****

**DBMS Project Report**

**Project: Hospital Management System**

**Group Members:**

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

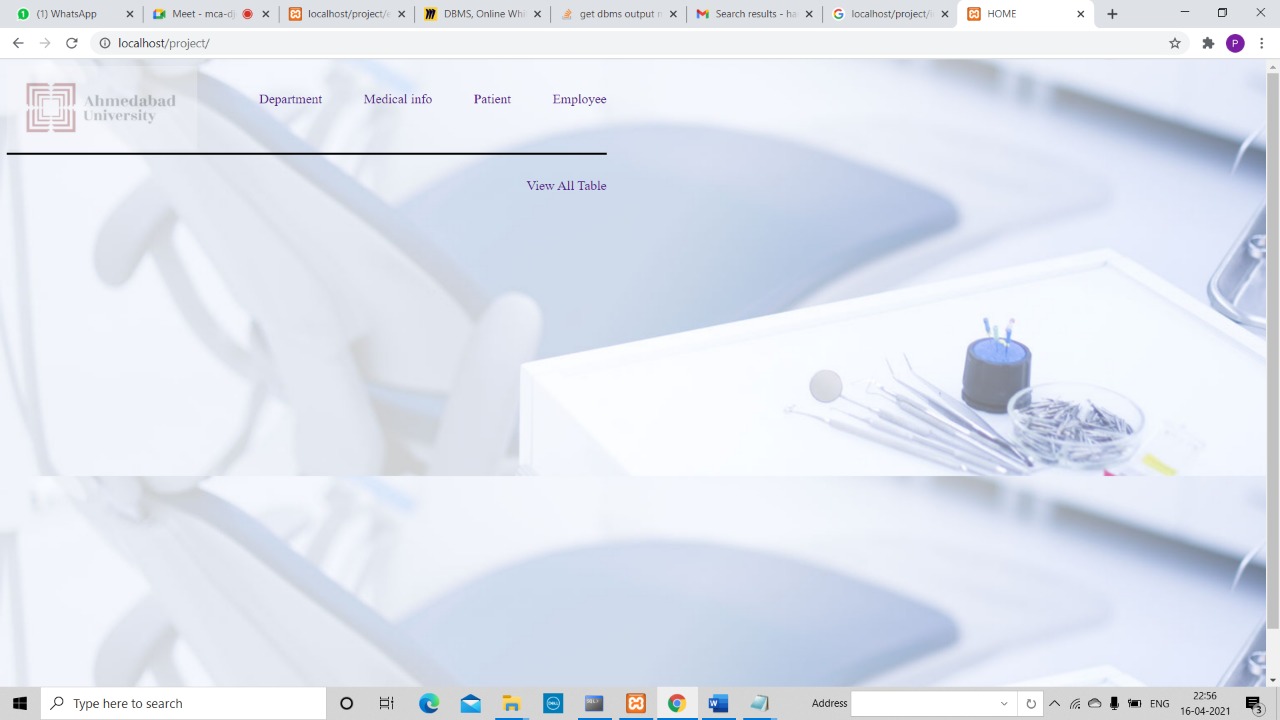
**Description:**

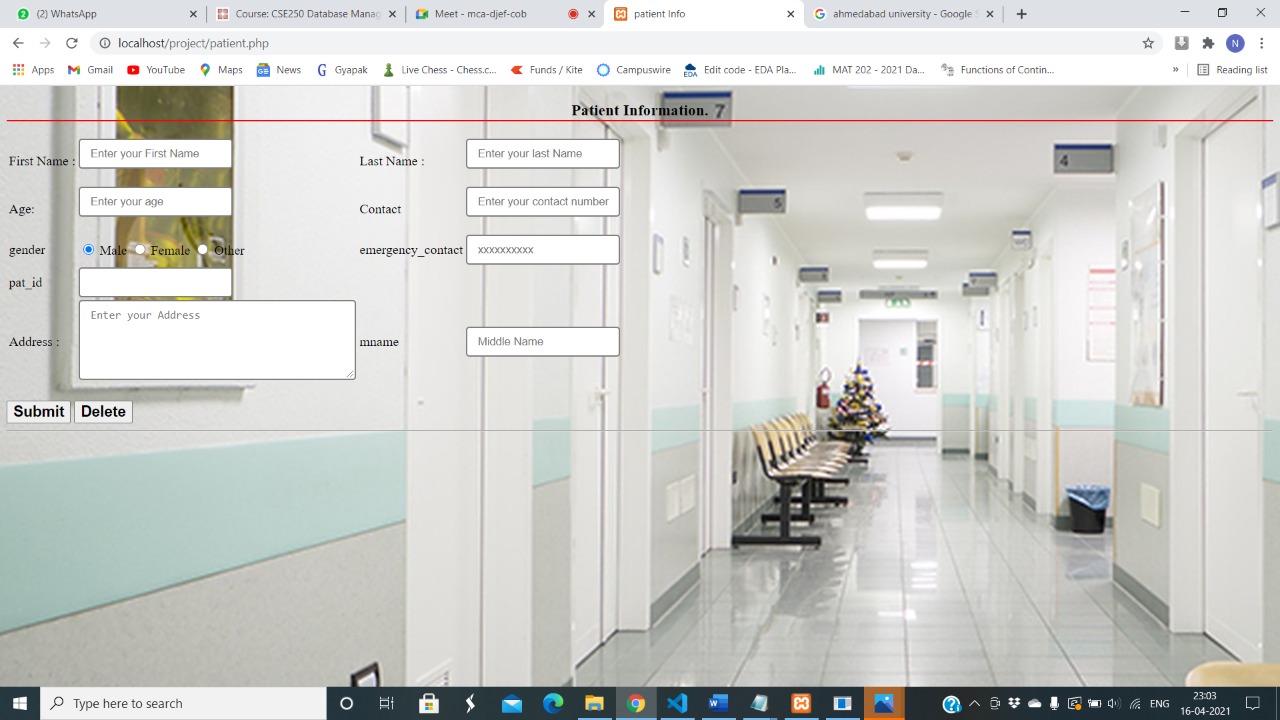
Our Project Definition is Hospital Management System. There is a ton of data that needs to be processed and retrieved on a daily basis for a massive organization like a hospital. There is details of patients and their data. Employees and their data. They also have to keep a track of inventory, departments, 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.

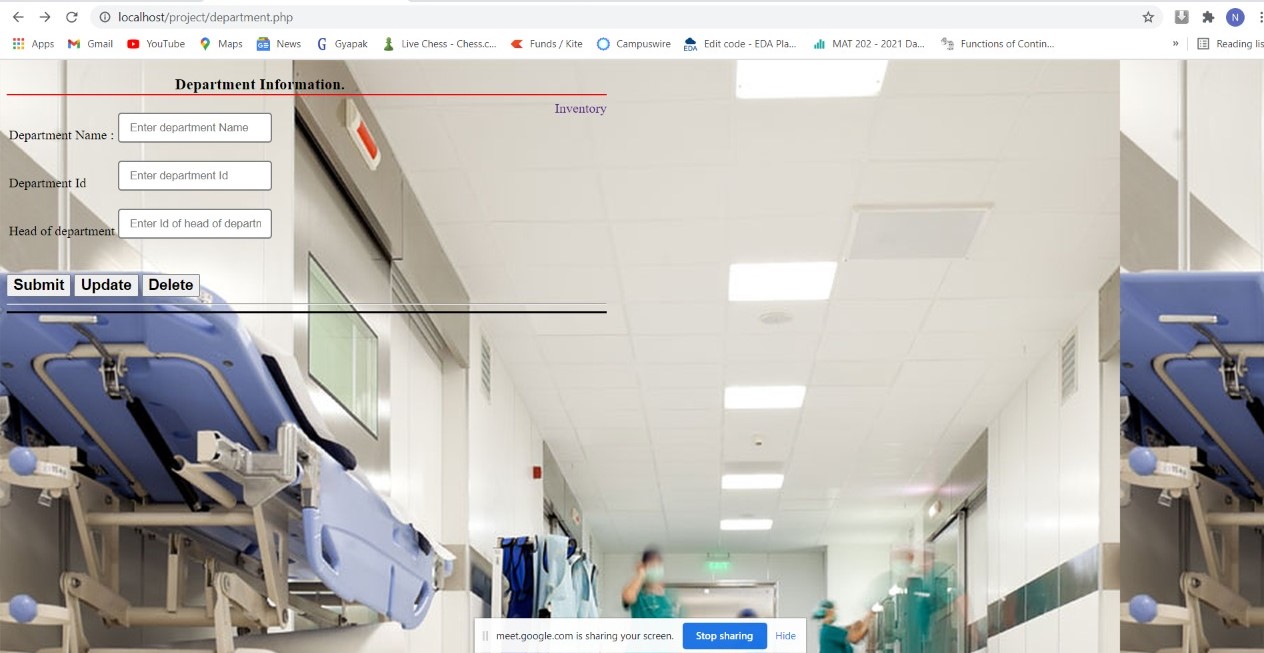
Features:

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:

