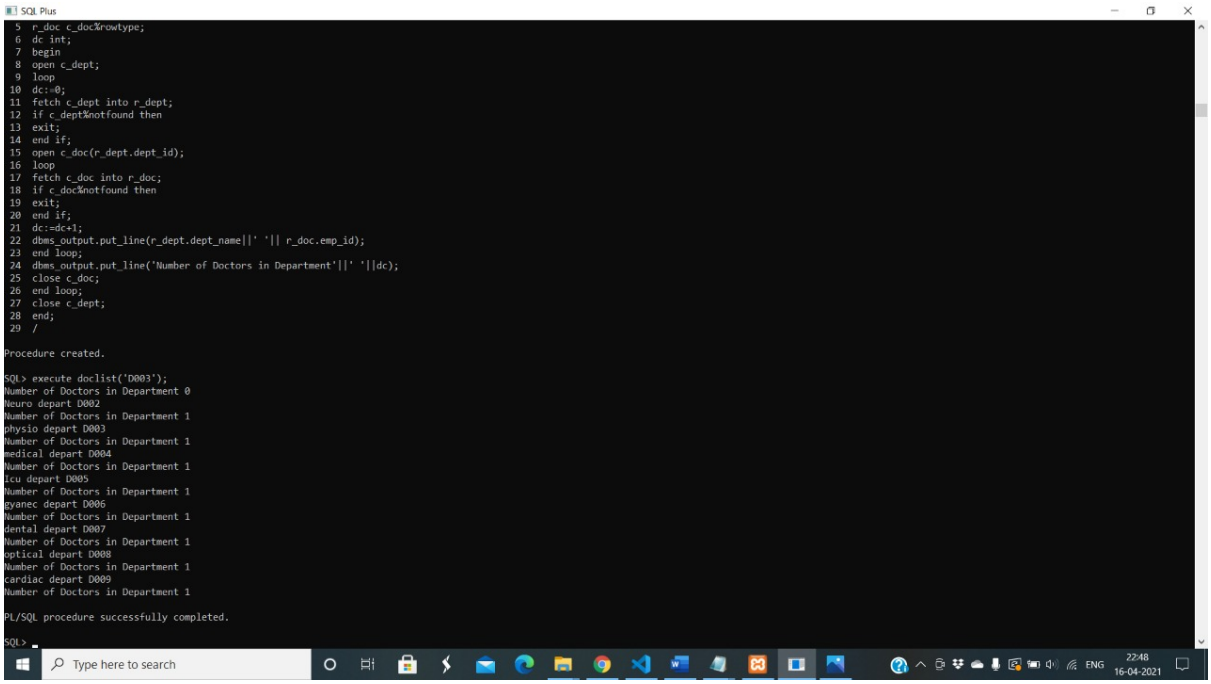
```
end if;

open c_admit (r_patient.pat_id);

loop

fetch c_admit into r_admit;

if c_admit%notfound then

exit;

end if;

dbms_output.put_line(r_patient.pat_id||'   '||r_patient.fname||'  '||r_patient.lname||'
'|| r_admit.room_no||' '||r_admit.case_number);

end loop;

close c_admit;

end loop;

close c_patient;

end;

/
```



5) For accessing List of Employees in a department

```
create or replace procedure doclist (dept_id varchar) as

cursor c_dept is select * from department ;

r_dept c_dept%rowtype;

cursor c_doc (deptid department.dept_id%type) is select * from doctor where
dept_id=deptid;

r_doc c_doc%rowtype;

dc int;

begin

open c_dept;

loop

dc:=0;

fetch c_dept into r_dept;

if c_dept%notfound then

exit;

end if;

open c_doc(r_dept.dept_id);

loop

fetch c_doc into r_doc;

if c_doc%notfound then

exit;

end if;

dc:=dc+1;

dbms_output.put_line(r_dept.dept_name||' '|| r_doc.emp_id);

end loop;

dbms_output.put_line('Number of Doctors in Department'||' '||dc);

close c_doc;

end loop;

close c_dept;

end;
```

/



6)Input Department and get list of available Treatments

create or replace procedure availabletreatments(dept_id varchar) as

cursor c_dept is select * from department ;

r_dept c_dept%rowtype;

cursor c_treatment(deptid department.dept_id%type) is select * from treatments where dep_id=deptid;

r_treatment c_treatment%rowtype;

