LONDON METROPOLITAN UNIVERSITY

Islington college
(इस्लिङ्टन कलेज)

Module Code & Module Title

CC6001KP Advanced Database Systems Development

Assessment Weightage & Type

25% Individual Coursework

Year and Semester

2018-19 Autumn / 2018-19 Spring

Student Name: Kishor Karki

London Met ID: 18029974

College ID: np01cp4a180171

Assignment Due Date: 22nd Nov. 2019

Assignment Submission Date: 29th Dec. 2019

Title (Where Required):

Word Count (Where Required):

I confirm that I understand my coursework needs to be submitted online via Google Classroom under the relevant module page before the deadline in order for my assignment to be accepted and marked. I am fully aware that late submissions will be treated as non-submission and a marks of zero will be awarded.

Contents

Table of Figures

List of Tables:

1. Introduction of the hospital.

Chirayu hospital which is a private hospital located in basundhara Kathmandu, it is mostly renowned for the best bone specialist hospital and also in various surgical. It is has very peace and healthy environment inside which is also the reason for the people to visit this hospital. The doctors are regularly present for their duties in addition highly qualified are the doctors and nurses. The nurses are good in terms of caring their patient along with on time medication. Now, talking about the aim of this hospital which are as follows:

- To provide good facilities to the patient.
- To be able to add more wards for many patient.
- To provide experienced doctors for the treatment.
- To provide peace and healthy environment in the hospital.

Similarly, some of the objectives of this hospital are as follows:

- To give priorities to the patient in emergency.
- To provide doctors available for 24 hours.
- To provide good security in the hospital.
- To provide comfortable beds and rooms to the patient as well as their care taker.

2. Current Business Activities and Operations.

Basically, in this hospital it gives priorities to each an every patients who have come to the hospital. It keeps the record of the patient, with their addresses along with contact, appointments, treatments and the bills. Whenever a patient visit the hospital, firstly the patient are required to provide their information. Then the patient should fill up their appointment or book the appointment. After that, according to the appointment, the patient is sent to a specific ward provided with a specific doctors. According to the appointment case, the doctors conducts a specific treatments. Then, after go through various steps lastly the patient should pay their bills and if the patient are the staff of the hospital then accordingly charges are given while paying the bills. So, in this way this hospital operates keeping the record of every patient and various other details.

3. Current Business Rules.

- A patient can have multiple address with multiple contacts.
- A patient can have multiple appointments with multiple treatments.
- There is only a ward for a particular appointment.
- There is only a doctor in a ward for a particular appointment.
- A patient pays the bills for the treatment he/she have been through.
- The staff of the hospital can also be the patient.
- The certified staff gets the treatment for free while others must pay their bills.

4. Identification of Entities and Attributes.

Table 1: Entities and Attributes

Entities	Attributes
Patient	Patient_id, Name, Patient_type, Staff_check
Patient_address	Address_id, City, zone, country, street, street no.
Patient_contact	Phone_number, Contact_id, cell_phone, Email, Fax no.
Appointment	Appointment_id, appointment_case, appointment_date
Doctors	Doctor_id, Doctor_name, Doctor_type
Ward	Ward_id, Ward_room, Ward_type
Treatment	Treatment_id, Treatments, Treatment_price
Payment	Payment_id, totalamount_payment, Doctorcharge_payment

5. Initial E-R Diagram.

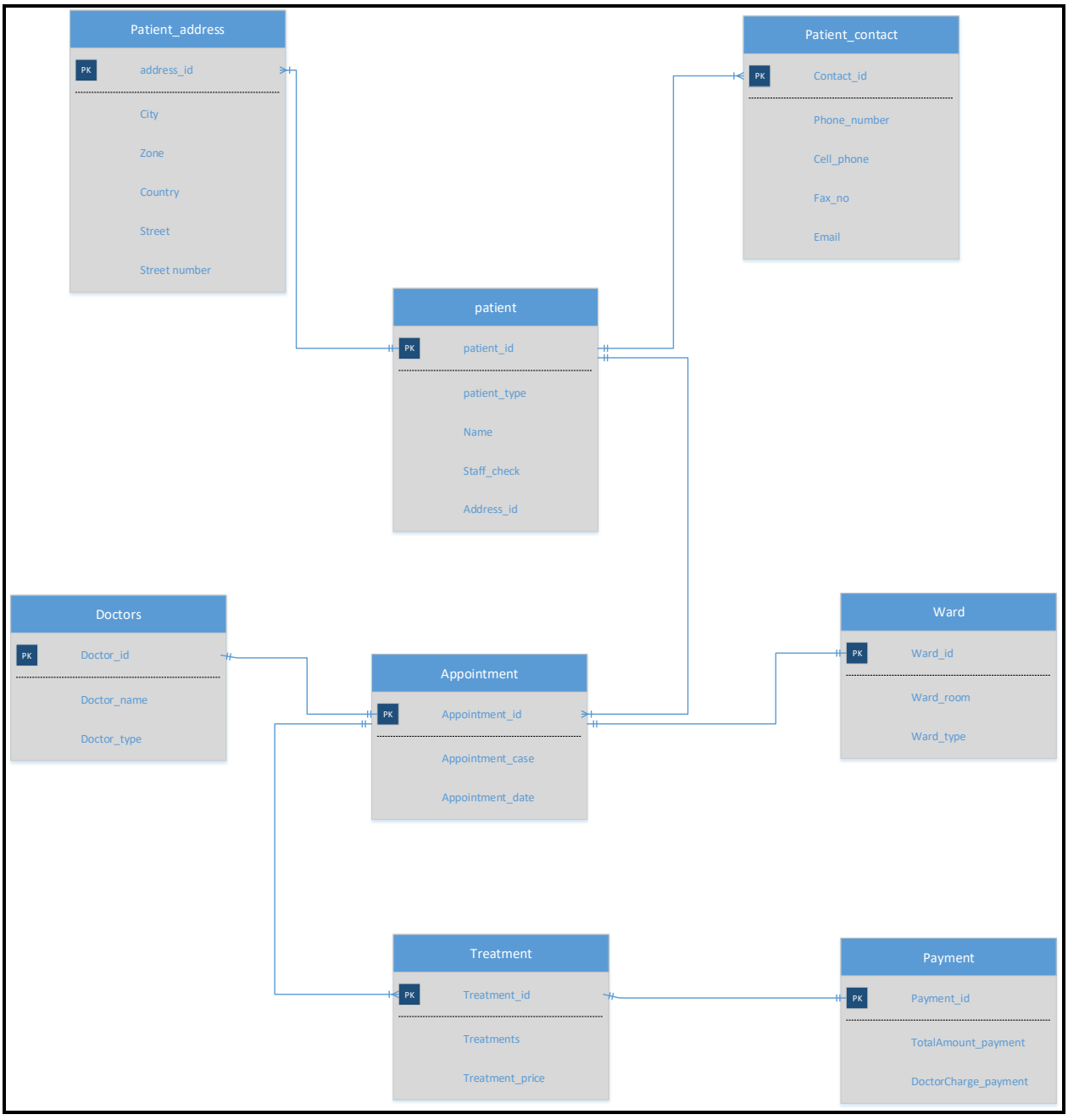

Figure 1: Initial ER Diagram

6. Normalization.

- UNF:

It is the first part of normalization, here, repeating group are kept in curly brackets.