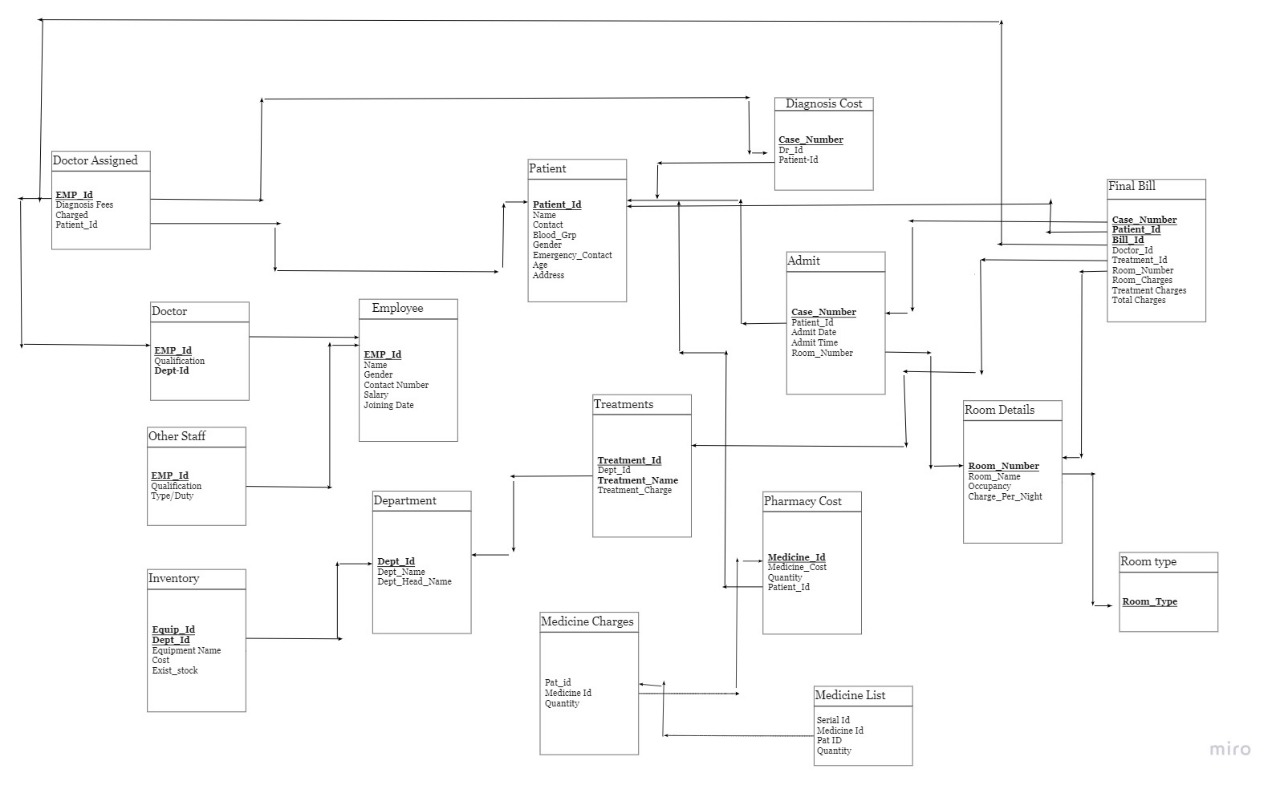
|  |  |
| --- | --- |
| Table Name | The function |
| Patient | Patient Registration and Details |
| Employee | Employee Details |
| Type | Room Types like ICUS, OT etc. |
| Room Details | Room Details of each Room Number |
| Treatment Given | The treatment that patient undergoes |
| Department | Information 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:**

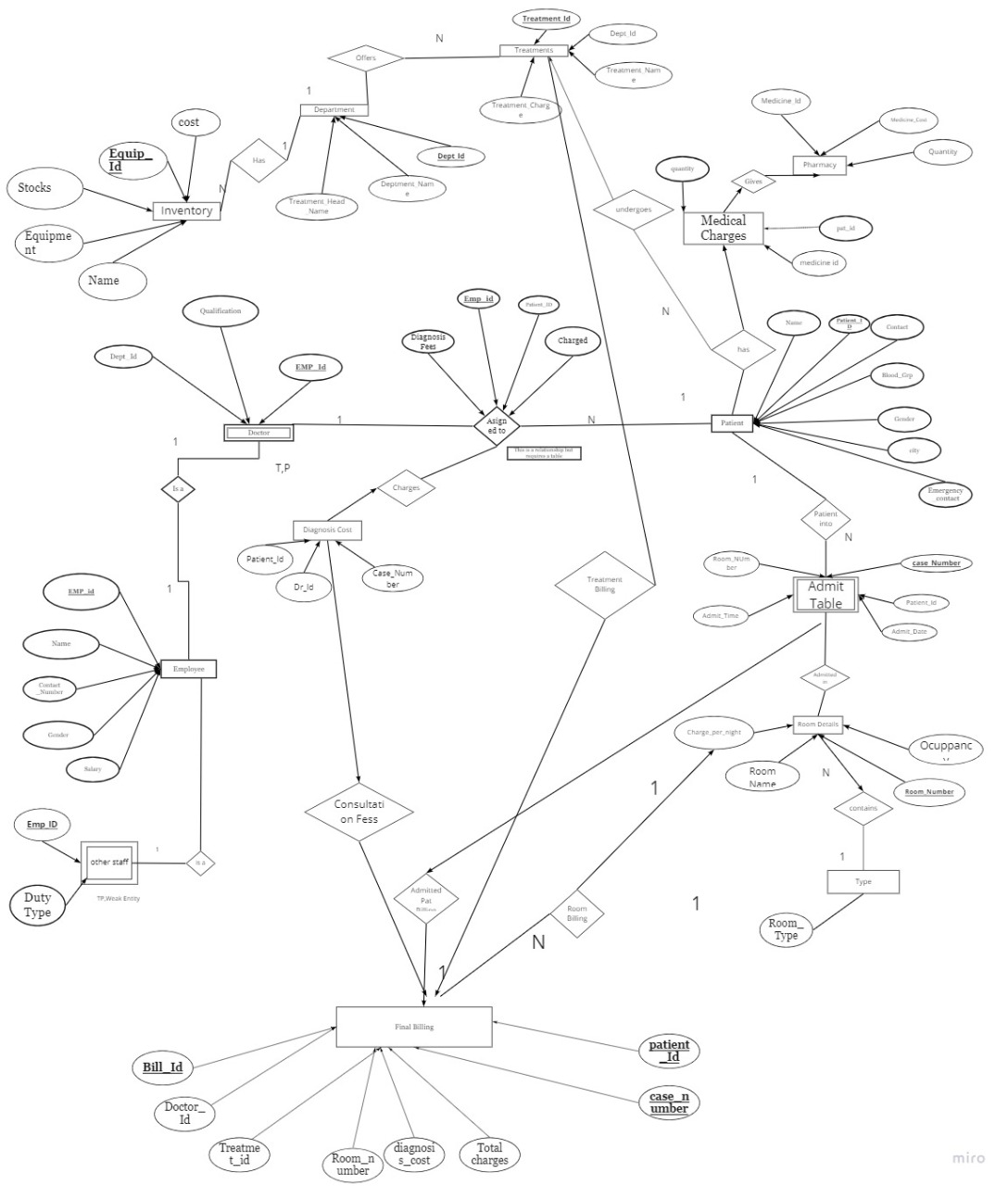
Here are the Schema and ER Diagram.

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

**Schema:**

****

**E-R Diagram:**



**Normalizations and additional tables taken into account:**

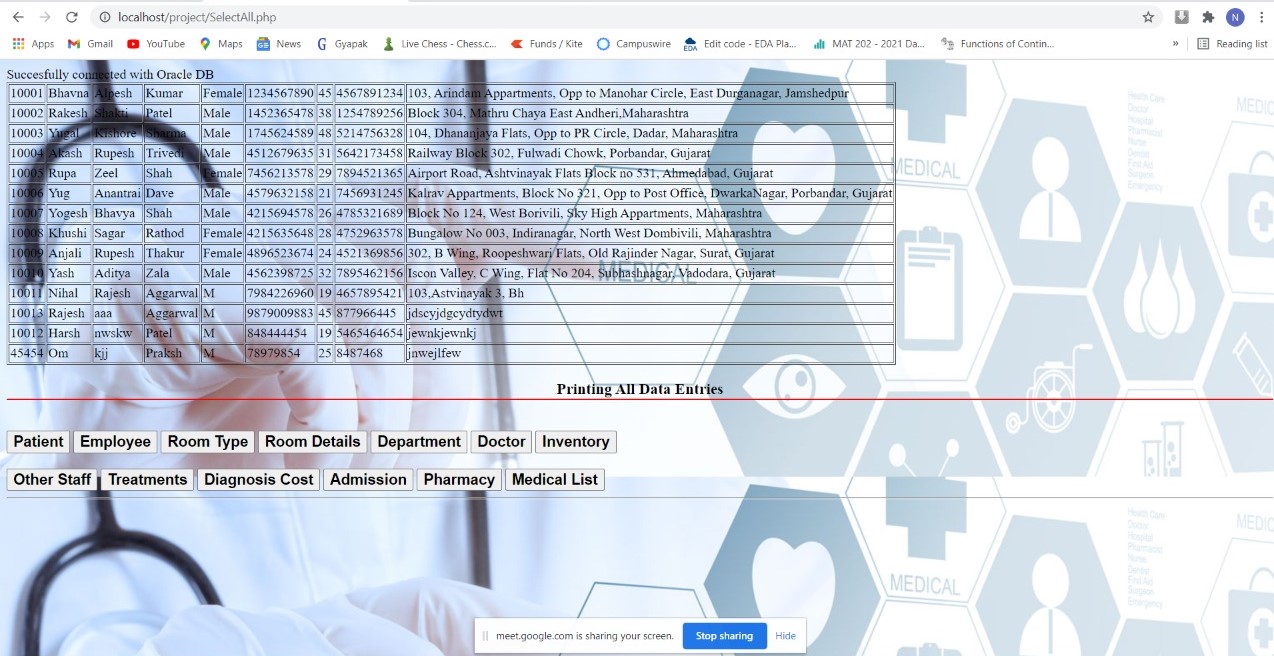
1. All Composite – are split into separate Attributes while implementing create commands
2. Tables like Room Details has attribute Room Type that has Partial Dependency on

