Module Code & Module Title

CC5051NA: Databases

Assessment Weightage & Type

50% Individual Coursework

Semester

2019 Autumn

Student Name: Girija Tamang

London Met ID:18030995

College ID:NP05CP4S190007

Assignment Due Date: 30th Dec 2019

Assignment Submission Date:30th Dec 2019

Academic Supervisor: Mr. Lekhnath Katuwal

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 mark of zero will be awarded.

Contents

Table of Figures.

1. Introduction

1.1 Introduction of the Hospital

Syangbo International Hospital was established with the vision of filling an existing void in the healthcare industry in Nepal. Syangbo International Hospital was established in 1994 by the Tamang Group, one of the leading corporate centers in Nepal. The Syangbo International Hospital is a 150-bed hospital; a state-of-the-art, super-specialty hospital located in Dharan. Designed as one of the leading providers of specialized super-specialty healthcare facilities in the country, Syangbo International Hospital is proud to provide in-depth expertise in the field of advanced medical and surgical procedures, as well as a robust combination of inpatient and outpatient services.

The hospital has special features such as a wellness center to help the community stay healthy. A combination of state-of-the-art technology in the hands of renowned physicians across the country have set new standards in healthcare. Stellar services in a warm, flexible and affordable environment have made us one of the most reliable hospitals in the country.

Aims and Objectives

- Improving the health status of poor and marginalized people living in rural areas with special health services programs.
- Provide basic preventive services to patients and surrounding communities, such as family planning, immunization and health awareness programs.
- Coordinate and cooperate with government agencies and stakeholders / donors concerned in order to strengthen our unity in the fight against disease.
- Provide appropriate quality treatment services for both in- and outpatient care.
- Provide a safe and therapeutic environment for all patient, staff and visitors. Increase overall satisfaction rates of patients, employees and visiting medical officers.

1.2 Current Business Activities and Operations

Syangbo International Hospital is committed to the consistent provision of reliable, patient-centered healthcare, which means providing everything a patient may need, from acute critical care to recovery to transitional treatment to home health services at an affordable cost. All the patient data is recorded in a patient database system under the supervision of staff. After the conformation of the hospital, patient get their appointment date and the details of staff for treatment process. Syangbo International hospital provides services such as inpatient care, emergency care, outpatient care, pharmacist, surgical and other medical services.

According to regulations, our polices and best practices, we handle, preserve and protect all information of our patient. We have security measures in place to preserve and protect the confidentiality, reliability, and availability of our systems and data. The medical record has four major sections:

• Administrative, including demographic and socio-economic data such as the patient's name, gender, date of birth, place of birth, permanent address of the patient and medical record number;

• Legal data, including a signed consent for the treatment of appointed physicians and authorisation for the release of information

• Financial data relating to the payment of medical services and hospital accommodation fees;

• Clinical data of the patient whether admitted to hospital or treated as an outpatient or emergency patient.

The quality and availability of data in the medical record is important to this hospital to ensure the optimum level of health care. It is important to note that accurate, timely and accessible data on health care plays a vital role in planning, developing and maintaining health care services.

1.3 Business Rules

1. An address can have one or many persons, but person has only one address.
2. Person may or may not be Staff of hospital, but Staff must be a person.
3. Person may or may not be Patient of hospital, but Patient must be a person.
4. Ward can conduct one or many appointments.
5. In each appointment patient is allocated in one ward.
6. An address can contain one or more email address.
7. Person may or may not have phone number.
8. Staff can have one or many appointments, but in every appointment there must a Staff.
9. Patient can take one or many appointments, but one patient must be in each appointment.
10. Address may or may not contain fax address/number.

1.4 Identification of Entities and Attributes

1.4.1 List of the created objects – Entities and Attributes

Address
Address_Id (pk)
Country
Province
State
Zone
City
Street
Street_Number
Phone_Number
Fax_Number
Email_Address

Person
Person_Id (pk)
Name
Age
Gender
Email
Phone_Number

Staff
Staff_Id (pk)
Post
Join_Date
Staff_Type
Salary

Patient
Patient_Id (pk)
Patient_Type
Blood group

Ward
Ward_Id (pk)
Ward_Type
Capacity

Appointment
Appointment_Id (pk)
Appointment_Date
Treatment_Fee

1.4.2 Identification and representation of the Primary Keys, Foreign Keys.

Entity Name	**Primary Key**	**Foreign Key**	**Reference Table**
Address	Address_Id		
Person	Person_Id	Address_Id	Address
Staff	Staff_Id	Person_Id	Person
Patient	Patient_Id	Person_Id	Person
Ward	Ward_Id		
Appointment	Appointment_Id	Ward_Id, Staff_Id, Patient_Id	Ward, Staff, Patient