Patient (Patient_id, Name, Patient_type, Staff_check, {Address_id, City, zone, country, street, street no., {Contact_id, Phone_number, cell_phone, Email, Fax no.}}, {Appointment_id, appointment_case, appointment_date, Ward_id, Ward_type, Ward_room, Doctor_id, Doctor_name, Doctor_type {Treatment_id, Treatments, Treatment_price, Payment_id, totalamount_payment, Doctorcharge_payment}})

- UNF to 1NF:

In order to be in 1NF the repeating group should be removed.

Patient_1 (Patient_id, Name, Patient_type, Staff_check)

Patient_address_1 (Patient_id*, Address_id, City, zone, country, street, street no.)

Patient_address_contact_1 (Patient_id*, Address_id*, Contact_id, Phone_number, cell_phone, Email, Fax no.)

Patient_appointment_1 (Patient_id*, Appointment_id, appointment_case, appointment_date, Ward_id, Ward_type, Ward_room, Doctor_id, Doctor_name, Doctor_type)

Patient_appoint_treatment_1 (Patient_id*, Appointment_id*, Treatment_id, Treatments, Treatment_price, Payment_id, totalamount_payment, Doctorcharge_payment)

a. 1NF to 2NF:

The repeating groups are removed now the data is in 1NF. The data can be normalized to 2NF by removing partial dependency.

Patient_1 (Patient_id, Name, Patient_type, Staff_check)

It is already in 2NF as it has only one candidate key and there is partial dependency.

Patient_address_1 (Patient_id*, Address_id, City, zone, country, street, street no.)

Here, candidate keys: Patient_id, Address_id

Non-key attributes: City, zone, country, street, street no.

Patient_id = X

Address_id = City, zone, country, street, street no.

Patient_id, Address_id = X

Patient_address_contact_1 (Patient_id*, Address_id*, Contact_id, Phone_number, cell_phone, Email, Fax no.)

Here, candidate keys: Patient_id, Contact_id, Address_id

Non-key attributes: Phone_number, cell_phone, Email, Fax no.

Patient_id = X

Contact_id = Phone_number, cell_phone, Email, Fax no.

Patient_id, Contact_id = X

Patient_id, Contact_id, Address_id =

Patient_id, Address_id= X

Contact_id, Address_id = X

Patient_appointment_1 (Patient_id*, Appointment_id, appointment_case, appointment_date, Ward_id, Ward_type, Ward_room, Doctor_id, Doctor_name, Doctor_type)

Here, candidate keys: Patient_id, Appointment_id, Ward_id, Doctor_id,

Non-key attributes: appointment_case, appointment_date, Ward_type, Ward_room, Doctor_name, Doctor_type

Patient_id = X

Appointment_id = appointment_case, appointment_date

Ward_id = Ward_type, Ward_room

Doctor_id = Doctor_name, Doctor_type

Patient_id, Appointment_id, Ward_id, Doctor_id =

Patient_id, Ward_id = X

Patient_id, Doctor_id = X

Ward_id, Appointment_id = X

Doctor_id, Appointment_id = X

Doctor_id, Ward_id = X

Patient_appoint_treatment_1 (Patient_id*, Appointment_id*, Treatment_id, Treatments, Treatment_price, Payment_id, totalamount_payment, Doctorcharge_payment)

Here, candidate keys: Patient_id, Appointment_id, Treatment_id, Payment_id

Non-key attributes: Treatments, Treatment_price, totalamount_payment, Doctorcharge_payment

Patient_id = X

Appointment_id = X

Treatment_id = Treatments, Treatment_price

Payment_id = totalamount_payment, Doctorcharge_payment

Patient_id, Appointment_id, Treatment_id, Payment_id =

Patient_id, Appointment_id = X

Patient_id, Treatment_id = X

Patient_id, Payment_id = X

Treatment_id, Appointment_id = X

Payment_id, Appointment_id = X

Treatment_id, Payment_id = X

Hence, the following tables are in 2NF form:

Patient _2 (Patient_id, Name, Patient_type, Staff_check)

Patient_address_2 (Address_id, City, zone, country, street, street no.)

Patient_contact_2 (Contact_id, Phone_number, cell_phone, Email, Fax no.)

Patient_detail_2 (Patient_id*, Patient_address*, Patient_contact*)

Appointment_2 (Appointment_id, appointment_case, appointment_date)

Ward_2 (Ward_id, Ward_type, Ward_room)

Doctors_2 (Doctor_id, Doctor_name, Doctor_type)

Appointment_detail_2 (Patient_id*, Appointment_id*, Ward_id*, Doctor_id*)

Treatment_2 (Treatment_id, Treatments, Treatment_price)

Payment_2 (Payment_id, totalamount_payment, Doctorcharge_payment)

Treatment_detail_2 (Patient_id*, Appointment_id*, Treatment_id*, Payment_id*)

b. 2NF to 3NF:

There is no transitive dependency in Patient, Patient_address, Patient_contact, Appointment, Treatment, Appointment_detail, Treatment_detail and payment.

Patient _3 (Patient_id, Name, Patient_type, Staff_check)

Patient_address_3 (Address_id, City, zone, country, street, street no.)

Patient_contact_3 (Contact_id, Phone_number, cell_phone, Email, Fax no.)

Patient_detail_3 (Patient_id*, Patient_address*, Patient_contact*)

Appointment_3 (Appointment_id, appointment_case, appointment_date)

Ward_3 (Ward_id, Ward_type, Ward_room)

Doctors_3 (Doctor_id, Doctor_name, Doctor_type)

Appointment_detail_3 (Patient_id*, Appointment_id*, Ward_id*, Doctor_id*)

Treatment_3 (Treatment_id, Treatments, Treatment_price)

Payment_3 (Payment_id, totalamount_payment, Doctorcharge_payment)

Treatment_detail_3 (Patient_id*, Appointment_id*, Treatment_id*, Payment_id*)