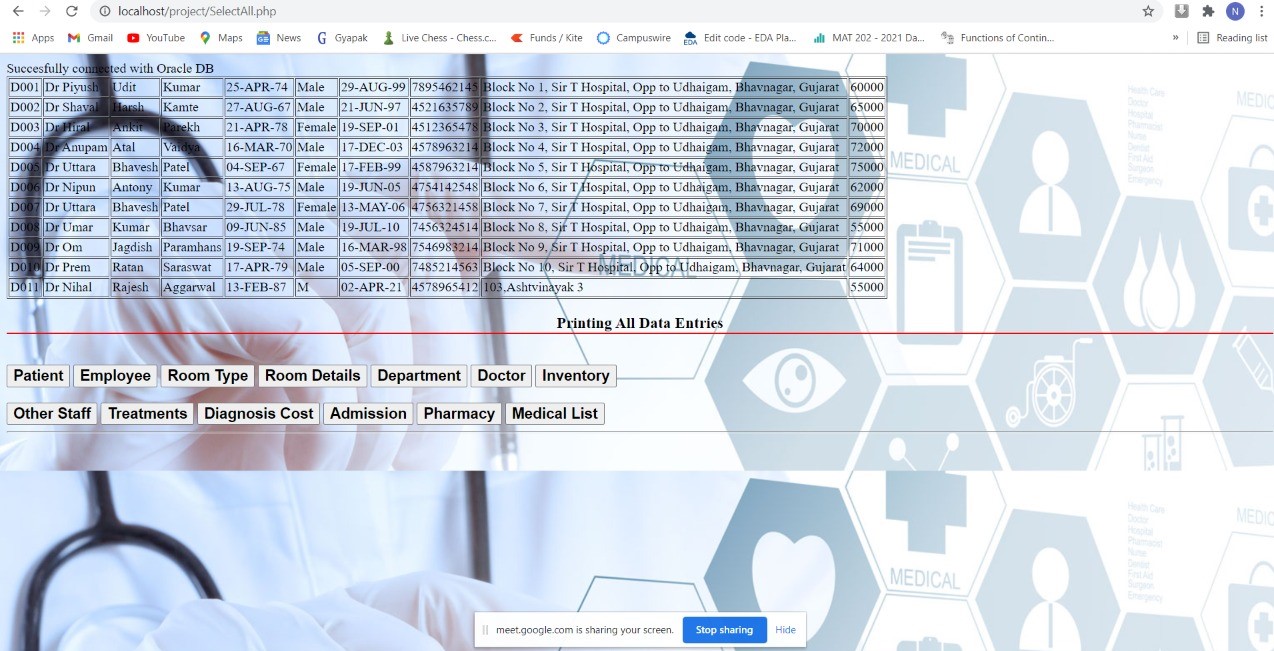
Room Number so We created a separate table that enumerates all the Room Types and then it can be directly referenced

1. 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.
2. 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.
3. 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.
4. 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;

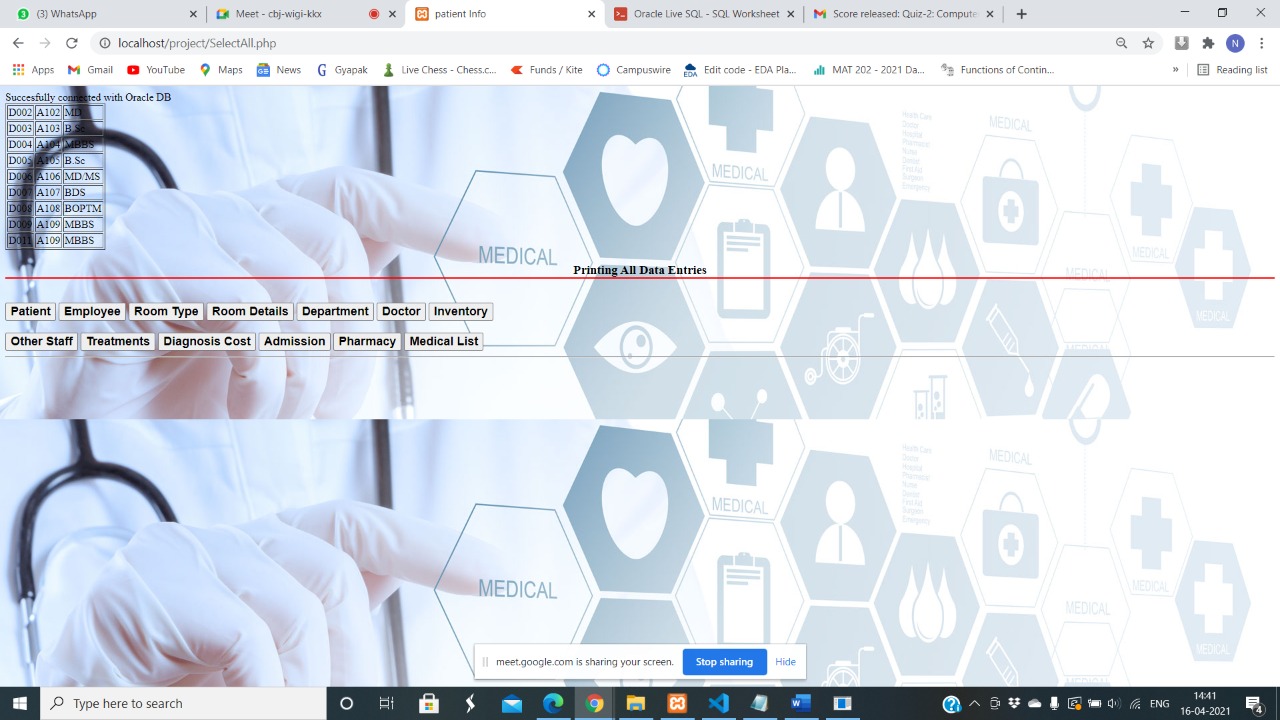
update diagnosis\_cost set dr\_id=null where dr\_id=:old.emp\_id;

end;