1.4.3 Entity Relationship Diagram

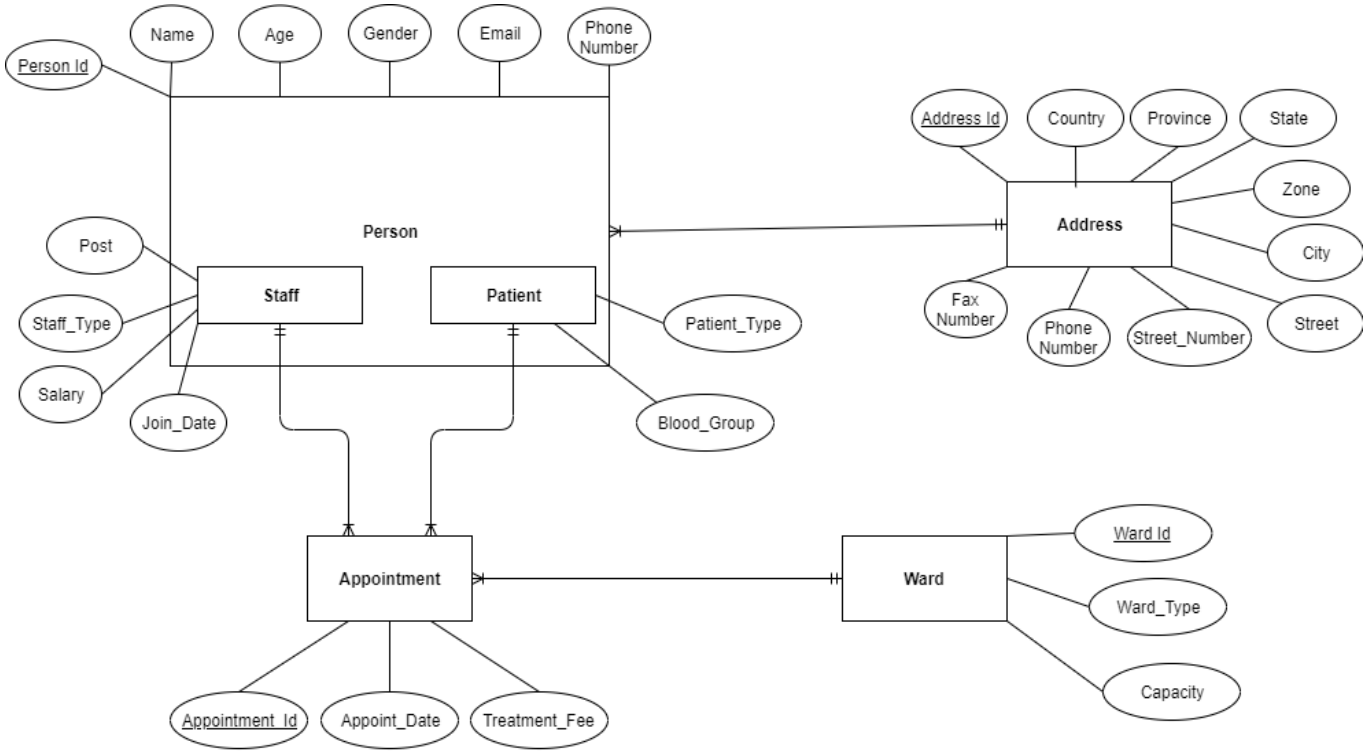

Figure 1: Entity Relationship Diagram

2. Normalization

2.1 Normalization from UNF to 3NF

UNF (Un-Normalized Form)

All the attributes are written, and entity name is given. All Repeating Group are written within curly braces {}.

People { People_Id, Name, Age, Gender, { Email }, { Phone_Number } , { Address_Id, Country, Province, State, Zone, City, Street, Street Number,{ Phone_Number}, {Fax_Number}, {Email } }, {Post, Join_Date, Staff_Type, Salary} ,{ Patient_Type, Blood_Group}, {Appointment_Id, Appoint_Date, Treatment_Fee, { Ward_Id , Ward_Type, Capacity } }

FIRST NORMALIZATION FORM (1NF)

All the tables are in 1NF because all the repeating group is eliminated, unique identifier is identified for new relationship.

Address (<u>Address Id</u>, Country, Province, State, Zone, City, Street, Street_Number)

Address_Email (<u>Address Email</u>, Address_Id*)

Address_Number (<u>Address Number</u>, Address_Id*)

Address_Fax (<u>Address Fax</u>, Address_Id*)

Person (<u>Person Id</u>, Name, Age, Gender, Address_Id*)

Person_Email (<u>Email</u>, Person_Id*)

Person_Number (<u>Cell Number</u>, Person_Id*)

Staff (<u>Staff Id</u>*, Post, Join_Date, Staff_Type, Salary)

Patient (<u>Patient Id*</u>, Blood_Group, Patient_Type)

Ward (<u>Ward Id</u>, Ward_Type, Capacity)

Appointment (<u>Appointment_Id</u>, Appoint_Date, Treatment_Fee, Staff_Id*, Patient_Id*, Ward_Id*)

SECOND NORMALIZATION FORM (2NF)

All the functional dependencies are reviewed. There is no partial functional dependency in all tables because the table does not contain any composite primary key. So, all the tables are already in 2NF.

Address (Address Id, Country, Province, State, Zone, City, Street, Street_Number)

Address_Email (Address Email, Address_Id*)

Address_Number (Address Number, Address_Id*)

Address_Fax (Address Fax, Address_Id*)

Person (Person Id, Name, Age, Gender, Address_Id *)

Person_Email (Email, Person_Id*)

Person_Number (Cell Number, Person_Id*)

Staff (Staff Id*, Post, Join_Date, Staff_Type, Salary)

Patient (Patient Id*, Blood_Group, Patient_Type)

Ward (Ward Id, Ward_Type, Capacity)

Appointment (Appointment_Id, Appoint_Date, Treatment_Fee, Staff_Id*, Patient_Id*, Ward_Id*)

Note: All the above tables did not contain composite primary key. So the table is in 2nf because there is not partial functional dependency.