7. Final E-R Diagram.

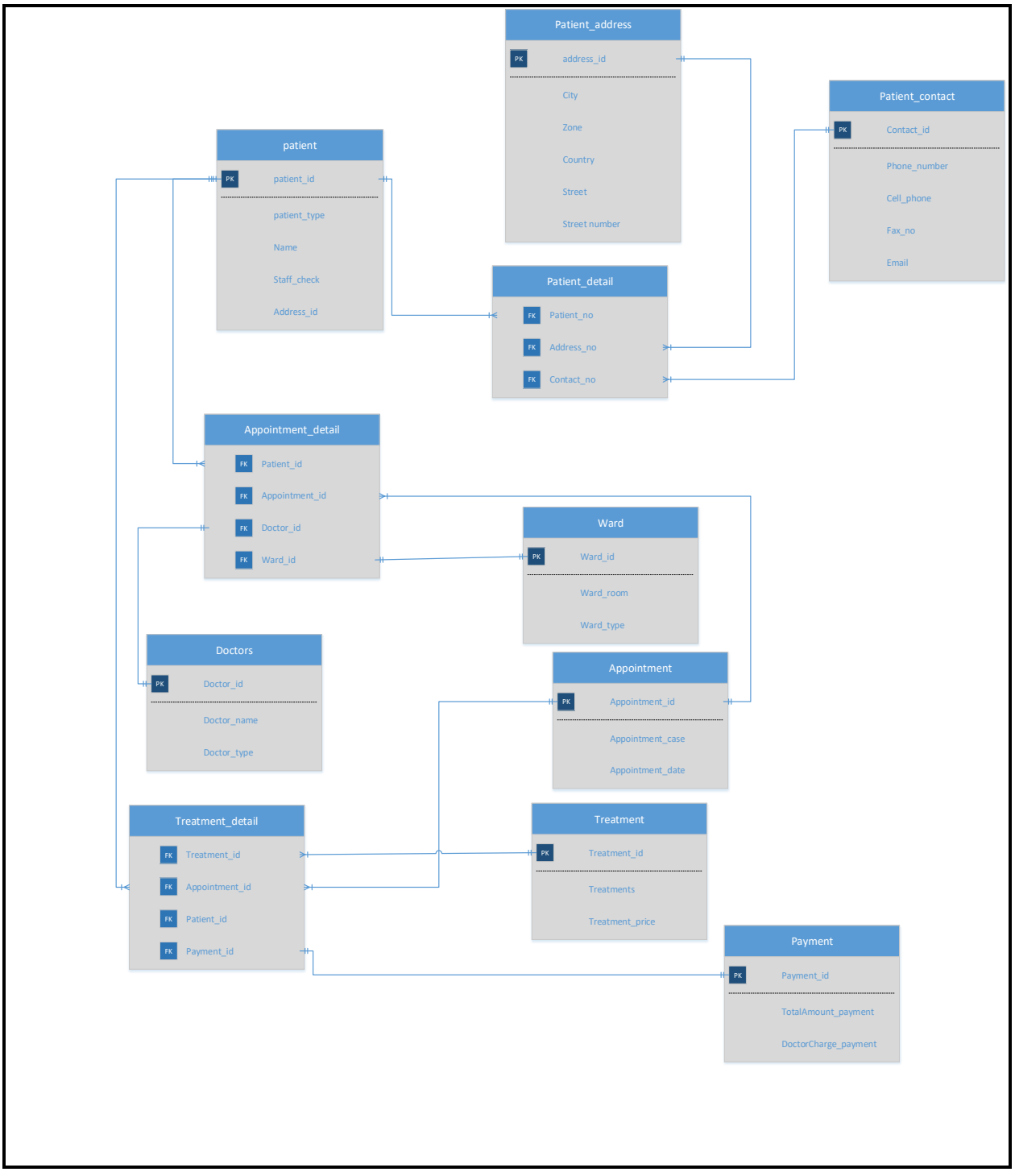

Figure 2: Final ER Diagram.

So this is the final ER diagram where it clearly show how the records of the patient is done and stored. This is the normalized ER-diagram in which patient detail are

separately kept and linked with other entities, and in the same way, appointment, treatment, payment, address and contact as well. Hence, now it easy to find the patient records.

8. Database Implementation.

a. Tables Generation (DDL Scripts).

Create table Patient (Patient_id number not null, Name varchar2(25) not null, Patient_type varchar2(20) not null, Staff_check varchar2(20) not null, Staff_type varchar2(20), constraint p_pk primary key(Patient_id));

```
SQL> create table Patient (Patient_id number not null, Name varchar2(25) not null, Patient_type varchar2
(20) not null, Staff_check varchar2(20) not null, Staff_type varchar2(20), constraint p_pk primary key(P
atient_id));

Table created.

SQL> desc patient;
 Name                                              Null?    Type
 ------------------------------------------------- -------- ------------------------------------
 PATIENT_ID                                        NOT NULL NUMBER
 NAME                                              NOT NULL VARCHAR2(25)
 PATIENT_TYPE                                      NOT NULL VARCHAR2(20)
 STAFF_CHECK                                       NOT NULL VARCHAR2(20)
 STAFF_TYPE                                                 VARCHAR2(20)
```

Figure 3: Creating Table Patient.

Create table Patient_address (Address_id number not null, City varchar2(20) not null, Zone varchar2(20) not null, Country varchar2(20) not null, Street varchar2(20) not null, Street_number number not null, constraint a_pk primary key(Address_id));

```
SQL> create table Patient_address (Address_id number not null, City varchar2(20) not null, Zone varchar2
(20) not null, Country varchar2(20) not null, Street varchar2(20) not null, Street_number number not nul
l, constraint a_pk primary key(Address_id));

Table created.

SQL> desc patient_address;
 Name                                              Null?    Type
 ------------------------------------------------- -------- ------------------------------------
 ADDRESS_ID                                        NOT NULL NUMBER
 CITY                                              NOT NULL VARCHAR2(20)
 ZONE                                              NOT NULL VARCHAR2(20)
 COUNTRY                                           NOT NULL VARCHAR2(20)
 STREET                                            NOT NULL VARCHAR2(20)
 STREET_NUMBER                                     NOT NULL NUMBER
```

Figure 4: Creating Table Patient_address.

Create table Patient_contact(Contact_id number not null, Phone_number number not null, Cell_phone varchar2(20), Fax_no varchar2(20), Email varchar2(50) not null, constraint c_pk primary key (Contact_id));

```
SQL> create table Patient_contact(Contact_id number not null, Phone_number number not null, Cell_phone v
archar2(20), Fax_no varchar2(20), Email varchar2(50) not null, constraint c_pk primary key (Contact_id))
;

Table created.

SQL> desc patient_contact;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 CONTACT_ID                                NOT NULL NUMBER
 PHONE_NUMBER                              NOT NULL NUMBER
 CELL_PHONE                                         VARCHAR2(20)
 FAX_NO                                             VARCHAR2(20)
 EMAIL                                     NOT NULL VARCHAR2(50)
```

Figure 5: Creating table Patient_contact.

Create table Patient_detail (Patient_no number not null, constraint pp_fk foreign key (patient_no) references patient (patient_id) on delete cascade, Address_no number not null, constraint aa_fk foreign key (Address_no) references patient_address(address_id) on delete cascade, Contact_no number not null, constraint cc_fk foreign key (contact_no) references patient_contact (contact_id) on delete cascade);

```
SQL> create table Patient_detail (Patient_no number not null, constraint pp_fk foreign key (patient_no)
references patient (patient_id) on delete cascade, Address_no number not null, constraint aa_fk foreign
key (Address_no) references patient_address(address_id) on delete cascade, Contact_no number not null, c
onstraint cc_fk foreign key (contact_no) references patient_contact (contact_id) on delete cascade);

Table created.

SQL> desc patient_detail;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 PATIENT_NO                                NOT NULL NUMBER
 ADDRESS_NO                                NOT NULL NUMBER
 CONTACT_NO                                NOT NULL NUMBER

SQL>
```

Figure 6: Creating Table Patient_detail.

Create table Doctors (Doctor_id number not null, Doctor_name varchar2(25) not null, Doctor_type varchar2(20) not null, constraint s_pk primary key (Doctor_id));

```
SQL> create table Doctors (Doctor_id number not null, Doctor_name varchar2(25) not null, Doctor_type var
char2(20) not null, constraint s_pk primary key (Doctor_id));

Table created.

SQL> desc doctors;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 DOCTOR_ID                                 NOT NULL NUMBER
 DOCTOR_NAME                               NOT NULL VARCHAR2(25)
 DOCTOR_TYPE                               NOT NULL VARCHAR2(20)
```

Figure 7: Creating Table Doctors.

Create table Ward(Ward_id number not null, Ward_room varchar2(20) not null, Ward_type varchar2(20) not null, constraint w_pk primary key (Ward_id));

```
SQL> create table Ward(Ward_id number not null, Ward_room varchar2(20) not null, Ward_type varchar2(20)
not null, constraint w_pk primary key (Ward_id));

Table created.

SQL> desc ward;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 WARD_ID                                   NOT NULL NUMBER
 WARD_ROOM                                 NOT NULL VARCHAR2(20)
 WARD_TYPE                                 NOT NULL VARCHAR2(20)

SQL>
```

Figure 8: Creating Table Ward.