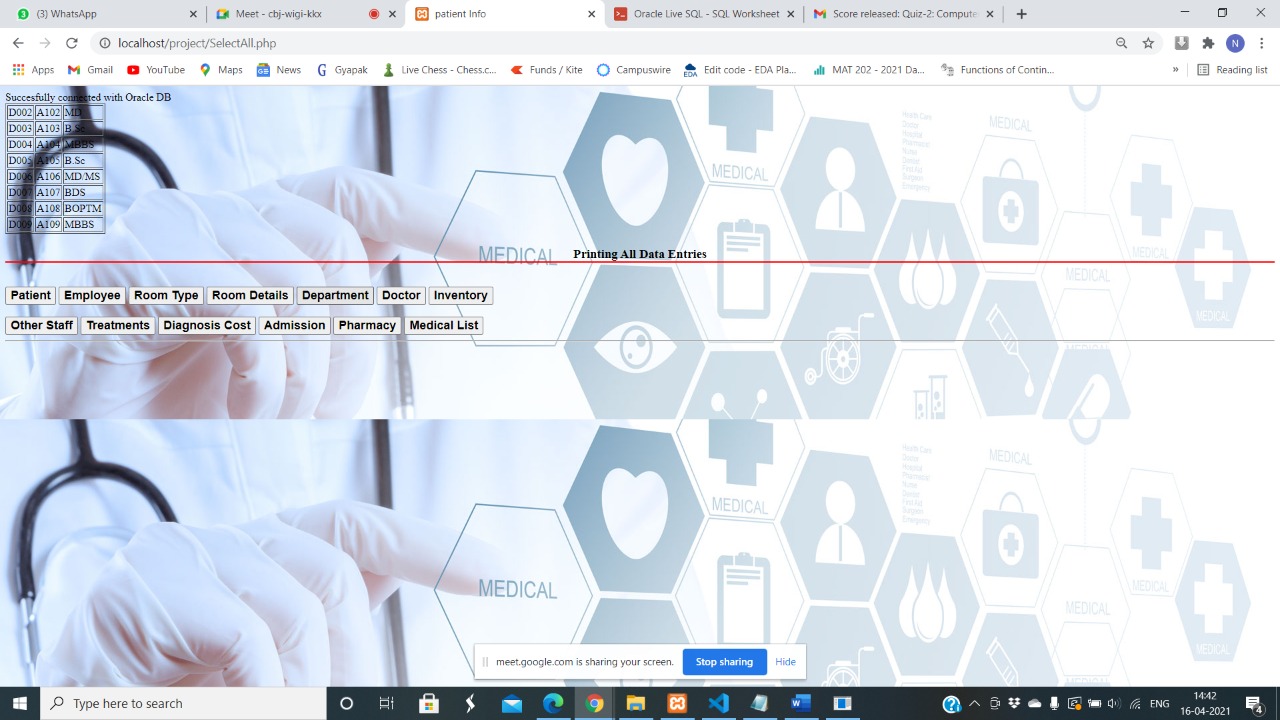
/

**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

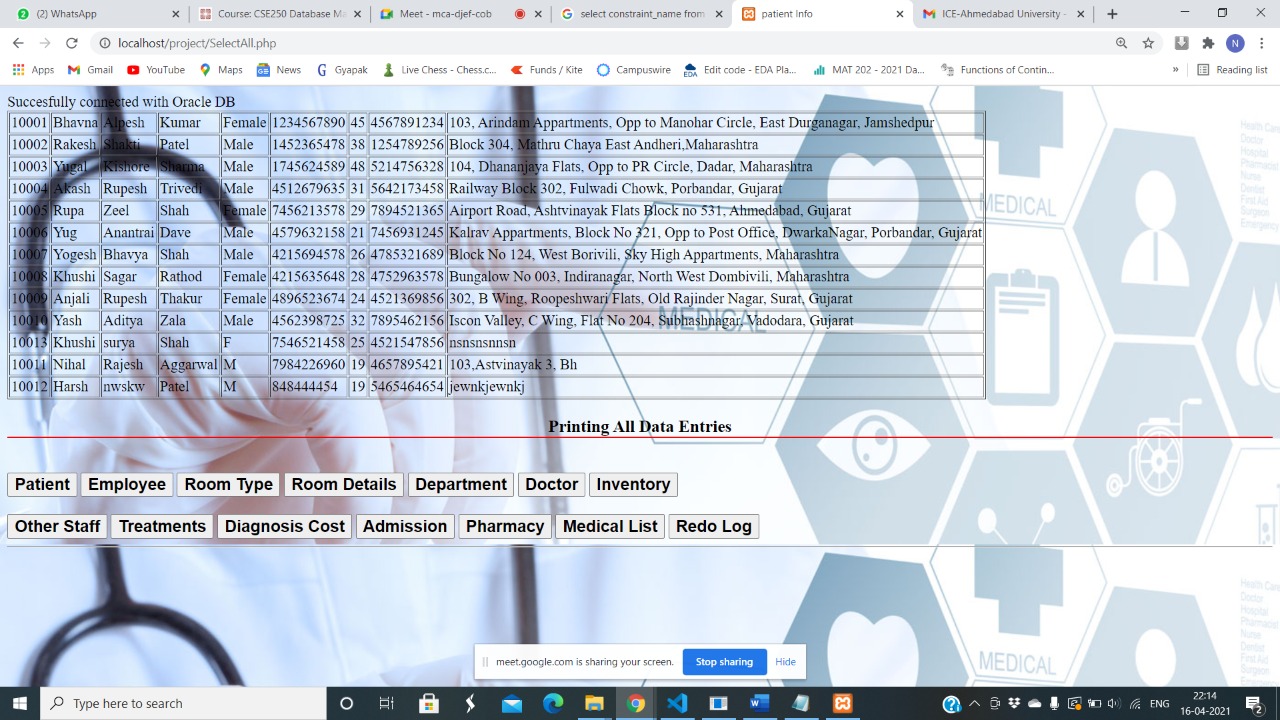
delete from admit where admit.patient\_id=:old.pat\_id;

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;

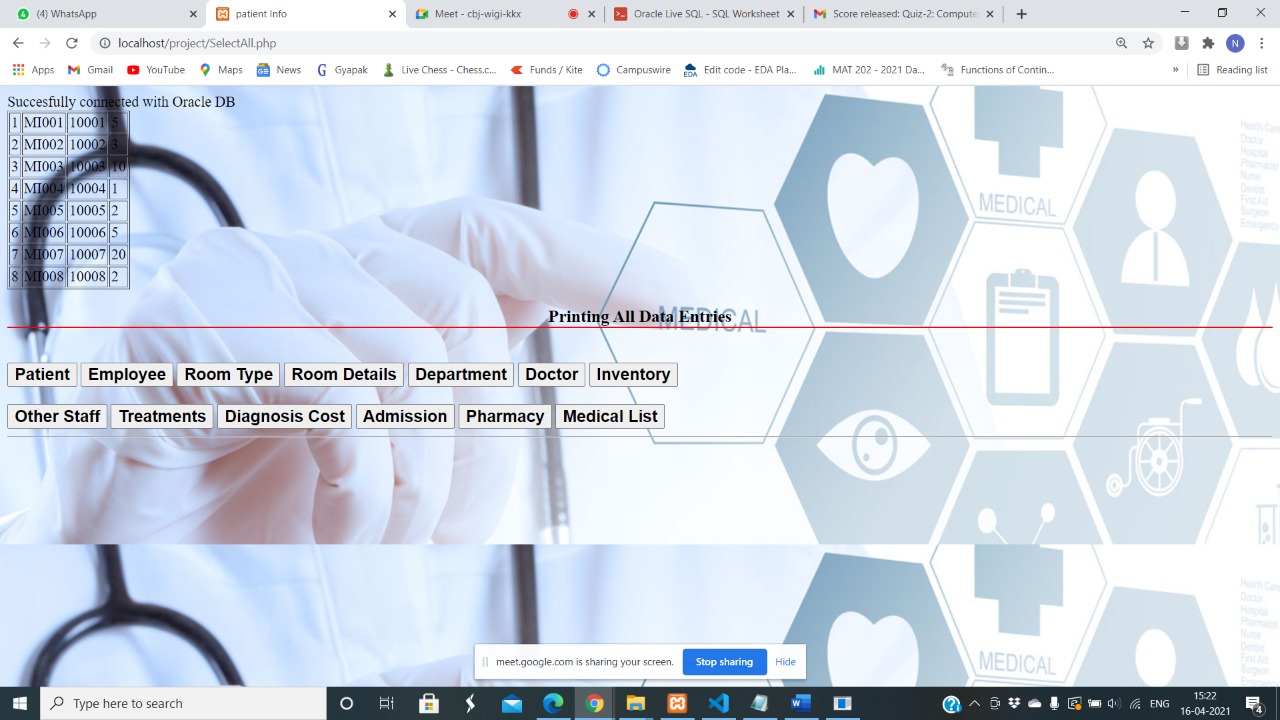
close count;

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');

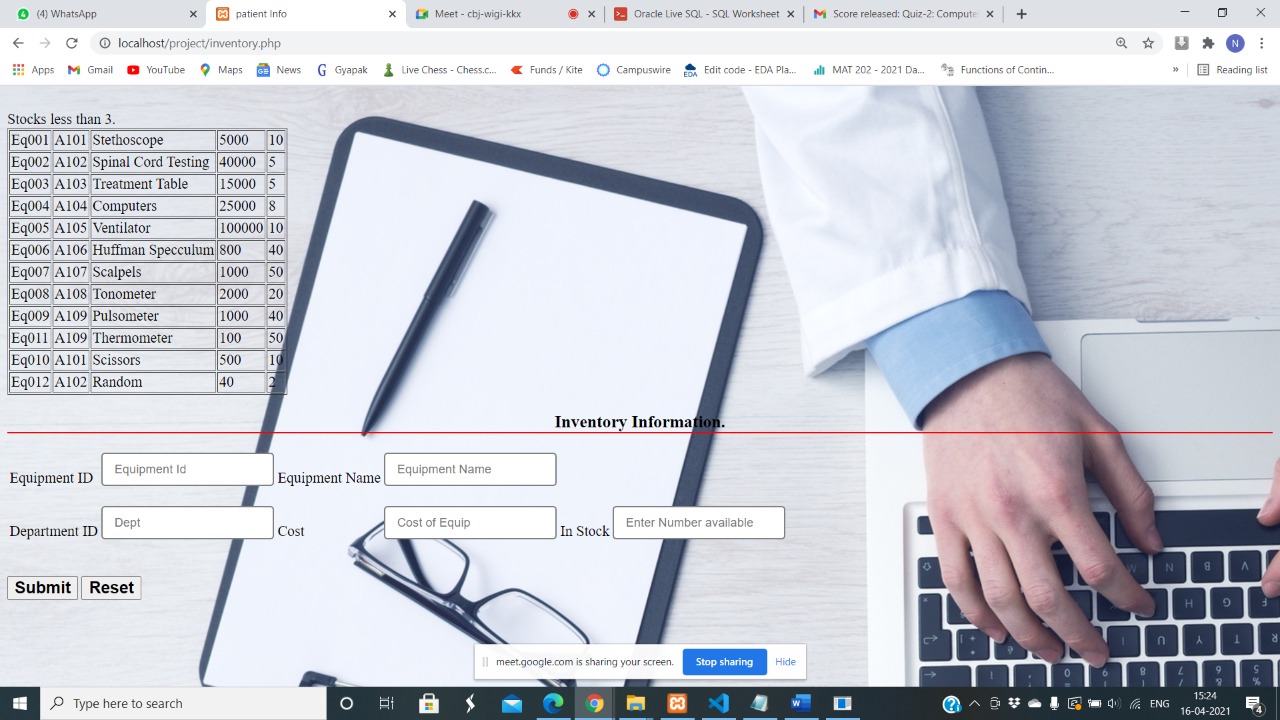
end if;