THIRD NORMALIZATION FORM (3NF)

Transitive dependency is checked and separated.

Address (Address Id, Country, Province, State, Zone, City, Street, Street_Number)

Address_Id → Country →

Address_Id → Province →

Address_Id → State →

Address_Id → Zone →

Address_Id → City →

Address_Id → Street →

Address_Id → Street_Number →

The Address table does not consist of a transitive dependence because non-key attribute doesn't give another non-key which indicates that the table is already in 3NF.

Address_Email (Address Email, Address_Id*)

Note: There is no more than one non-key attribute in the Address_Email table. The Address_Email table does not consist of a transitive dependency, which means that the table is in 3NF.

Address_Number (Address Number, Address_Id*)

Note: There is no more than one non-key attribute in the Address_Number table. The Address_Number table does not consist of a transitive dependency, which means t hat the table is in 3NF.

Address_Fax (Address Fax, Address_Id*)

Note: There is no more than one non-key attribute in the Address_Fax table. Therefore, it has no transitive dependency. The table is in 3NF.

Person (Person Id, Name, Age, Gender, Address_Id *)

Person Id → Name →

Person Id → Age →

Person Id → Gender

Note: The Person table does not consist of a transitive dependence because non-key attribute doesn't give another non-key which indicates that the table is already in 3NF.

Person_Email (Email, Person_Id*)

Note: There is no more than one non-key attribute in the Person_Email table. The Person_Email table does not consist of a transitive dependency, which means that the table is in 3NF.

Person_Number (Cell Number, Person_Id*)

Note: There is no more than one non-key attribute in the Person_Number table. Therefore, it has no transitive dependency. The table is in 3NF.

Staff (Staff Id*, Post, Join_Date, Staff_Type, Salary)

Staff Id* → Staff_Type →

Staff Id* → Post→

Staff Id*→ Salary→

Staff Id*→ Join_Date→

Note: The Staff table does not consist of a transitive dependence because non-key attribute doesn't give another non-key which indicates that the table is already in 3NF.

Patient (Patient Id*, Blood_Group, Patient_Type)

Patient Id* → Patient_Type →

Patient Id* → Blood_Group →

Note: The Patient table does not consist of a transitive dependence because non-key attribute doesn't give another non-key which indicates that the table is already in 3NF.

Ward (Ward Id, Ward_Type, Capacity)

Ward Id→ Ward_Type →

Ward Id→ Capacity →

Note: The Ward table does not consist of a transitive dependence because non-key attribute doesn't give another non-key which indicates that the table is already in 3NF.

Appointment (Appointment_Id, Appoint_Date, Treatment_Fee, Staff_Id*, Patient_Id*, Ward_Id*)

Appointment Id → Appoint_Date →

Appointment Id → Treatment_Fee →

Note: The Appointment table does not consist of a transitive dependence because non-key attribute doesn't give another non-key which indicates that the table is already in 3NF.