Create table Appointment(Appointment_id number not null, constraint ap_pk primary key(appointment_id), Appointment_case varchar2(20) not null, Appointment_date date not null);

```
SQL> Create table Appointment(Appointment_id number not null, constraint ap_pk primary key(appointment_i
d), Appointment_case varchar2(20) not null, Appointment_date date not null);

Table created.

SQL> desc appointment;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 APPOINTMENT_ID                            NOT NULL NUMBER
 APPOINTMENT_CASE                          NOT NULL VARCHAR2(20)
 APPOINTMENT_DATE                          NOT NULL DATE
```

Figure 9: Creating Table Appointment.

Create table Appointment_detail (Patient_id number not null, Appointment_id number not null, Doctor_id number not null, Ward_id number not null, constraint patt_fk foreign key (patient_id) references patient (patient_id) on delete cascade, constraint app_fk foreign key (appointment_id) references appointment (appointment_id) on delete cascade, constraint stt_fk foreign key (Doctor_id) references Doctors(Doctor_id) on delete cascade, constraint wad_fk foreign key (ward_id) references ward (ward_id) on delete cascade);

```
SQL> create table Appointment_detail (Patient_id number not null, Appointment_id number not null, Doctor
_id number not null, Ward_id number not null, constraint patt_fk foreign key (patient_id) references pat
ient (patient_id) on delete cascade, constraint app_fk foreign key (appointment_id) references appointme
nt (appointment_id) on delete cascade, constraint stt_fk foreign key (Doctor_id) references Doctors(Doct
or_id) on delete cascade, constraint wad_fk foreign key (ward_id) references ward (ward_id) on delete ca
scade);

Table created.

SQL> desc appointment_detail;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 PATIENT_ID                                NOT NULL NUMBER
 APPOINTMENT_ID                            NOT NULL NUMBER
 DOCTOR_ID                                 NOT NULL NUMBER
 WARD_ID                                   NOT NULL NUMBER

SQL>
```

Figure 10: Creating Table Appointment_detail.

Create table Payment (payment_id number not null, constraint pp_pk primary key(payment_id), totalamount_payment Varchar2(20), Doctorcharge_payment varchar2(20));

```
SQL> create table Payment (payment_id number not null, constraint pp_pk primary key(payment_id), totalamo
unt_payment Varchar2(20), Doctorcharge_payment varchar2(20));

Table created.

SQL> desc payment;
 Name                                              Null?    Type
 ------------------------------------------------- -------- ------------------------------------
 PAYMENT_ID                                        NOT NULL NUMBER
 TOTALAMOUNT_PAYMENT                                        VARCHAR2(20)
 DOCTORCHARGE_PAYMENT                                       VARCHAR2(20)

SQL>
```

Figure 11: Creating Table Payment.

Create table Treatment_detail (Patient_id number not null, Appointment_id number not null, Treatment_id number not null, Payment_id number not null, constraint ppp_fk foreign key (patient_id) references patient(patient_id) on delete cascade, constraint apt_fk foreign key (appointment_id) references appointment(appointment_id)on delete cascade, constraint tr_fk foreign key (treatment_id) references treatment (treatment_id) on delete cascade, constraint pll_fk foreign key (payment_id) references payment(payment_id) on delete cascade);

```
SQL> create table Treatment_detail (Patient_id number not null, Appointment_id number not null, Treatmen
t_id number not null, Payment_id number not null, constraint ppp_fk foreign key (patient_id) references
patient(patient_id) on delete cascade, constraint apt_fk foreign key (appointment_id) references appoint
ment(appointment_id)on delete cascade, constraint tr_fk foreign key (treatment_id) references treatment
(treatment_id) on delete cascade, constraint pll_fk foreign key (payment_id) references payment(payment_
id) on delete cascade);

Table created.

SQL> desc treatment_detail;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 PATIENT_ID                                NOT NULL NUMBER
 APPOINTMENT_ID                            NOT NULL NUMBER
 TREATMENT_ID                              NOT NULL NUMBER
 PAYMENT_ID                                NOT NULL NUMBER

SQL>
```

Figure 12: Creating Table Treatment_detail.

Create table Treatment(Treatment_id number not null, constraint t_pk primary key (treatment_id), Treatments varchar2(25) not null, Treatment_price number not null);

```
SQL> create table Treatment(Treatment_id number not null, constraint t_pk primary key (treatment_id), Tr
eatments varchar2(25) not null, Treatment_price number not null);

Table created.

SQL> desc treatment;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 TREATMENT_ID                              NOT NULL NUMBER
 TREATMENTS                                NOT NULL VARCHAR2(25)
 TREATMENT_PRICE                           NOT NULL NUMBER

SQL>
```

Figure 13: Creating Table Treatment.

b. Populate DB tables.

Insert into patient values (1, 'Kishor karki', 'new', 'yes', 'Certified');

Insert into patient values (2, 'David Gurung', 'new', 'no', 'Null');

Insert into patient values (3, 'Ravi Tamang', 'regular', 'yes', 'Uncertified');

Insert into patient values (4, 'Justin adhikari', 'new', 'yes', 'Uncertified');

Insert into patient values (5, 'Rajan Shrestha', 'new', 'yes', 'Certified');

Insert into patient values (6, 'Shiva kc', 'regular', 'no', 'Null');

Insert into patient values (7, 'Himal sherpa', 'regular', 'no', 'Null');

Insert into patient values (8, 'Furba yadav', 'new', 'no', 'NUll');

```
SQL> insert into patient values (1, 'Kishor karki', 'new', 'yes', 'Certified');
1 row created.
SQL> insert into patient values (2, 'David Gurung', 'new', 'no', 'Null');
1 row created.
SQL> insert into patient values (3, 'Ravi Tamang', 'regular', 'yes', 'Uncertified');
1 row created.
SQL> insert into patient values (4, 'Justin adhikari', 'new', 'yes', 'Uncertified');
1 row created.
SQL> insert into patient values (5, 'Rajan Shrestha', 'new', 'yes', 'Certified');
1 row created.
SQL> insert into patient values (6, 'Shiva kc', 'regular', 'no', 'Null');
1 row created.
SQL> insert into patient values (7, 'Himal sherpa', 'regular', 'no', 'Null');
1 row created.
SQL> insert into patient values (8, 'Furba yadav', 'new', 'no', 'NUll');
1 row created.
SQL> commit;
Commit complete.
```

Figure 14: Inserting values in table Patient.