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

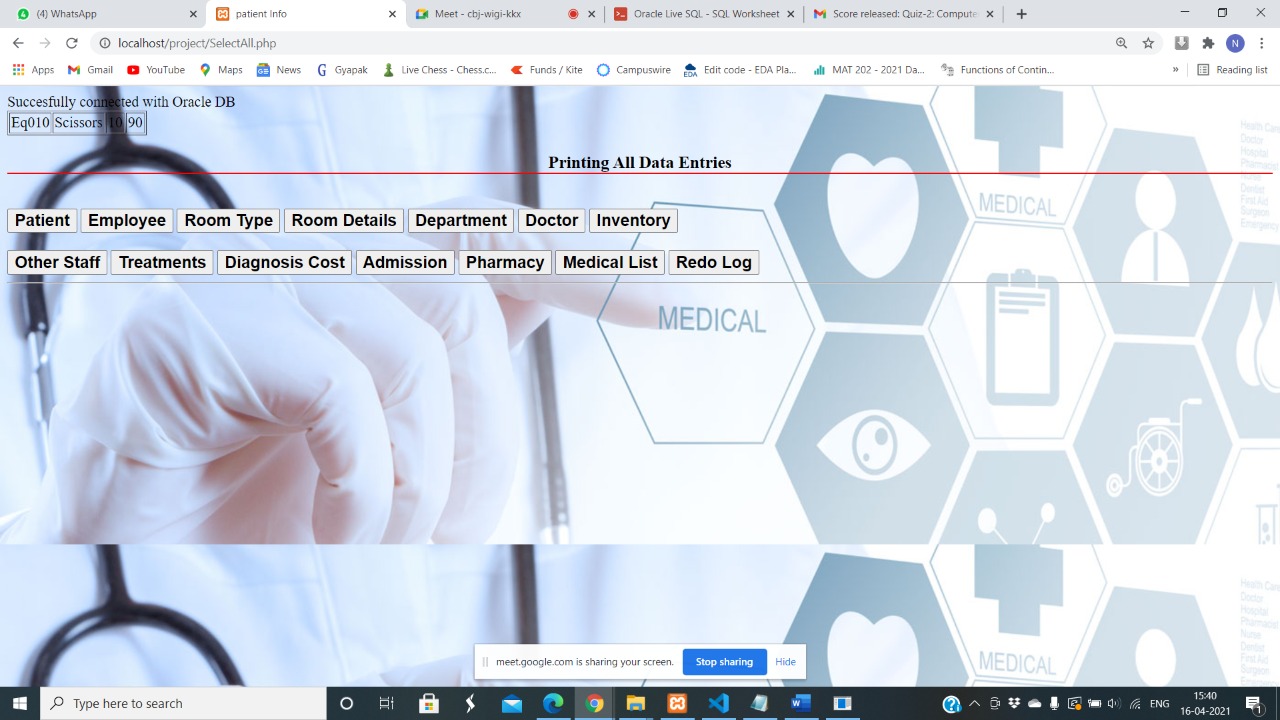
insert into redolog\_values values(:old.equip\_id,:old.equip\_name,:old.stock,:new.stock);

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

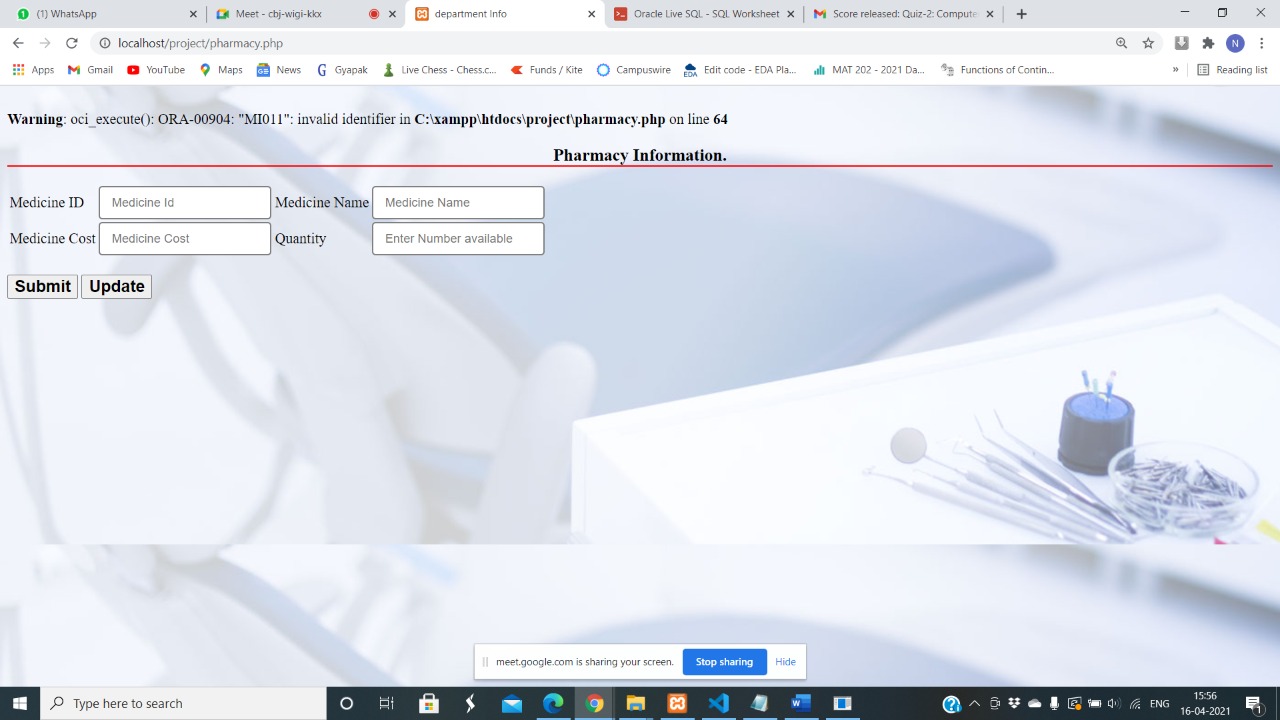
raise\_application\_error(-200016,'Enter a Proper Medicine Name ');

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;

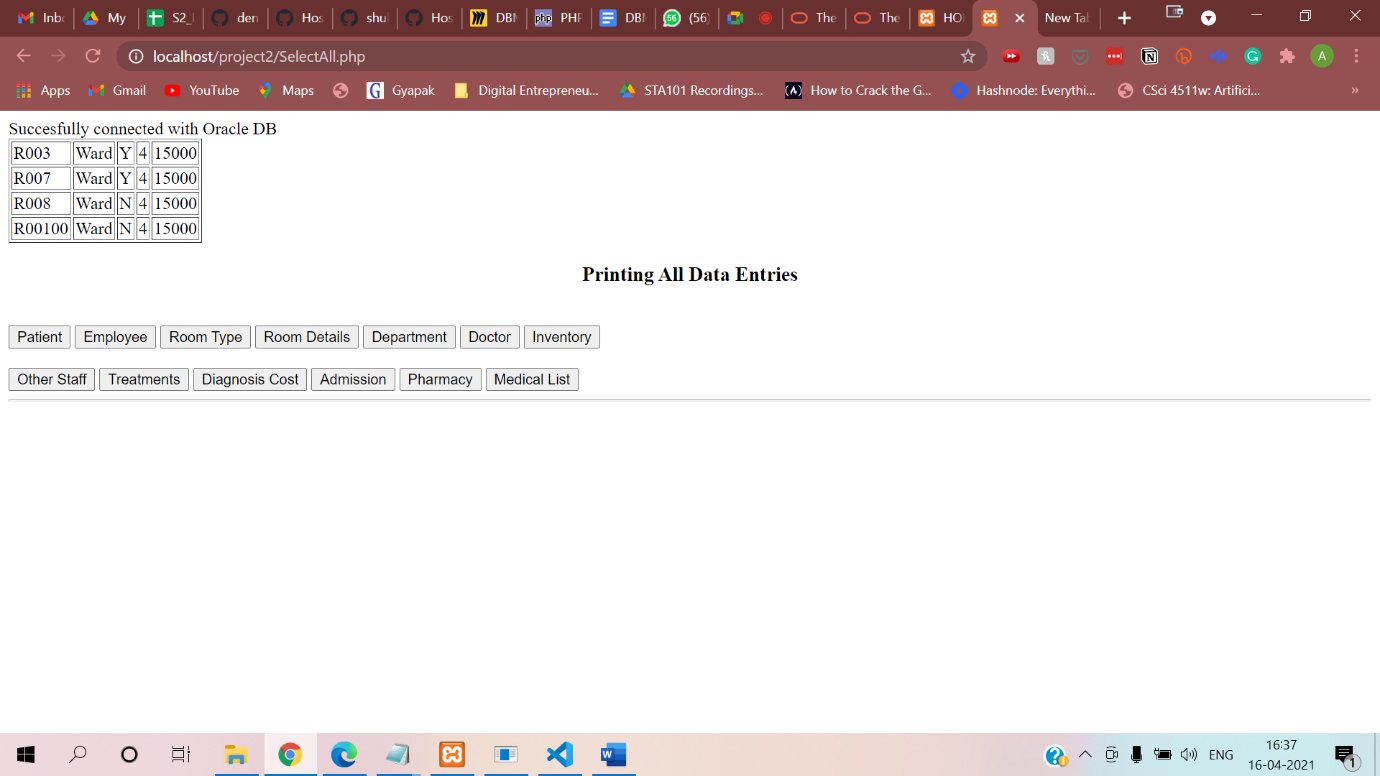
end loop;

close c\_status;