2.2 Final Entity Relationship Diagram

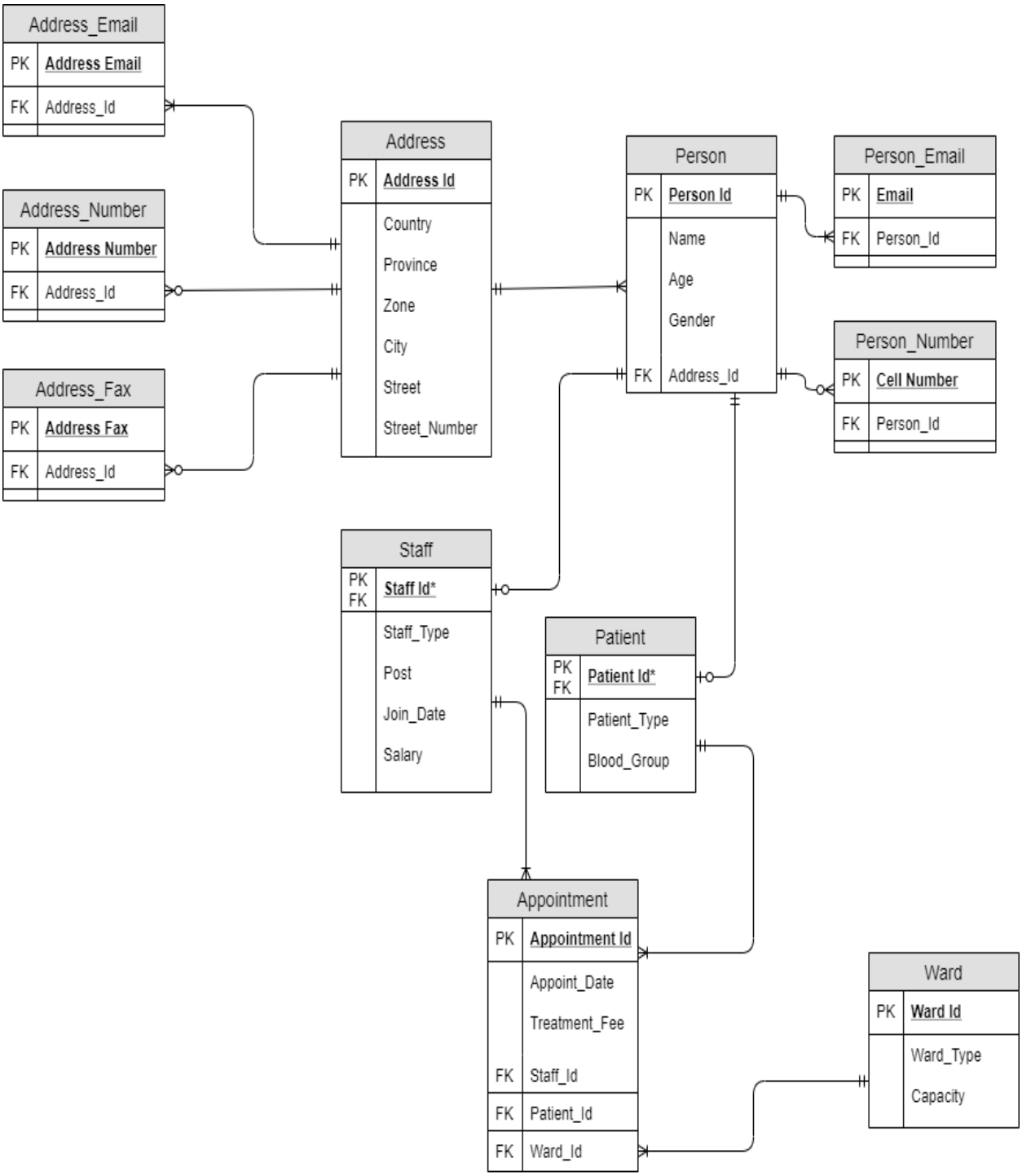

Figure 2: Final Entity Relationship Diagram.

2.3 Assumptions

- A hospital contains many wards and each ward has a unique ward number, its type, and capacity.
- In every appointment of the patient, one ward is allocated with unique appoint id.
- In address, there may be one or more people, but People have only one address.
- A person may be a patient or a staff of a hospital, but staff and patients must a person with a unique identity.
- The identity of each staff is unique and related entry should be present in staff and appointment table.
- The identity of the patient must be present in the Patient table and it must be unique.
- The staff has one or many appointments on a date.
- Each staff has a fixed monthly salary with separated consultation fees for treatment.
- The patient can take one or more appointments for treatment. Once an appointment is done it cannot be cancelled by the patient.
- The staff, as well as the patient, must provide his\her email address.
- There must be staff, patient as well as ward information in every appointment table with appoint date.

3. Implementation

3.1 Relations creation order

The correct sequence of creation of the final tables is as follows.

1. Address
2. Address_Fax
3. Address_Email
4. Address_Number
5. Person
6. Person_Email
7. Person_Number
8. Staff
9. Patient
10. Ward
11. Appointment

3.2 Relations drop order

The correct sequence of the final tables dropped is as follows.

1. Appointment

```
SQL> drop table Appointment;

Table dropped.
```

Figure 3: Table Appointment Dropped.

2. Ward

```
SQL> drop table Ward;

Table dropped.
```

Figure 4: Table Ward Dropped.

3. Patient

```
SQL> drop table Patient;

Table dropped.
```

Figure 5: Table Patient Dropped.

4. Staff

```
SQL> drop table Staff;

Table dropped.
```

Figure 6: Table Staff Dropped.

5. Person_Number

```
SQL> drop table Person_Number;

Table dropped.
```

Figure 7: Table Patient_Number Dropped

6. Person_Email

```
SQL> drop table Person_Email;

Table dropped.
```

Figure 8: Table Person_Email Dropped.

7. Person

```
SQL> drop table Person;

Table dropped.
```

Figure 9: Table Person Dropped.

8. Address_Number

```
SQL> drop table Address_Number;

Table dropped.
```

Figure 10: Table Address_Number Dropped.

9. Address_Email

```
SQL> drop table Address_Email;

Table dropped.
```

Figure 11: Table Address_Email Dropped.

10. Address_Fax

```
SQL> drop table Address_Fax;

Table dropped.
```

Figure 12: Table Address_Fax Dropped.

11. Address

```
SQL> drop table Address;

Table dropped.
```

Figure 13: Table Address Dropped.

3.3 Create Statement for each Relations

1. Creating Table Address.

Create Table Address (Address_Id varchar(5), Country varchar(25) not null, Province int not null, State varchar(50) not null, Zone varchar(50) not null, City varchar(50) not null, Street varchar(50) not null, Street_Number int not null, constraint address_pk primary key(Address_Id));

```
SQL> Create Table Address(Address_Id varchar(5), Country varchar(25) not null, Province int not null, State varchar(50)
not null, Zone varchar(50) not null, City varchar(50) not null, Street varchar(50) not null, Street_Number int not null,
 constraint address_pk primary key(Address_Id) );

Table created.
```

Figure 14: Creating Table Address.

2. Creating Table Address_Email

Create table Address_Email (Address_Email varchar (80), Address_Id varchar (5) not null, constraint email_pk primary key (Address_Email), constraint emailadd_fk foreign key (Address_Id) references Address (Address_Id));

```
SQL> Create table Address_Email (Address_Email varchar(80), Address_Id varchar(5) not null, constraint email_pk primary
key (Address_Email), constraint emailadd_fk foreign key (Address_Id) references Address (Address_Id));

Table created.
```

Figure 15: Creating Table Address_Email.