Insert into patient_address values (1, 'Bhairahawa', 'Lumbini', 'Nepal', 'B.p path', 7);

Insert into patient_address values (2, 'Syangha', 'Gandaki', 'Nepal', 'Kalikalkot', 21);

Insert into patient_address values (3, 'Kathmandu', 'Bagmati', 'Nepal', 'Chakrapth', 36);

Insert into patient_address values (4, 'California', 'Northern Coastal', 'USA', 'Crenshaw', 18);

Insert into patient_address values (5, 'Lalitpur', 'Bagmati', 'Nepal', 'Khokhana', 32);

Insert into patient_address values (6, 'Butwal', 'Lumbini', 'Nepal', 'Janak path', 20);

Insert into patient_address values (7, 'vancouver', 'Downtown', 'Canada', '29th Ave station', 15);

Insert into patient_address values (8, 'MahendraNagar', 'Mahakali', 'Nepal', 'Bheemdatta', 9);

```
SQL> insert into patient_address values (1, 'Bhairahawa', 'Lumbini', 'Nepal', 'B.p path', 7);

1 row created.

SQL> insert into patient_address values (2, 'Syangha', 'Gandaki', 'Nepal', 'Kalikalkot', 21);

1 row created.

SQL> insert into patient_address values (3, 'Kathmandu', 'Bagmati', 'Nepal', 'Chakrapth', 36);

1 row created.

SQL> insert into patient_address values (4, 'California', 'Northern Coastal', 'USA', 'Crenshaw', 18);

1 row created.

SQL> insert into patient_address values (5, 'Lalitpur', 'Bagmati', 'Nepal', 'Khokhana', 32);

1 row created.

SQL> insert into patient_address values (6, 'Butwal', 'Lumbini', 'Nepal', 'Janak path', 20);

1 row created.

SQL> insert into patient_address values (7, 'vancouver', 'Downtown', 'Canada', '29th Ave station', 15);

1 row created.

SQL> insert into patient_address values (8, 'MahendraNagar', 'Mahakali', 'Nepal', 'Bheemdatta', 9);

1 row created.

SQL> commit;

Commit complete.

SQL>
```

Figure 15: Inserting values in table Patient_address.

Insert into patient_contact values (1, 523643, 'NULL', '7894235', 'kishor@yahoo.com');

Insert into patient_contact values (2, 451236, '9807445238', 'NULL', 'gurung@hotmail.com');

Insert into patient_contact values (3, 256324, 'NULL', '5487962', 'rtamang@yahoo.com');

Insert into patient_contact values (4, 365156, 'NULL', '3489756', 'jadhikari123@gmail.com');

Insert into patient_contact values (5, 188945, '984578913', 'NULL', 'rajan55@yahoo.com');

Insert into patient_contact values (6, 789456, 'NULL', '8564258', 'kc_shiva@gmail.com');

Insert into patient_contact values (7, 652345, 'NULL', 'NULL', 'sherpa456@hotmail.com');

Insert into patient_contact values (8, 854126, '9810236487', 'NULL', 'furba789@gmail.com');

```
SQL> insert into patient_contact values (1, 523643, 'NULL', '7894235', 'kishor@yahoo.com');

1 row created.

SQL> insert into patient_contact values (2, 451236, '9807445238', 'NULL', 'gurung@hotmail.com');

1 row created.

SQL> insert into patient_contact values (3, 256324, 'NULL', '5487962', 'rtamang@yahoo.com');

1 row created.

SQL> insert into patient_contact values (4, 365156, 'NULL', '3489756', 'jadhikari123@gmail.com');

1 row created.

SQL> insert into patient_contact values (5, 188945, '984578913', 'NULL', 'rajan55@yahoo.com');

1 row created.

SQL> insert into patient_contact values (6, 789456, 'NULL', '8564258', 'kc_shiva@gmail.com');

1 row created.

SQL> insert into patient_contact values (7, 652345, 'NULL', 'NULL', 'sherpa456@hotmail.com');

1 row created.

SQL> insert into patient_contact values (8, 854126, '9810236487', 'NULL', 'furba789@gmail.com');

1 row created.

SQL> commit;

Commit complete.

SQL>
```

Figure 16: Inserting values in table Patient_contact.

Insert into patient_detail values (1, 3, 4);

Insert into patient_detail values (2, 5, 6);

Insert into patient_detail values (3, 7, 8);

Insert into patient_detail values (4, 2, 1);

Insert into patient_detail values (5, 8, 3);

Insert into patient_detail values (6, 4, 5);

Insert into patient_detail values (7, 6, 2);

Insert into patient_detail values (8, 1, 7);

```
SQL> insert into patient_detail values (1, 3, 4);

1 row created.

SQL> insert into patient_detail values (2, 5, 6);

1 row created.

SQL> insert into patient_detail values (3, 7, 8);

1 row created.

SQL> insert into patient_detail values (4, 2, 1);

1 row created.

SQL> insert into patient_detail values (5, 8, 3);

1 row created.

SQL> insert into patient_detail values (6, 4, 5);

1 row created.

SQL> insert into patient_detail values (7, 6, 2);

1 row created.

SQL> insert into patient_detail values (8, 1, 7);

1 row created.

SQL> commit;

Commit complete.

SQL>
```

Figure 17: Inserting values in table Patient_detail.

Insert into Doctors values (1, 'Ram kc', 'Certified');

Insert into Doctors values (2, 'Bhanu tamang', 'Uncertified');

Insert into Doctors values (3, 'Prashant karki', 'Uncertified');

Insert into Doctors values (4, 'Rupesh hamal', 'Certified');

Insert into Doctors values (5, 'Bibek Maharajan', 'Uncertified');

Insert into Doctors values (6, 'Manish BK', 'Certified');

Insert into Doctors values (7, 'Sashi Gurung', 'Certified');

```
SQL> insert into Doctors values (1, 'Ram kc', 'Certified');

1 row created.

SQL> insert into Doctors values (2, 'Bhanu tamang', 'Uncertified');

1 row created.

SQL> insert into Doctors values (3, 'Prashant karki', 'Uncertified');

1 row created.

SQL> insert into Doctors values (4, 'Rupesh hamal', 'Certified');

1 row created.

SQL> insert into Doctors values (5, 'Bibek Maharajan', 'Uncertified');

1 row created.

SQL> insert into Doctors values (6, 'Manish BK', 'Certified');

1 row created.

SQL> insert into Doctors values (7, 'Sashi Gurung', 'Certified');

1 row created.

SQL> commit;

Commit complete.

SQL>
```

Figure 18: Inserting values in table Doctors.

Insert into Ward values (1, 'W011', 'Normal');

Insert into Ward values (2, 'W012', 'private');

Insert into Ward values (3, 'W013', 'emergency');

Insert into Ward values (4, 'W014', 'private');

Insert into Ward values (5, 'W015', 'normal');

Insert into Ward values (6, 'W016', 'Normal');

Insert into Ward values (7, 'W017', 'emergency');

```
SQL> insert into Ward values (1, 'W011', 'Normal');

1 row created.

SQL> insert into Ward values (2, 'W012', 'private');

1 row created.

SQL> insert into Ward values (3, 'W013', 'emergency');

1 row created.

SQL> insert into Ward values (4, 'W014', 'private');

1 row created.

SQL> insert into Ward values (5, 'W015', 'normal');

1 row created.

SQL> insert into Ward values (6, 'W016', 'Normal');

1 row created.

SQL> insert into Ward values (7, 'W017', 'emergency');

1 row created.

SQL>
SQL> commit;

Commit complete.

SQL>
```

Figure 19: Inserting values in table Ward.

Insert into Appointment values (1, 'Headache', '22-Jan-19');

Insert into Appointment values (2, 'Fever', '22-Jan-19');

Insert into Appointment values (3, 'Kidney Operation', '23-Jan-19');

Insert into Appointment values (4, 'Malaria', '24-Jan-19');

Insert into Appointment values (5, 'Brain Tumor', '24-Jan-19');

Insert into Appointment values (6, 'Heart Transplant', '25-Jan-19');

Insert into Appointment values (7, 'Lung Cancer', '26-Jan-19');

```
SQL> insert into Appointment values (1, 'Headache', '22-Jan-19');

1 row created.

SQL> insert into Appointment values (2, 'Fever', '22-Jan-19');

1 row created.

SQL> insert into Appointment values (3, 'Kidney Operation', '23-Jan-19');

1 row created.

SQL> insert into Appointment values (4, 'Malaria', '24-Jan-19');

1 row created.

SQL> insert into Appointment values (5, 'Brain Tumor', '24-Jan-19');

1 row created.

SQL> insert into Appointment values (6, 'Heart Transplant', '25-Jan-19');

1 row created.

SQL> insert into Appointment values (7, 'Lung Cancer', '26-Jan-19');

1 row created.

SQL> commit;

Commit complete.

SQL>
```

Figure 20: Inserting values in table Appointment.