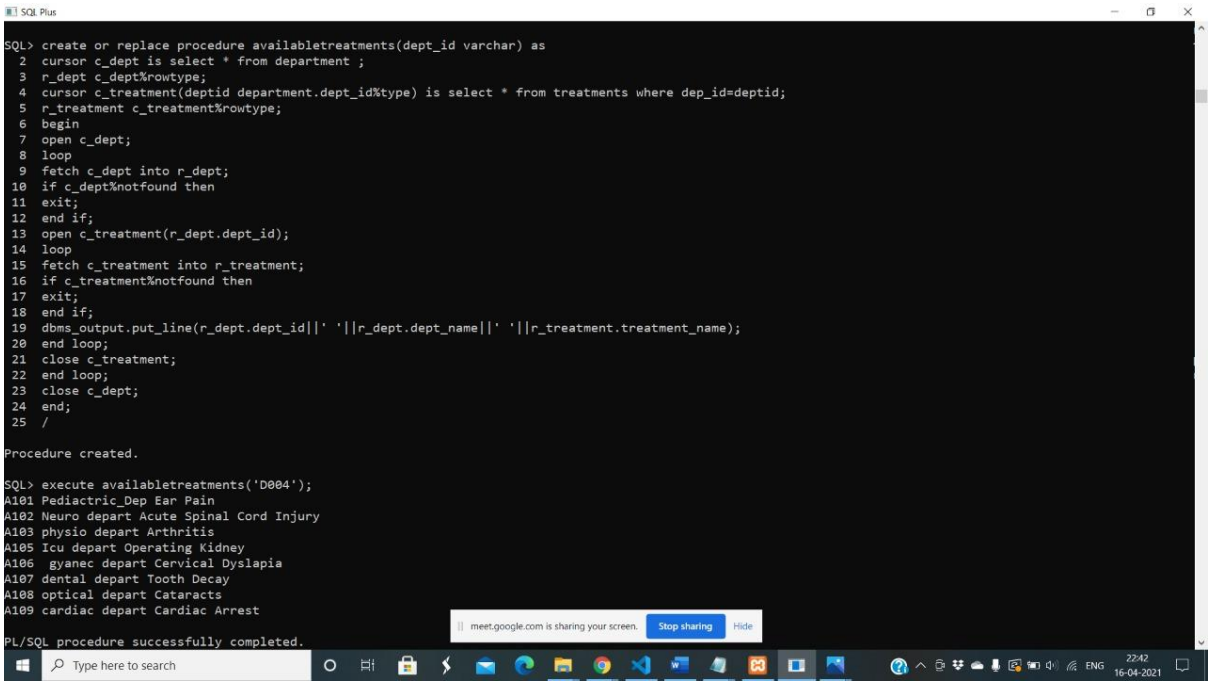
begin

open c_dept;

loop

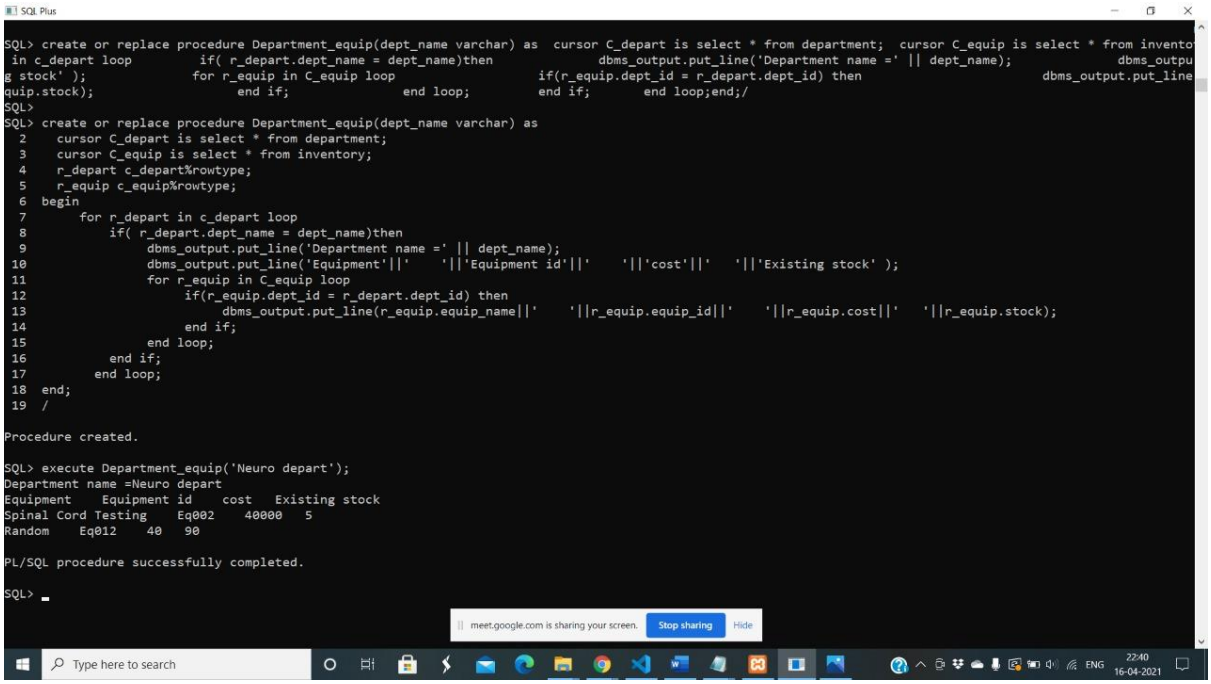
fetch c_dept into r_dept;

```
if c_dept%notfound then

exit;

end if;

open c_treatment(r_dept.dept_id);

loop

fetch c_treatment into r_treatment;

if c_treatment%notfound then

exit;

end if;

dbms_output.put_line(r_dept.dept_id||' '||r_dept.dept_name||'
'||r_treatment.treatment_name);

end loop;

close c_treatment;

end loop;

close c_dept;

end;

/
```