3. Creating Table Address_Number

Create table Address_Number (Address_Number int, Address_Id varchar (5) not null, constraint num_pk primary key (Address_Number), constraint number_fk foreign key (Address_Id) references Address (Address_Id));

```
SQL> Create table Address_Number (Address_Number int, Address_Id varchar (5) not null, constraint num_pk primary key (Ad
dress_Number), constraint number_fk foreign key (Address_Id) references Address (Address_Id));

Table created.
```

Figure 16: Creating Table Address_Number.

4. Creating Table Address_Fax

Create table Address_Fax (Address_Fax int, Address_Id varchar (5) not null, constraint fax_pk primary key (Address_Fax), constraint addfax_fk foreign key (Address_Id) references Address (Address_Id));

```
SQL> Create table Address_Fax (Address_Fax int, Address_Id varchar (5) not null, constraint fax_pk primary key
  2  (Address_Fax), constraint addfax_fk foreign key (Address_Id) references Address (Address_Id));

Table created.
```

Figure 17: Creating Table Address_Fax.

5. Creating Table Person

Create Table Person (Person_Id varchar (5), Name varchar (60) not null, Age int not null, Gender varchar (10) not null, Address_Id varchar (5) not null, constraint per_pk primary key (Person_Id), constraint person_fk foreign key (Address_Id) references Address (Address_Id));

```
SQL> create table Person (Person_Id varchar(5), Name varchar(60) not null, Age int not null,
  2  Gender varchar(10) not null,Address_Id varchar(5) not null, constraint per_pk primary key
  3  (Person_Id), constraint person_fk foreign key (Address_Id) references Address (Address_Id));

Table created.
```

Figure 18: Creating Table Person.

6. Creating Table Person_Email

Create Table Person_Email (Email varchar (80), Person_Id varchar (5) not null, constraint em_pk primary key (Email), constraint email_fk foreign key (person_Id) references Person (Person_Id));

```
SQL> create table Person_Email (Email varchar(80), Person_Id varchar(5) not null,
  2  constraint em_pk primary key (Email), constraint email_fk foreign key (person_Id) references Person(Person_Id));

Table created.
```

Figure 19: Creating Table Person_Email.

7. Creating Table Person_Number

Create Table Person_Number (Cell_Number int, Person_Id varchar (5) not null, constraint n_pk primary key (Cell_Number), constraint no_fk foreign key (Person_Id) references Person (Person_Id));

```
SQL> create table Person_Number(Cell_Number int, Person_Id varchar(5) not null,
  2  constraint n_pk primary key (Cell_Number),
  3  constraint no_fk foreign key (Person_Id) references Person(Person_Id));

Table created.
```

Figure 20: Creating Table Person_Number.

8. Creating Table Staff

Create Table Staff(Staff_Id varchar (5),Staff_Type varchar (18) not null, Post varchar (15) not null, Salary int not null, Join_Date Date not null, constraint staff_pk primary key (Staff_Id), constraint stf_fk foreign key (Staff_Id) references Person(Person_Id));

```
SQL> create table Staff(Staff_Id varchar(5),Staff_Type varchar(18) not null,
  2  Post varchar(15) not null, Salary int not null, Join_Date Date not null,
  3  constraint staff_pk primary key (Staff_Id),
  4  constraint stf_fk foreign key (Staff_Id) references Person(Person_Id));

Table created.
```

Figure 21: Creating Table Staff.

9. Creating Table Patient

Create Table Patient (Patient_Id varchar (5), Patient_Type varchar (15) not null, Blood_Group varchar (5) not null, constraint pat_pk primary key (Patient_Id), constraint pat_fk foreign key (Patient_Id) references Person(Person_Id));

```
SQL> create table Patient(Patient_Id varchar(5),Patient_Type varchar(15) not null,
  2  Blood_Group varchar(5) not null,
  3  constraint pat_pk primary key (Patient_Id),
  4  constraint pat_fk foreign key (Patient_Id) references Person(Person_Id));

Table created.
```

Figure 22: Creating Table Patient.

10. Creating Table Ward

Create Table Ward (Ward_Id varchar (5), Ward_Type varchar (20) not null, Capacity int not null, constraint wd_pk primary key (Ward_Id));

```
SQL> create table Ward(Ward_Id varchar(5), Ward_Type varchar(20) not null, Capacity int not null,
  2  constraint wd_pk primary key (Ward_Id));

Table created.
```

Figure 23: Creating Table Ward.

11. Creating Table Appointment

Create Table Appointment (Appointment_Id varchar (5), Appoint_Date Date not null, Treatment_Fee int, Staff_Id varchar (5) not null, Patient_Id varchar (5) not null, Ward_Id varchar(5) not null, constraint ap_pk primary key (Appointment_Id), constraint appat_fk foreign key (Patient_Id) references Patient (Patient_Id), constraint apsat_fk foreign key (Staff_Id) references Staff (Staff_Id), constraint apwd_fk foreign key (Ward_Id) references ward(Ward_Id));

```
SQL> create table Appointment(Appointment_Id varchar(5), Appoint_Date Date not null,
  2  Treatment_Fee int, Staff_Id varchar(5) not null, Patient_Id varchar(5) not null,
  3  Ward_Id varchar(5) not null,constraint ap_pk primary key (Appointment_Id),
  4  constraint appat_fk foreign key (Patient_Id) references Patient(Patient_Id),
  5  constraint apsat_fk foreign key (Staff_Id) references Staff(Staff_Id),
  6  constraint apwd_fk foreign key (Ward_Id) references ward(Ward_Id));

Table created.
```

Figure 24: Creating Table Appointment.