Insert into Appointment_detail values (1, 2, 7, 6);

Insert into Appointment_detail values (2, 3, 3, 7);

Insert into Appointment_detail values (3, 1, 2, 2);

Insert into Appointment_detail values (4, 6, 5, 3);

Insert into Appointment_detail values (5, 7, 6, 4);

Insert into Appointment_detail values (6, 5, 1, 1);

Insert into Appointment_detail values (7, 4, 4, 5);

```
SQL> insert into Appointment_detail values (1, 2, 7, 6);

1 row created.

SQL> insert into Appointment_detail values (2, 3, 3, 7);

1 row created.

SQL> insert into Appointment_detail values (3, 1, 2, 2);

1 row created.

SQL> insert into Appointment_detail values (4, 6, 5, 3);

1 row created.

SQL> insert into Appointment_detail values (5, 7, 6, 4);

1 row created.

SQL> insert into Appointment_detail values (6, 5, 1, 1);

1 row created.

SQL> insert into Appointment_detail values (7, 4, 4, 5);

1 row created.

SQL> commit;

Commit complete.

SQL>
```

Figure 21: Inserting values in table Appointment_detail.

Insert into Treatment values (1, 'Medicine', 600);

Insert into Treatment values (2, 'Steam Bath', 700);

Insert into Treatment values (3, 'Surgery', 800);

Insert into Treatment values (4, 'Blood test', 600);

Insert into Treatment values (5, 'Sugar test', 550);

Insert into Treatment values (6, 'Injection', 750);

Insert into Treatment values (7, 'X-ray', 650);

```
SQL> insert into Treatment values (1, 'Medicine', 600);

1 row created.

SQL> insert into Treatment values (2, 'Steam Bath', 700);

1 row created.

SQL> insert into Treatment values (3, 'Surgery', 800);

1 row created.

SQL> insert into Treatment values (4, 'Blood test', 600);

1 row created.

SQL> insert into Treatment values (5, 'Sugar test', 550);

1 row created.

SQL> insert into Treatment values (6, 'Injection', 750);

1 row created.

SQL> insert into Treatment values (7, 'X-ray', 650);

1 row created.

SQL> commit;

Commit complete.
```

Figure 22: Inserting values in table Treatment.

Insert into Payment values (1, '5000', '2000');

Insert into Payment values (2, 'free', 'free');

Insert into Payment values (3, '3000', '1500');

Insert into Payment values (4, 'free', 'free');

Insert into Payment values (5, '5000', '2000');

Insert into Payment values (6, '6000', '2500');

Insert into Payment values (7, '4000', '1500');

```
SQL> insert into Payment values (1, '5000', '2000');

1 row created.

SQL> insert into Payment values (2, 'free', 'free');

1 row created.

SQL> insert into Payment values (3, '3000', '1500');

1 row created.

SQL> insert into Payment values (4, 'free', 'free');

1 row created.

SQL> insert into Payment values (5, '5000', '2000');

1 row created.

SQL> insert into Payment values (6, '6000', '2500');

1 row created.

SQL> insert into Payment values (7, '4000', '1500');

1 row created.

SQL> commit;

Commit complete.

SQL>
```

Figure 23: Inserting values in table Payment.

Insert into Treatment_detail values (1, 5, 7, 6);

Insert into Treatment_detail values (2, 3, 6, 4);

Insert into Treatment_detail values (3, 6, 4, 7);

Insert into Treatment_detail values (4, 7, 5, 3);

Insert into Treatment_detail values (5, 2, 3, 1);

Insert into Treatment_detail values (6, 1, 2, 5);

Insert into Treatment_detail values (7, 4, 1, 2);

```
SQL> insert into Treatment_detail values (1, 5, 7, 6);

1 row created.

SQL> insert into Treatment_detail values (2, 3, 6, 4);

1 row created.

SQL> insert into Treatment_detail values (3, 6, 4, 7);

1 row created.

SQL> insert into Treatment_detail values (4, 7, 5, 3);

1 row created.

SQL> insert into Treatment_detail values (5, 2, 3, 1);

1 row created.

SQL> insert into Treatment_detail values (6, 1, 2, 5);

1 row created.

SQL> insert into Treatment_detail values (7, 4, 1, 2);

1 row created.

SQL> commit;

Commit complete.

SQL>
```

Figure 24: Inserting values in table Treatment_detail.

c. Final tables.

Select * from Patient;

```
PATIENT_ID NAME                      PATIENT_TYPE        STAFF_CHECK         STAFF_TYPE
---------- ------------------------- ------------------- ------------------- --------------------
         1 Kishor karki              new                 yes                 Certified
         2 David Gurung              new                 no                  Null
         3 Ravi Tamang               regular             yes                 Uncertified
         4 Justin adhikari           new                 yes                 Uncertified
         5 Rajan Shrestha            new                 yes                 Certified
         6 Shiva kc                  regular             no                  Null
         7 Himal sherpa              regular             no                  Null
         8 Furba yadav               new                 no                  NUll

8 rows selected.
```

Figure 25: Patient table.

Select * from Patient_address;

```
SQL> select * from patient_address;

ADDRESS_ID CITY                ZONE                COUNTRY             STREET              STREET_NUMBER
---------- ------------------- ------------------- ------------------- ------------------- -------------
         1 Bhairahawa          Lumbini             Nepal               B.p path                        7
         2 Syangha             Gandaki             Nepal               Kalikalkot                     21
         3 Kathmandu           Bagmati             Nepal               Chakrapth                      36
         4 California          Northern Coastal    USA                 Crenshaw                       18
         5 Lalitpur            Bagmati             Nepal               Khokhana                       32
         6 Butwal              Lumbini             Nepal               Janak path                     20
         7 vancouver           Downtown            Canada              29th Ave station               15
         8 MahendraNagar       Mahakali            Nepal               Bheemdatta                      9

8 rows selected.

SQL>
```

Figure 26: Patient_address Table.

Select * from Patient_contact;

```
SQL> select * from patient_contact;

CONTACT_ID PHONE_NUMBER CELL_PHONE          FAX_NO              EMAIL
---------- ------------ -------------------- ------------------- --------------------------------------------------
         1       523643 NULL                 7894235             kishor@yahoo.com
         2       451236 9807445238           NULL                gurung@hotmail.com
         3       256324 NULL                 5487962             rtamang@yahoo.com
         4       365156 NULL                 3489756             jadhikari123@gmail.com
         5       188945 984578913            NULL                rajan55@yahoo.com
         6       789456 NULL                 8564258             kc_shiva@gmail.com
         7       652345 NULL                 NULL                sherpa456@hotmail.com
         8       854126 9810236487           NULL                furba789@gmail.com

8 rows selected.

SQL>
```

Figure 27: Patient_contact Table.

Select * from Patient_detail;

```
SQL> select * from patient_detail;

PATIENT_NO ADDRESS_NO CONTACT_NO
---------- ---------- ----------
         1          3          4
         2          5          6
         3          7          8
         4          2          1
         5          8          3
         6          4          5
         7          6          2
         8          1          7

8 rows selected.

SQL>
```

Figure 28: Patient_detail Table.