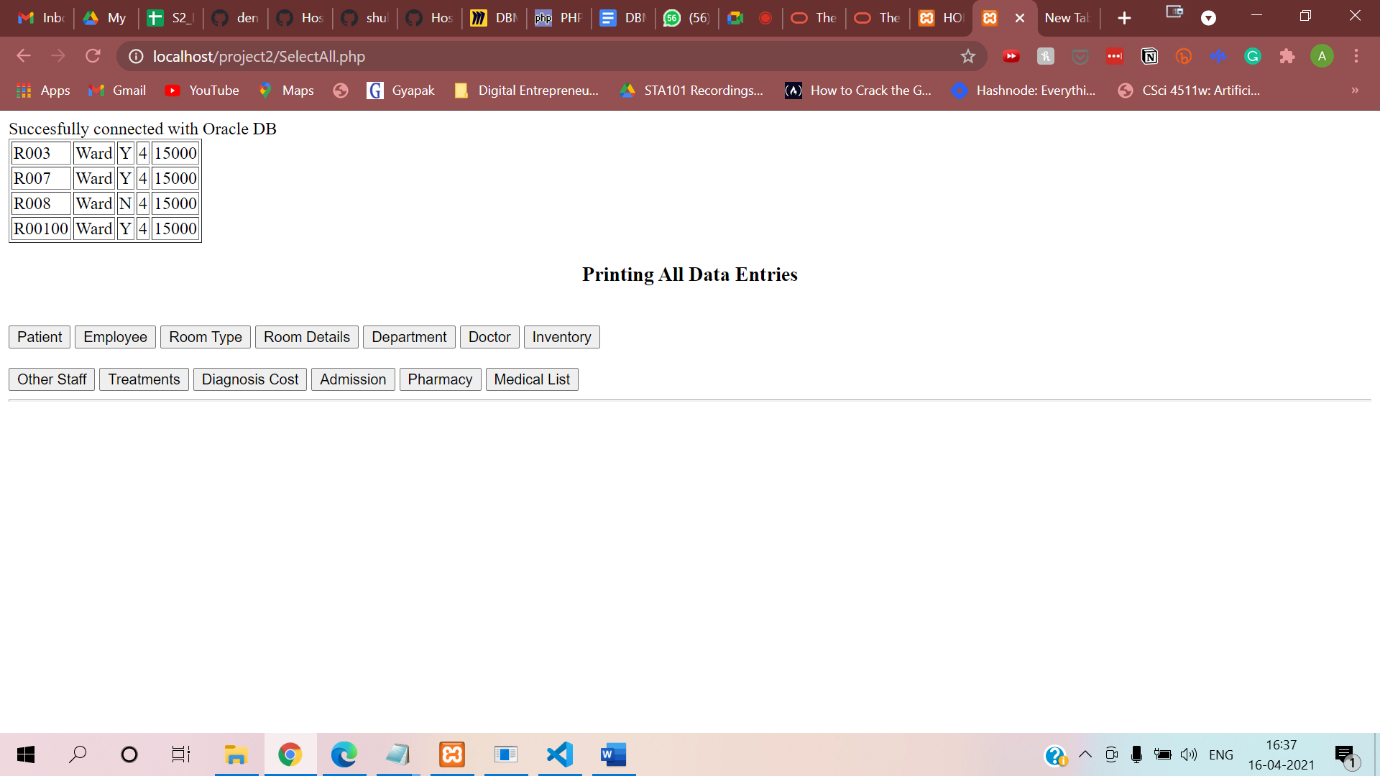
end;

/

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

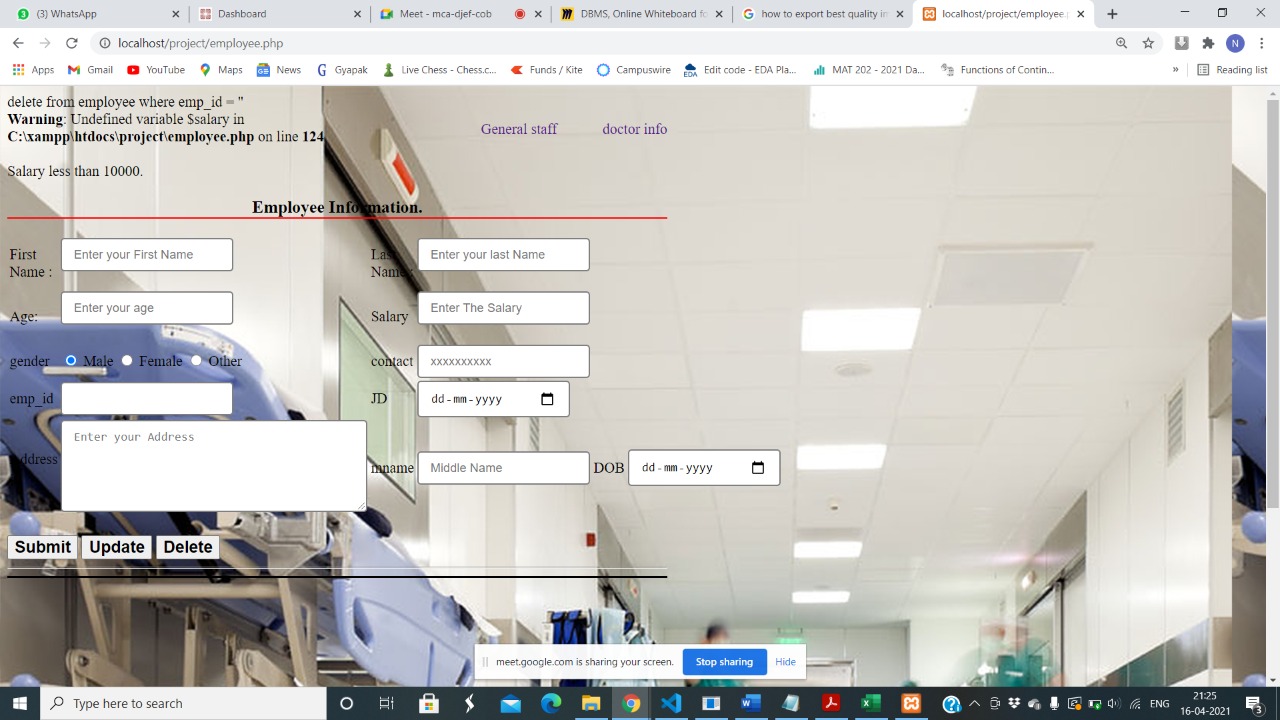
if (:NEW.salary < 10000 ) then

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;

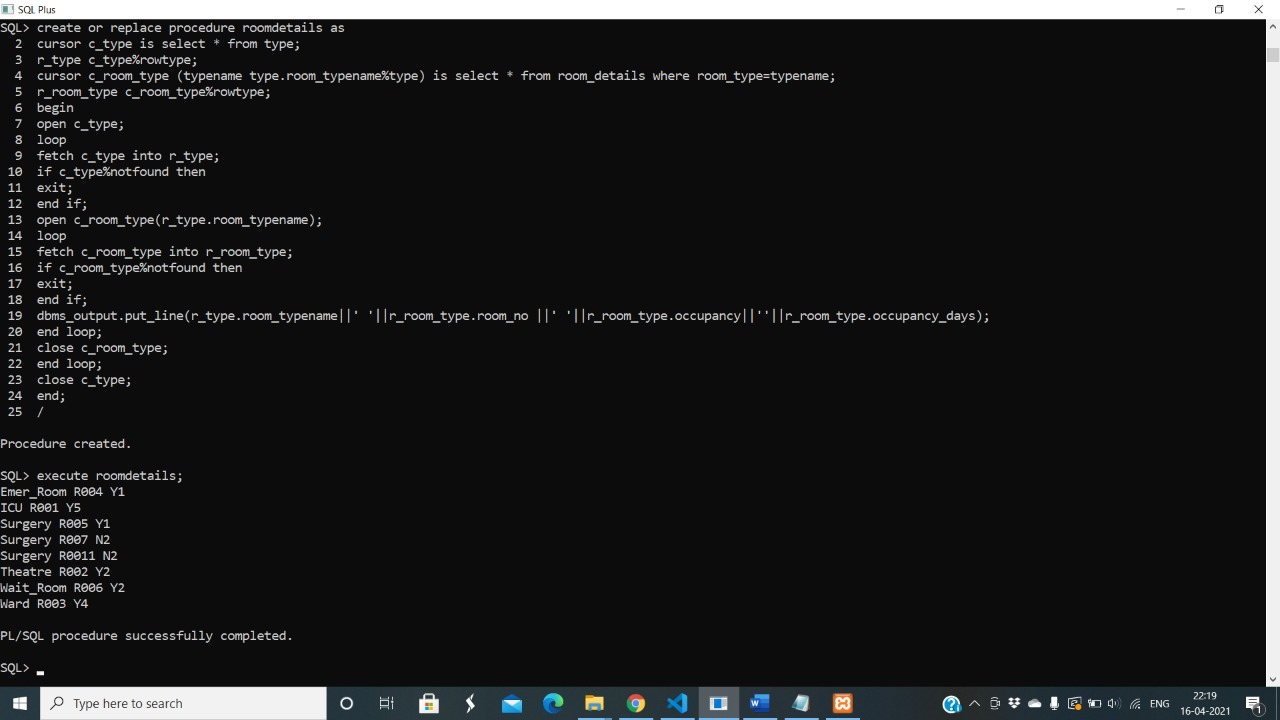
close c\_room\_type;

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;

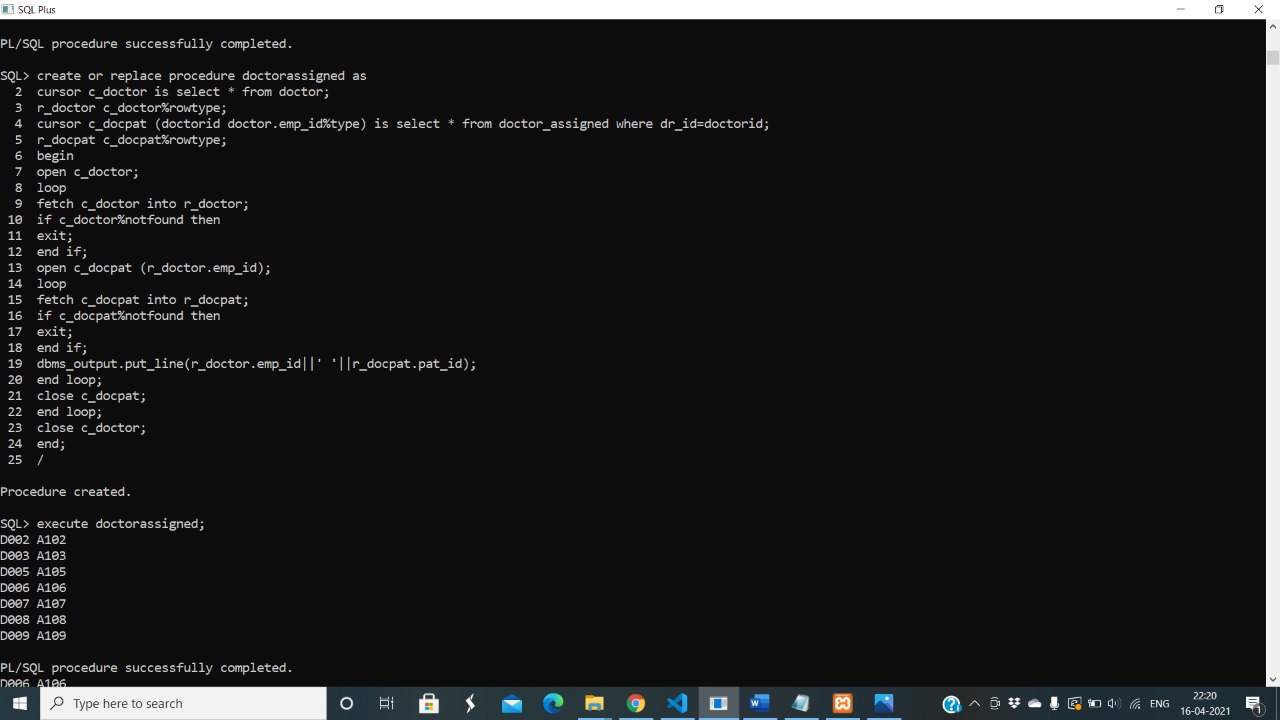
close c\_docpat;