3.4 Insert Statement

1) Inserting data in table Address.

Insert all into Address (Address_Id, Country, Province, State, Zone, City, Street, Street_Number) values ('A101','Nepal',1,'Purba','Koshi','Dharan','Fulbari',1) into Address (Address_Id, Country, Province, State, Zone, City, Street, Street_Number) values ('A202','Nepal',1,'Purba','Koshi','Dharan','Bagarkot',2) into Address (Address_Id, Country, Province, State, Zone, City, Street, Street_Number) values ('A303','Nepal',1,'Purba','Sunsari','Dharan','Kalopul',10) into Address (Address_Id, Country, Province, State, Zone, City, Street, Street_Number) values ('A404','Nepal',1,'Purba','Sunsari','Itahari','Homes',5) into Address (Address_Id, Country, Province, State, Zone, City, Street, Street_Number) values ('A505','Nepal',1,'Purba','Morang','Koshi','Dhulari',3) into Address (Address_Id, Country, Province, State, Zone, City, Street, Street_Number) values ('A606','Nepal',1,'Central','Bagmati','Katmandu','Chabel',9) into Address (Address_Id, Country, Province, State, Zone, City, Street, Street_Number) values ('A707','India',4,'Central','TamilNadu','Chennai','Madras',19) into Address (Address_Id, Country, Province, State, Zone, City, Street, Street_Number) values ('A808','Nepal',3,'Central','Bagmati','Kathmandu','Sukedhara',6) select * from dual;

```
SQL> insert all into Address(Address_Id,Country,Province, State, Zone, City, Street, Street_Number) values
  2  ('A101','Nepal',1,'Purba','Koshi','Dharan','Fulbari',1)
  3   into Address(Address_Id,Country,Province, State, Zone, City, Street, Street_Number) values
  4  ('A202','Nepal',1,'Purba','Koshi','Dharan','Bagarkot',2)
  5   into Address(Address_Id,Country,Province, State, Zone, City, Street, Street_Number) values
  6  ('A303','Nepal',1,'Purba','Sunsari','Dharan','Kalopul',10)
  7   into Address(Address_Id,Country,Province, State, Zone, City, Street, Street_Number) values
  8  ('A404','Nepal',1,'Purba','Sunsari','Itahari','Homes',5)
  9   into Address(Address_Id,Country,Province, State, Zone, City, Street, Street_Number) values
 10  ('A505','Nepal',1,'Purba','Morang','Koshi','Dhulari',3)
 11   into Address(Address_Id,Country,Province, State, Zone, City, Street, Street_Number) values
 12  ('A606','Nepal',1,'Central','Bagmati','Katmandu','Chabel',9)
 13   into Address(Address_Id,Country,Province, State, Zone, City, Street, Street_Number) values
 14  ('A707','India',4,'Central','TamilNadu','Chennai','Madras',19)
 15   into Address(Address_Id,Country,Province, State, Zone, City, Street, Street_Number) values
 16  ('A808','Nepal',3,'Central','Bagmati','Kathmandu','Sukedhara',6)
 17  select * from dual;

8 rows created.
```

Figure 25: Inserting data in table Address.

2) Inserting data in table Address_Email

insert all into Address_Email (Address_Email, Address_Id) values ('fulbari@gmail.com', 'A101')

into Address_Email (Address_Email, Address_Id) values ('dharanful@gmail.com', 'A101')

into Address_Email (Address_Email, Address_Id) values ('bagarkot@gmail.com', 'A202')

into Address_Email (Address_Email, Address_Id) values ('kalopul@gmail.com', 'A303')

into Address_Email (Address_Email, Address_Id) values ('homes@gmail.com', 'A404')

into Address_Email(Address_Email,Address_Id) values ('homes23@gmail.com', 'A404')

into Address_Email (Address_Email, Address_Id) values ('dhulari@gmail.com', 'A505')

into Address_Email (Address_Email, Address_Id) values ('chabel@gmail.com', 'A606')

into Address_Email (Address_Email, Address_Id) values ('madras@gmail.com', 'A707')

into Address_Email(Address_Email,Address_Id)values ('sukedhara@gmail.com', 'A808')

into Address_Email (Address_Email, Address_Id) values ('ktmsuk@gmail.com', 'A808')

select * from dual;

```
SQL> insert all into Address_Email (Address_Email, Address_Id) values ('fulbari@gmail.com','A101')
  2    into Address_Email (Address_Email, Address_Id) values ('dharanful@gmail.com','A101')
  3   into Address_Email (Address_Email, Address_Id) values ('bagarkot@gmail.com','A202')
  4    into Address_Email (Address_Email, Address_Id) values ('kalopul@gmail.com','A303')
  5    into Address_Email (Address_Email, Address_Id) values ('homes@gmail.com','A404')
  6    into Address_Email(Address_Email,Address_Id) values ('homes23@gmail.com','A404')
  7    into Address_Email (Address_Email, Address_Id) values ('dhulari@gmail.com','A505')
  8    into Address_Email (Address_Email, Address_Id) values ('chabel@gmail.com','A606')
  9    into Address_Email (Address_Email, Address_Id) values ('madras@gmail.com','A707')
 10   into Address_Email(Address_Email,Address_Id)values ('sukedhara@gmail.com','A808')
 11   into Address_Email (Address_Email, Address_Id) values ('ktmsuk@gmail.com','A808')
 12   select * from dual;

11 rows created.
```

Figure 26: Inserting data in table Address_Email.