Select * from Doctors;

```
SQL> select * from doctors;

 DOCTOR_ID DOCTOR_NAME                DOCTOR_TYPE
---------- ------------------------- --------------------
         1 Ram kc                    Certified
         2 Bhanu tamang              Uncertified
         3 Prashant karki            Uncertified
         4 Rupesh hamal              Certified
         5 Bibek Maharajan           Uncertified
         6 Manish BK                 Certified
         7 Sashi Gurung              Certified

7 rows selected.

SQL>
```

Figure 29: Doctors Table.

Select * from Ward;

```
SQL> select * from Ward;

   WARD_ID WARD_ROOM            WARD_TYPE
---------- -------------------- --------------------
         1 W011                 Normal
         2 W012                 private
         3 W013                 emergency
         4 W014                 private
         5 W015                 normal
         6 W016                 Normal
         7 W017                 emergency

7 rows selected.

SQL>
```

Figure 30: Ward Table.

Select * from Appointment;

```
SQL> select * from appointment;

APPOINTMENT_ID APPOINTMENT_CASE     APPOINTME
-------------- -------------------- ---------
             1 Headache             22-JAN-19
             2 Fever                22-JAN-19
             3 Kidney Operation     23-JAN-19
             4 Malaria              24-JAN-19
             5 Brain Tumor          24-JAN-19
             6 Heart Transplant     25-JAN-19
             7 Lung Cancer          26-JAN-19

7 rows selected.

SQL>
```

Figure 31: Appointment Table.

Select * from Appointment_detail;

```
SQL> select * from appointment_detail;

PATIENT_ID APPOINTMENT_ID  DOCTOR_ID    WARD_ID
---------- -------------- ---------- ----------
         1              2          7          6
         2              3          3          7
         3              1          2          2
         4              6          5          3
         5              7          6          4
         6              5          1          1
         7              4          4          5

7 rows selected.

SQL>
```

Figure 32: Appointment_detail Table.

Select * from Treatment_detail;

```
SQL> select * from treatment_detail;

PATIENT_ID APPOINTMENT_ID TREATMENT_ID PAYMENT_ID
---------- -------------- ------------ ----------
         1              5            7          6
         2              3            6          4
         3              6            4          7
         4              7            5          3
         5              2            3          1
         6              1            2          5
         7              4            1          2

7 rows selected.

SQL>
```

Figure 33: Treatment_detail Table.

Select * from Payment;

```
SQL> select * from payment;

PAYMENT_ID TOTALAMOUNT_PAYMENT  DOCTORCHARGE_PAYMENT
---------- -------------------- --------------------
         1 5000                 2000
         2 free                 free
         3 3000                 1500
         4 free                 free
         5 5000                 2000
         6 6000                 2500
         7 4000                 1500

7 rows selected.

SQL>
```

Figure 34: Payment Table.

Select * from Treatment;

```
SQL> select * from treatment;

TREATMENT_ID TREATMENTS                TREATMENT_PRICE
------------ ------------------------- ---------------
           1 Medicine                              600
           2 Steam Bath                            700
           3 Surgery                               800
           4 Blood test                            600
           5 Sugar test                            550
           6 Injection                             750
           7 X-ray                                 650

7 rows selected.

SQL>
```

Figure 35: Treatment Table.

9. Database Querying.

- List all patients, regular and new.

Select * from patient where patient_type in (‘regular’, ‘new’);

```
SQL> select * from patient where patient_type in ('regular', 'new');

PATIENT_ID NAME                      PATIENT_TYPE        STAFF_CHECK         STAFF_TYPE
---------- ------------------------- ------------------- ------------------- --------------------
         1 Kishor karki              new                 yes                 Certified
         2 David Gurung              new                 no                  Null
         3 Ravi Tamang               regular             yes                 Uncertified
         4 Justin adhikari           new                 yes                 Uncertified
         5 Rajan Shrestha            new                 yes                 Certified
         6 Shiva kc                  regular             no                  Null
         7 Himal sherpa              regular             no                  Null
         8 Furba yadav               new                 no                  NUll

8 rows selected.

SQL>
```

Figure 36: First Query.

- List all patients with their addresses.

Select d.patient_no, p.name, a.city, a.zone, a.country, a.street, a.street_number, d.address_no from patient p, patient_address a, patient_detail d where d.patient_no= p.patient_id and d.address_no =a.address_id order by d.patient_no asc;

```
SQL> select d.patient_no, p.name, a.city, a.zone, a.country, a.street, a.street_number, d.address_no from patient p, patient_address a, patient_detail d where
d.patient_no = p.patient_id and d.address_no = a.address_id order by d.patient_no asc;

PATIENT_NO NAME                      CITY                ZONE                COUNTRY             STREET              STREET_NUMBER ADDRESS_NO
---------- ------------------------- ------------------- ------------------- ------------------- ------------------- ------------- ----------
         1 Kishor karki              Kathmandu           Bagmati             Nepal               Chakrapth                      36          3
         2 David Gurung              Lalitpur            Bagmati             Nepal               Khokhana                       32          5
         3 Ravi Tamang               vancouver           Downtown            Canada              29th Ave station               15          7
         4 Justin adhikari           Syangha             Gandaki             Nepal               Kalikalkot                     21          2
         5 Rajan Shrestha            MahendraNagar       Mahakali            Nepal               Bheemdatta                      9          8
         6 Shiva kc                  California          Northern Coastal    USA                 Crenshaw                       18          4
         7 Himal sherpa              Butwal              Lumbini             Nepal               Janak path                     20          6
         8 Furba yadav               Bhairahawa          Lumbini             Nepal               B.p path                        7          1

8 rows selected.

SQL>
```

Figure 37: Second Query.

- For a given certified doctor, find all the appointment he/she have been conducted and the amount he/she got for conducting the appointment.

Select d.doctor_id, d.doctor_name, a.appointment_id, a.appointment_case, t.treatments, p.doctorcharge_payment from doctors d, appointment a, treatment t, payment p, appointment_detail ad, treatment_detail td where d.doctor_type='Certified' and d.doctor_id=ad.doctor_id and a.appointment_id=ad.appointment_id and a.appointment_id = td.appointment_id and t.treatment_id=td.treatment_id and p.payment_id=td.payment_id;

```
SQL> select d.doctor_id, d.doctor_name, a.appointment_id, a.appointment_case, t.treatments, p.doctorcharge_payment from doctors d, appointment
 a, treatment t, payment p, appointment_detail ad, treatment_detail td where d.doctor_type='Certified' and d.doctor_id=ad.doctor_id and a.appo
intment_id=ad.appointment_id and a.appointment_id = td.appointment_id and t.treatment_id=td.treatment_id and p.payment_id=td.payment_id;

 DOCTOR_ID DOCTOR_NAME             APPOINTMENT_ID APPOINTMENT_CASE     TREATMENTS              DOCTORCHARGE_PAYMENT
---------- ----------------------- -------------- -------------------- ----------------------- --------------------
         7 Sashi Gurung                         2 Fever                Surgery                 2000
         4 Rupesh hamal                         4 Malaria              Medicine                free
         6 Manish BK                            7 Lung Cancer          Sugar test              1500
         1 Ram kc                               5 Brain Tumor          X-ray                   2500
```

Figure 38: Third Query.

- List all staffs that are also a patient.