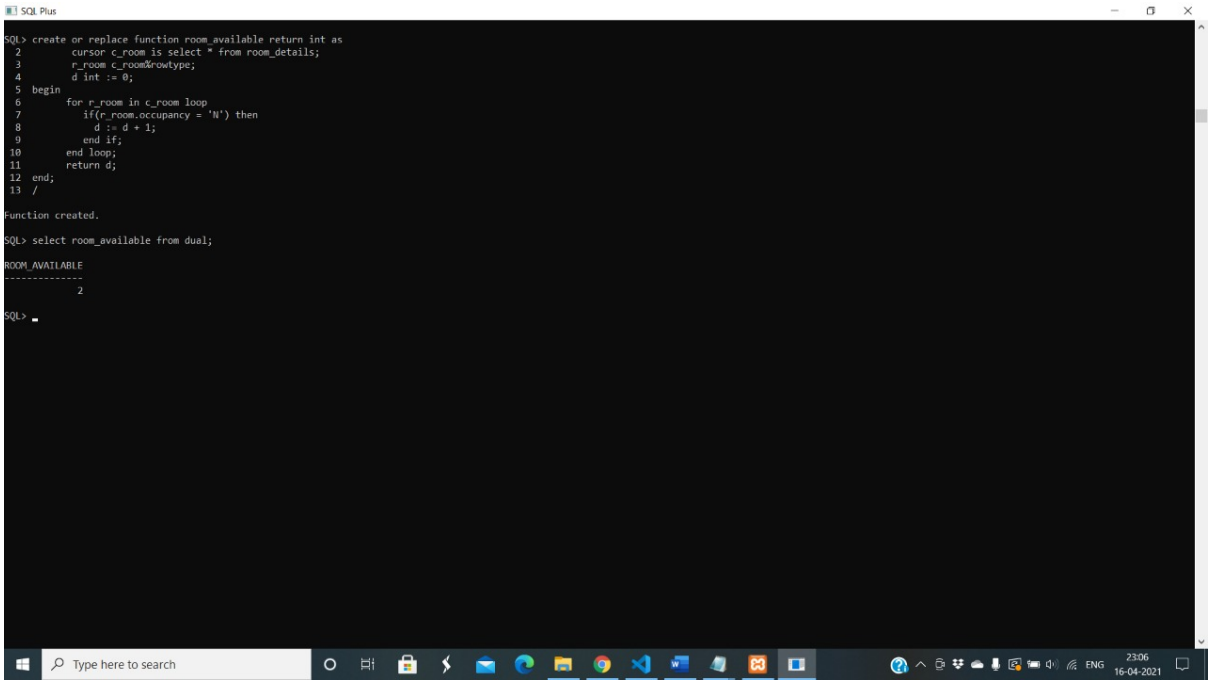
7)Input Department and Get all necessary Inventory details Department wise

```
create or replace procedure Department_equip(dept_name varchar) as
  cursor C_depart is select * from department;
  cursor C_equip is select * from inventory;
  r_depart c_depart%rowtype;
  r_equip c_equip%rowtype;
begin
    for r_depart in c_depart loop
       if( r_depart.dept_name = dept_name)then
          dbms_output.put_line('Department name =' || dept_name);
          dbms_output.put_line('Equipment'||'    '||'Equipment id'||'    '||'cost'||'
'||'Existing stock' );
          for r_equip in C_equip loop
             if(r_equip.dept_id = r_depart.dept_id) then
                dbms_output.put_line(r_equip.equip_name||'    '||r_equip.equip_id||'
'||r_equip.cost||'   '||r_equip.stock);
             end if;
          end loop;
       end if;
     end loop;
end;
/
```



8)Function to print the number of Rooms Available

```
create or replace function room_available return int as
      cursor c_room is select * from room_details;
      r_room c_room%rowtype;
      d int := 0;
begin
     for r_room in c_room loop
        if(r_room.occupancy = 'N') then
          d := d + 1;
        end if;
     end loop;
     return d;
end;
/
```


SQL Plus
SQL> create or replace function room_available return int as
2 cursor c_room is select * from room_details;
3 r_room c_room%rowtype;
4 d int := 0;
5 begin
6 for r_room in c_room loop
7 if(r_room.occupancy = 'N') then
8 d := d + 1;
9 end if;
10 end loop;
11 return d;
12 end;
13 /
Function created.
SQL> select room_available from dual;
ROOM_AVAILABLE
2
SQL>
Type here to search
ENG
23:06
16-04-2021