3) Inserting data in table Address_Number.

insert into Address_Number values (9812345678, 'A101');

insert into Address_Number values (9823456789, 'A101');

insert into Address_Number values (9811223344, 'A404');

insert into Address_Number values (9811780223, 'A606');

insert into Address_Number values (9822780223, 'A606');

insert into Address_Number values (9812332111, 'A707');

insert into Address_Number values (9811780224, 'A808');

insert into Address_Number values (9811780225, 'A808');

```
SQL> insert into Address_Number values (9812345678,'A101');
1 row created.
SQL> insert into Address_Number values (9823456789,'A101');
1 row created.
SQL> insert into Address_Number values (9811223344,'A404');
1 row created.
SQL> insert into Address_Number values (9811780223,'A606');
1 row created.
SQL> insert into Address_Number values (9822780223,'A606');
1 row created.
SQL> insert into Address_Number values (9812332111,'A707');
1 row created.
SQL> insert into Address_Number values (9811780224,'A808');
1 row created.
SQL> insert into Address_Number values (9811780225,'A808');
1 row created.
```

Figure 27: Inserting data in table Address_Number.

4) Inserting data in table Address_Fax.

insert into Address_Fax values (12345678, 'A202');

insert into Address_Fax values (87654321, 'A303');

insert into Address_Fax values (98765432, 'A303');

insert into Address_Fax values (11223344, 'A505');

insert into Address_Fax values (44332211, 'A505');

insert into Address_Fax values (12131415, 'A707');

insert into Address_Fax values (51413121, 'A707');

insert into Address_Fax values (12344321, 'A808');

```
SQL> insert into Address_Fax values (12345678,'A202');
1 row created.
SQL> insert into Address_Fax values (87654321,'A303');
1 row created.
SQL> insert into Address_Fax values (98765432,'A303');
1 row created.
SQL> insert into Address_Fax values (11223344,'A505');
1 row created.
SQL> insert into Address_Fax values (44332211,'A505');
1 row created.
SQL> insert into Address_Fax values (12131415,'A707');
1 row created.
SQL> insert into Address_Fax values (51413121,'A707');
1 row created.
SQL> insert into Address_Fax values (12344321,'A808');
1 row created.
```

Figure 28: Inserting data in table Address_Fax

5) Inserting data in table Person.

```
insert into Person values ('P501','Cezal Gautam',20,'Female','A101');
insert into Person values ('P502','Karuna Gurung',22,'Female','A303');
insert into Person values ('P503','Amir Roy',32,'Male','A707');
insert into Person values ('P504','Nilu Kapoor',25,'Female','A707');
insert into Person values ('P505','Nil Gurung',22,'Male','A202');
insert into Person values ('P506','Sofiya Rana',20,'Female','A404');
insert into Person values ('P507','Raju Lama',23,'Male','A505');
insert into Person values ('P508','Aruna Tamang',29,'Female','A606');
insert into Person values ('P509','Rohan Magar',30,'Male','A808');
insert into Person values ('P510','Sneha Regmi',22,'Female','A404');
insert into Person values ('P511','Pujan Parsai',26,'Male','A606');
insert into Person values ('P512','Reshma Giri',27,'Female','A505');
insert into Person values ('P513','Sudip Rai',28,'Male','A303');
insert into Person values ('P514','Amrit Rai',29,'Male','A202');
insert into Person values ('P515','Jishu Gurung',24,'Female','A808');
```

```
SQL> insert into Person values ('P500','Girija Tamang',20,'Male','A101');
1 row created.
SQL> insert into Person values ('P501','Cezal Gautam',20,'Female','A101');
1 row created.
SQL> insert into Person values ('P502','Karuna Gurung',22,'Female','A303');
1 row created.
SQL> insert into Person values ('P503','Amir Roy',32,'Male','A707');
1 row created.
SQL> insert into Person values ('P504','Nilu Kapoor',25,'Female','A707');
1 row created.
SQL> insert into Person values ('P505','Nil Gurung',22,'Male','A202');
1 row created.
SQL> insert into Person values ('P506','Sofiya Rana',20,'Female','A404');
1 row created.
SQL> insert into Person values ('P507','Raju Lama',23,'Male','A505');
1 row created.
SQL> insert into Person values ('P508','Aruna Tamang',29,'Female','A606');
1 row created.
SQL> insert into Person values ('P509','Rohan Magar',30,'Male','A808');
1 row created.
SQL> insert into Person values ('P510','Sneha Regmi',22,'Female','A404');
1 row created.
SQL> insert into Person values ('P511','Pujan Parsai',26,'Male','A606');
1 row created.
SQL> insert into Person values ('P512','Reshma Giri',27,'Female','A505');
1 row created.
SQL> insert into Person values ('P513','Sudip Rai',28,'Male','A303');
1 row created.
SQL> insert into Person values ('P514','Amrit Rai',29,'Male','A202');
1 row created.
SQL> insert into Person values ('P515','Jishu Gurung',24,'Female','A808');
1 row created.
```

Figure 29: Inserting data in table Person.