Select * from patient where staff_check='yes';

```
SQL> select * from patient where staff_check='yes';

PATIENT_ID NAME                      PATIENT_TYPE         STAFF_CHECK          STAFF_TYPE
---------- ------------------------- -------------------- -------------------- --------------------
         1 Kishor karki              new                  yes                  Certified
         3 Ravi Tamang               regular              yes                  Uncertified
         4 Justin adhikari           new                  yes                  Uncertified
         5 Rajan Shrestha            new                  yes                  Certified
```

Figure 39: Fourth Query.

- List all uncertified doctors who have been attended an appointment for a treatment and the amount he/she have paid.

Select p.patient_id, p.name, p.patient_type, a.appointment_case, a.appointment_date, t.treatments, pa.totalamount_payment, pa.doctorcharge_payment from patient p, appointment a, treatment t, payment pa, appointment_detail ad, treatment_detail td where p.staff_type='Uncertified' and p.patient_id=ad.patient_id and p.patient_id=td.patient_id and a.appointment_id=ad.appointment_id and t.treatment_id=td.treatment_id and pa.payment_id=td.payment_id;

```
SQL> select p.patient_id, p.name, p.patient_type, a.appointment_case, a.appointment_date, t.treatments, pa.totalamount_payment, pa.doctorcharge_payment from patient p, appointment a, treatme
nt t, payment pa, appointment_detail ad, treatment_detail td where p.staff_type='Uncertified' and p.patient_id=ad.patient_id and p.patient_id=td.patient_id and a.appointment_id=ad.appointmen
t_id and t.treatment_id=td.treatment_id and pa.payment_id=td.payment_id;

PATIENT_ID NAME                 PATIENT_TYPE       APPOINTMENT_CASE    APPOINTME TREATMENTS             TOTALAMOUNT_PAYMENT  DOCTORCHARGE_PAYMENT
---------- ------------------------ ------------------- ------------------- --------- ------------------------ -------------------- --------------------
         3 Ravi Tamang              regular            Headache            22-JAN-19 Blood test               4000                 1500
         4 Justin adhikari          new                Heart Transplant    25-JAN-19 Sugar test               3000                 1500
```

Figure 40: Fifth Query.

- List all appointments that have been conducted in an emergency ward.

Select a.appointment_id, a.appointment_case, a.appointment_date from appointment a, ward w, appointment_detail ad where w.ward_type='emergency' and ad.appointment_id=a.appointment_id and w.ward_id= ad.ward_id;

```
SQL> select a.appointment_id, a.appointment_case, a.appointment_date from appointment a, ward w, appointment_detail ad where w.ward_type='emergency' and ad.appointment_id=a.appointment_id an
d w.ward_id= ad.ward_id;

APPOINTMENT_ID APPOINTMENT_CASE     APPOINTME
-------------- -------------------- ---------
             3 Kidney Operation     23-JAN-19
             6 Heart Transplant     25-JAN-19
```

Figure 41: Sixth Query.

- List all staffs who have conducted an appointment on a given date.

Select a.appointment_id, a.appointment_case, a.appointment_date, d.doctor_name, d.doctor_type from appointment a, doctors d, appointment_detail ad where ad.appointment_id=a.appointment_id and d.doctor_id = ad.doctor_id;

```
SQL> select a.appointment_id, a.appointment_case, a.appointment_date, d.doctor_name, d.doctor_type from appointment a, doctors d, appointment_detail ad where ad.appointment_id=a.appointment_
id and d.doctor_id = ad.doctor_id;

APPOINTMENT_ID APPOINTMENT_CASE     APPOINTME DOCTOR_NAME              DOCTOR_TYPE
-------------- -------------------- --------- ------------------------ --------------------
             5 Brain Tumor          24-JAN-19 Ram kc                   Certified
             1 Headache             22-JAN-19 Bhanu tamang             Uncertified
             3 Kidney Operation     23-JAN-19 Prashant karki           Uncertified
             4 Malaria              24-JAN-19 Rupesh hamal             Certified
             6 Heart Transplant     25-JAN-19 Bibek Maharajan          Uncertified
             7 Lung Cancer          26-JAN-19 Manish BK                Certified
             2 Fever                22-JAN-19 Sashi Gurung             Certified

7 rows selected.
```

Figure 42: Seventh Query.

- List all patients booked for an appointment on a given date.

Select p.patient_id, p.name, a.appointment_case, a.appointment_date from patient p, appointment a, appointment_detail ad where p.patient_id=ad.patient_id and a.appointment_id=ad.appointment_id;

```
SQL> select p.patient_id, p.name, a.appointment_case, a.appointment_date from patient p, appointment a, appointment_detail ad where p.patient_id=ad.patient_id and a.appointment_id=ad.appoint
ment_id;

PATIENT_ID NAME                      APPOINTMENT_CASE     APPOINTME
---------- ------------------------- -------------------- ---------
         1 Kishor karki              Fever                22-JAN-19
         2 David Gurung              Kidney Operation     23-JAN-19
         3 Ravi Tamang               Headache             22-JAN-19
         4 Justin adhikari           Heart Transplant     25-JAN-19
         5 Rajan Shrestha            Lung Cancer          26-JAN-19
         6 Shiva kc                  Brain Tumor          24-JAN-19
         7 Himal sherpa              Malaria              24-JAN-19

7 rows selected.
```

Figure 43: Eighth Query.

10. Drop Queries.

```
drop table appointment_detail;
drop table treatment_detail;
drop table patient;
drop table patient_address;
drop table patient_contact;
drop table appointment;
drop table treatment;
drop table payment;
drop table ward;
drop table doctors;
drop table patient_detail;
```

```
SQL> drop table doctors;

Table dropped.

SQL> drop table patient_detail;

Table dropped.

SQL>
```

```
SQL> drop table appointment_detail;
Table dropped.
SQL> drop table treatment_detail;
Table dropped.
SQL> drop table patient;
Table dropped.
SQL> drop table patient_address;
Table dropped.
SQL> drop table patient_contact;
Table dropped.
SQL> drop table appointment;
Table dropped.
SQL> drop table treatment;
Table dropped.
SQL> drop table payment;
Table dropped.
SQL> drop table ward;
Table dropped.
SQL>
```

Figure 44: Drop queries

11. Critical Evaluation.

My experience while dealing with this coursework was a big challenge. Though, I had studied the basic for this subject in my first year of college, but there were many new things that I had not learned in my past due to which I had some difficulties in understanding this module for a while. However, as there is solution for any sort of problems what I did is firstly going through all the lecture slides and doing various research on web and some time by watching YouTube videos. Though, I was preparing myself but when it comes to the normalization part of this coursework where I had most difficulties it looked me a week to solve my problems. Although, I had done my normalization but, due to unawareness of the queries that I had to do for this coursework I had to make some changes in the middle of the process due to I had to create and insert the table again and again. Even though I was facing problems regarding this coursework, it taught me a good lesson of interacting with our teachers, friends and taught me the value of timed due to which I was able to complete this coursework.

12. Critical Assessment of coursework.

Hence, this coursework have me a clear concept on why database is used and its purpose. Its main purpose is to store data and to maintain for the needs in the future. It is used in various organisation in order to keep the records and various things. As, the world is rising in terms of the technology and recording data in the papers will be time consuming and will be hard to handle whereas the use of database can solve the problem of time consuming as the data can be store within second and it does not take much time. As, this coursework is based on the database it gave a deep knowledge about the database and its importance and taught me how to store data with repetition and many more.