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;

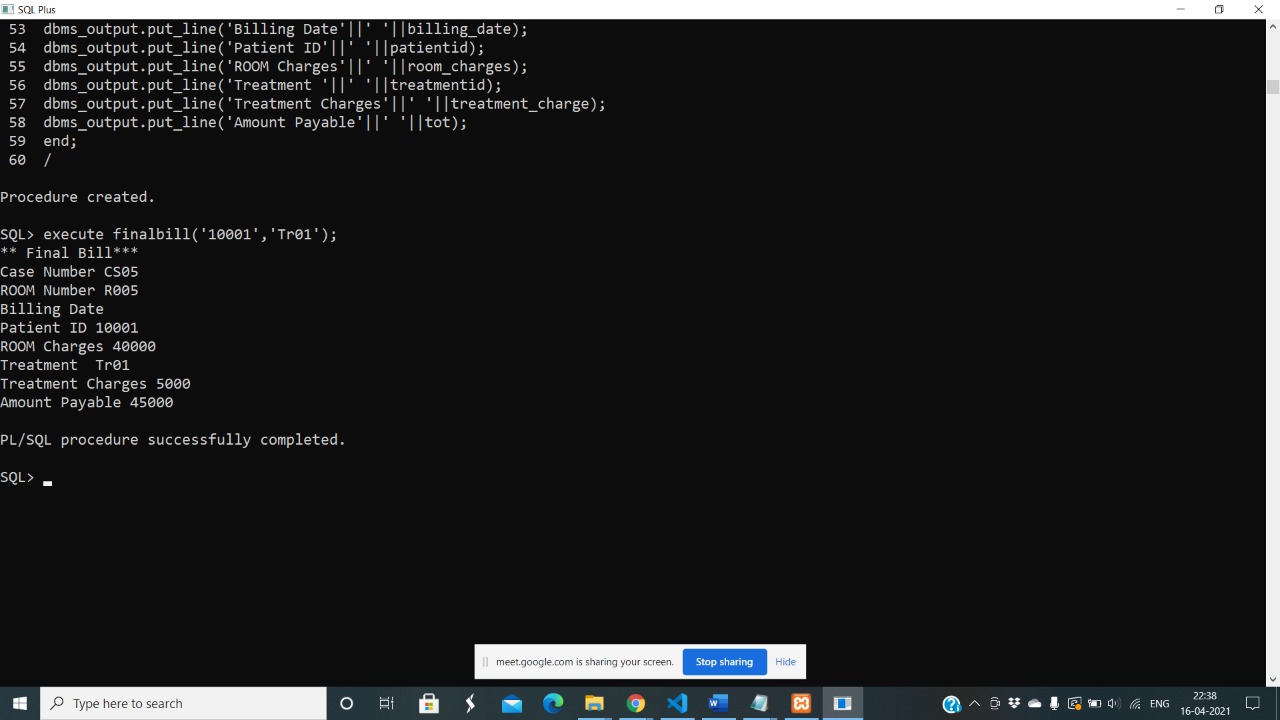
tot:=room\_charges+treatment\_charge;

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;

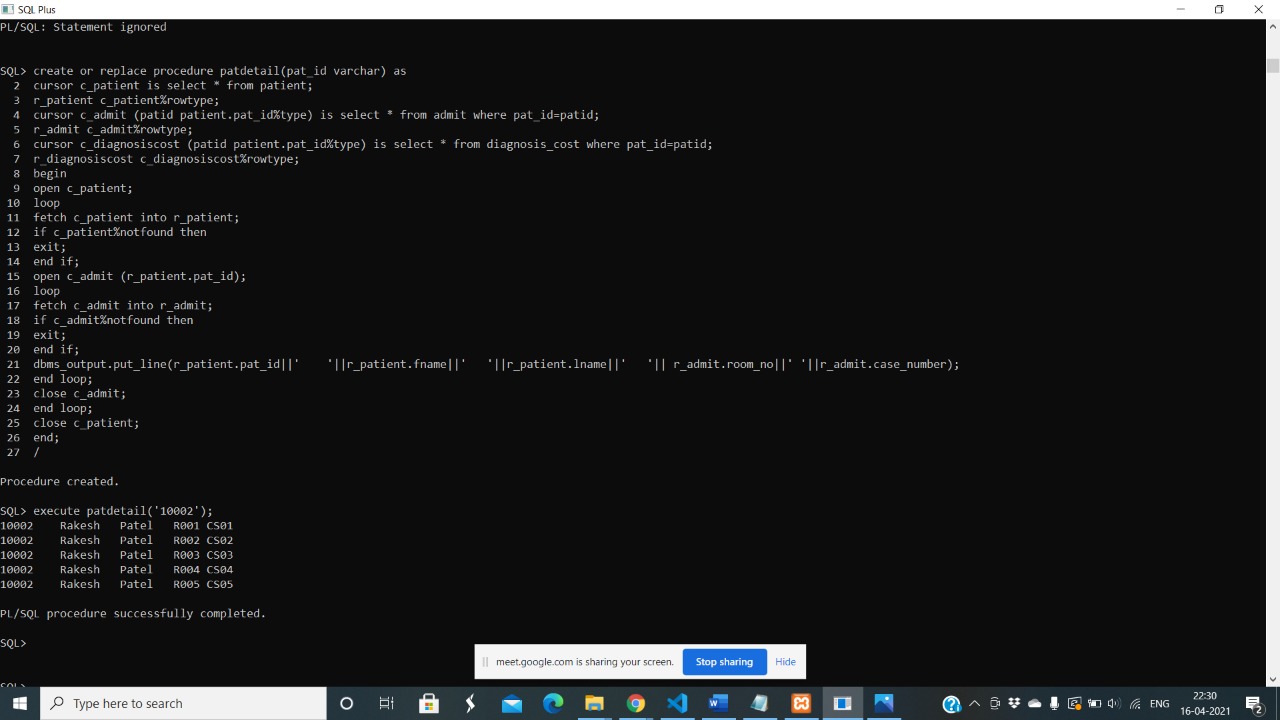
close c\_admit;

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);

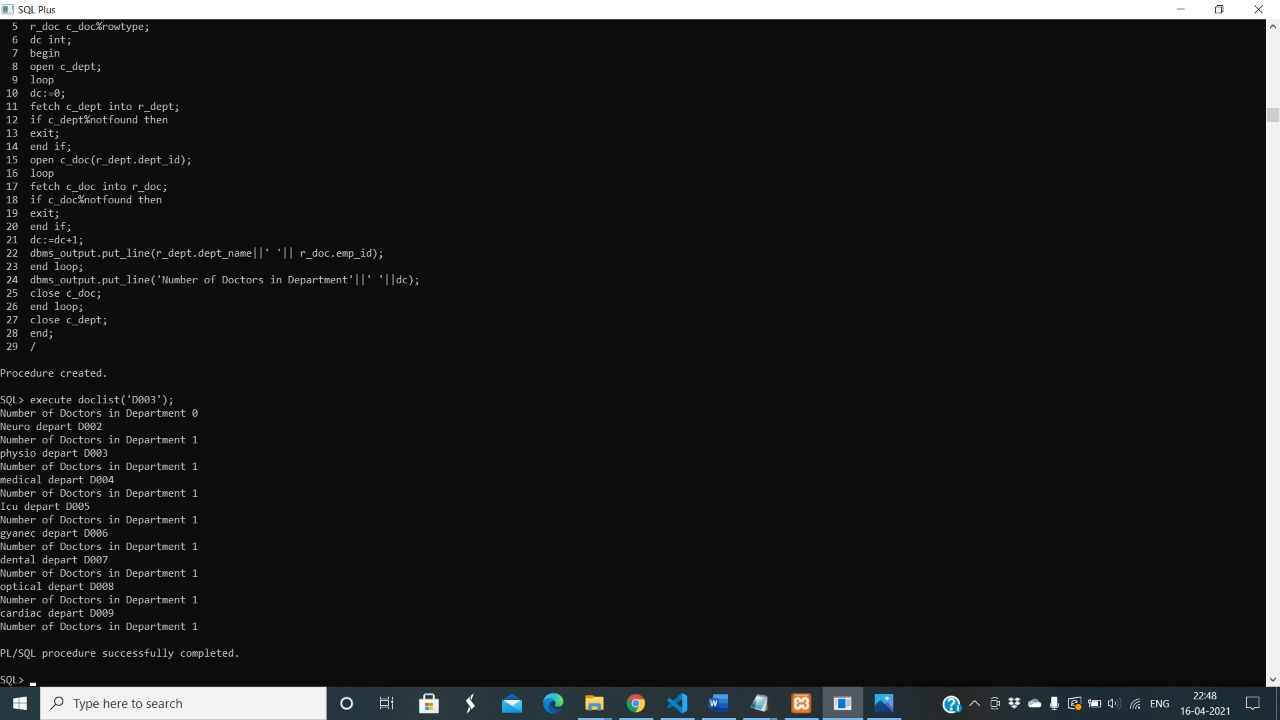
close c\_doc;

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;

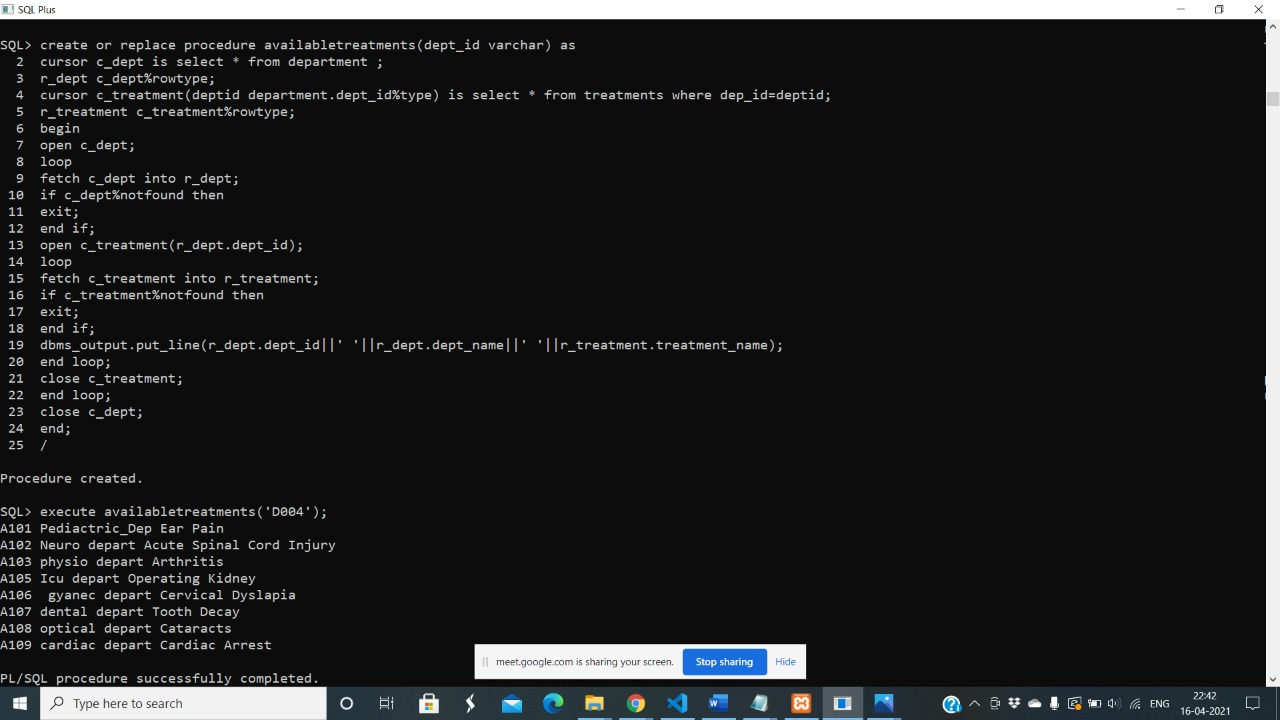
close c\_treatment;

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;

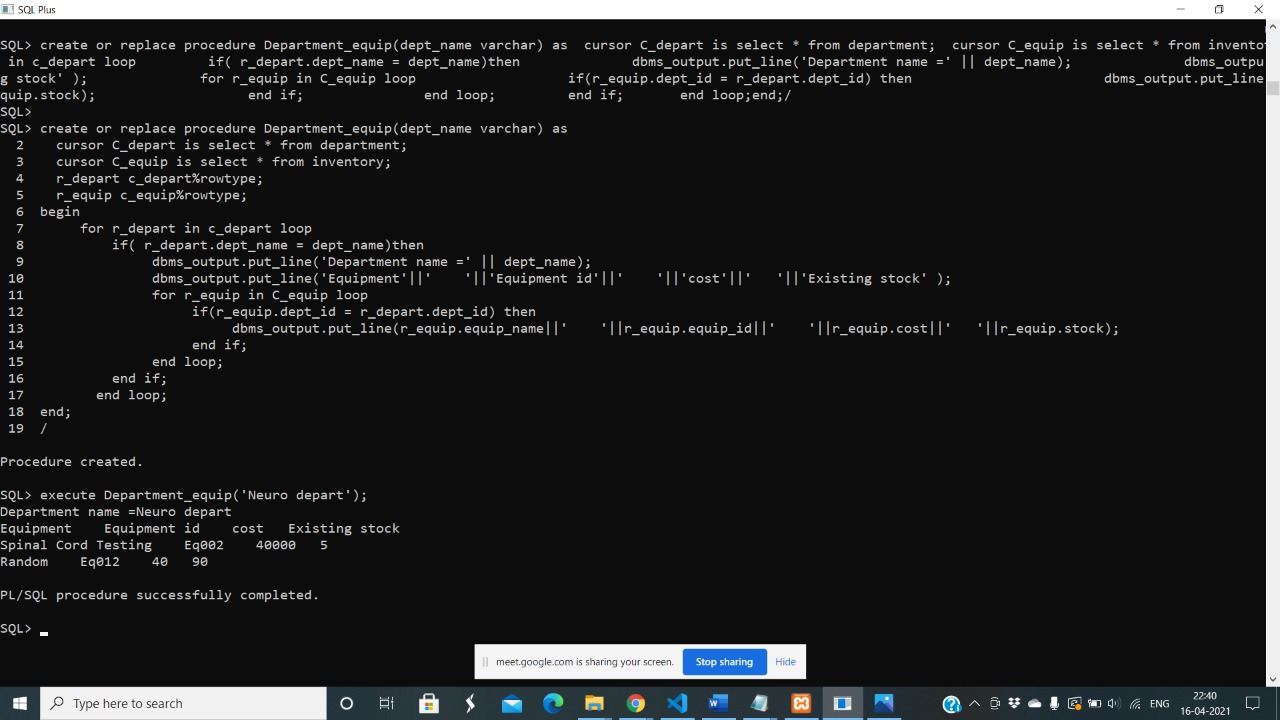
end loop;

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;

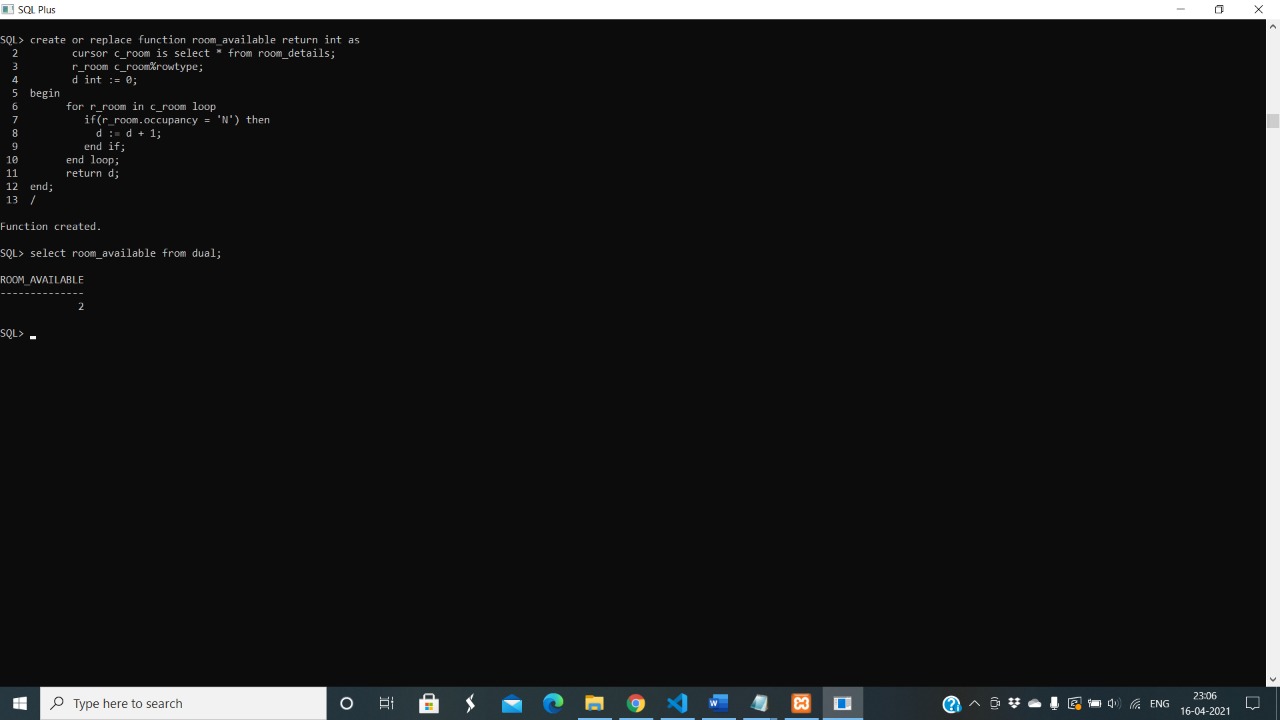
end if;

end loop;

return d;

end;

/



9)

10)