6) Inserting data in Person_Number.

insert into Person_Number values (9812345678,'P500');

insert into Person_Number values (9812345679,'P501');

insert into Person_Number values (9812345670,'P502');

insert into Person_Number values (9812345671,'P503');

insert into Person_Number values (9812345672,'P504');

insert into Person_Number values (9812345673,'P509');

insert into Person_Number values (9812345674,'P500');

insert into Person_Number values (9812345675,'P503');

insert into Person_Number values (9812345676,'P505');

```
SQL> insert into Person_Number values (9812345678,'P500');
1 row created.
SQL> insert into Person_Number values (9812345679,'P501');
1 row created.
SQL> insert into Person_Number values (9812345670,'P502');
1 row created.
SQL> insert into Person_Number values (9812345671,'P503');
1 row created.
SQL> insert into Person_Number values (9812345672,'P504');
1 row created.
SQL> insert into Person_Number values (9812345673,'P509');
1 row created.
SQL> insert into Person_Number values (9812345674,'P500');
1 row created.
SQL> insert into Person_Number values (9812345675,'P503');
1 row created.
SQL> insert into Person_Number values (9812345676,'P505');
1 row created.
```

Figure 30: Inserting data in Person_Number.

7) Inserting data in Person_Email.

```
insert into Person_Email values ('girija@gmail.com','P500');
insert into Person_Email values ('cezal@gmail.com','P501');
insert into Person_Email values ('karuna@gmail.com','P502');
insert into Person_Email values ('amir@gmail.com','P503');
insert into Person_Email values ('nilu@gmail.com','P504');
insert into Person_Email values ('nil@gmail.com','P505');
insert into Person_Email values ('sofiya@gmail.com','P506');
insert into Person_Email values ('raju@gmail.com','P507');
insert into Person_Email values ('aruna@gmail.com','P508');
insert into Person_Email values ('rohan@gmail.com','P509');
insert into Person_Email values ('sneha@gmail.com','P510');
insert into Person_Email values ('pujan@gmail.com','P511');
insert into Person_Email values ('reshma@gmail.com','P512');
insert into Person_Email values ('sudip@gmail.com','P513');
insert into Person_Email values ('amrit@gmail.com','P514');
insert into Person_Email values ('jishu@gmail.com','P515');
```

```
SQL> insert into Person_Email values ('girija@gmail.com','P500');
1 row created.
SQL> insert into Person_Email values ('cezal@gmail.com','P501');
1 row created.
SQL> insert into Person_Email values ('karuna@gmail.com','P502');
1 row created.
SQL> insert into Person_Email values ('amir@gmail.com','P503');
1 row created.
SQL> insert into Person_Email values ('nilu@gmail.com','P504');
1 row created.
SQL> insert into Person_Email values ('nil@gmail.com','P505');
1 row created.
SQL> insert into Person_Email values ('sofiya@gmail.com','P506');
1 row created.
SQL> insert into Person_Email values ('raju@gmail.com','P507');
1 row created.
```

```
SQL> insert into Person_Email values ('aruna@gmail.com','P508');

1 row created.

SQL> insert into Person_Email values ('rohan@gmail.com','P509');

1 row created.

SQL> insert into Person_Email values ('sneha@gmail.com','P510');

1 row created.

SQL> insert into Person_Email values ('pujan@gmail.com','P511');

1 row created.

SQL> insert into Person_Email values ('reshma@gmail.com','P512');

1 row created.

SQL> insert into Person_Email values ('sudip@gmail.com','P513');

1 row created.

SQL> insert into Person_Email values ('amrit@gmail.com','P514');

1 row created.

SQL> insert into Person_Email values ('jishu@gmail.com','P515');

1 row created.
```

Figure 31: Inserting data in table Person_Email.

8) Inserting data in table Staff

insert into Staff values ('P503','Certified','Doctor',90000,'02-jan-2009');
insert into Staff values ('P500','Certified','Doctor',90000,'01-feb-2010');
insert into Staff values ('P504','Certified','Nurse',50000,'25-may-2010');
insert into Staff values ('P515','Certified','Nurse',50000,'10-dec-2010');
insert into Staff values ('P509','Certified','Assistant',70000,'01-nov-2009');
insert into Staff values ('P513','UnCertified','Doctor',70000,'10-nov-2010');
insert into Staff values ('P508','UnCertified','Nurse',40000,'23-aug-2010');
insert into Staff values ('P507','UnCertified','Assistant',30000,'16-feb-2010');
insert into Staff values ('P502','UnCertified','Doctor',40000,'28-mar-2009');
insert into Staff values ('P501','UnCertified','Assistant',60000,'15-dec-2010');

```
SQL> insert into Staff values ('P503','Certified','Doctor',90000,'02-jan-2009');

1 row created.

SQL> insert into Staff values ('P500','Certified','Doctor',90000,'01-feb-2010');

1 row created.

SQL> insert into Staff values ('P504','Certified','Nurse',50000,'25-may-2010');

1 row created.

SQL> insert into Staff values ('P515','Certified','Nurse',50000,'10-dec-2010');

1 row created.

SQL> insert into Staff values ('P509','Certified','Assistant',70000,'01-nov-2009');

1 row created.

SQL> insert into Staff values ('P513','UnCertified','Doctor',70000,'10-nov-2010');

1 row created.

SQL> insert into Staff values ('P508','UnCertified','Nurse',40000,'23-aug-2010');

1 row created.

SQL> insert into Staff values ('P507','UnCertified','Assistant',30000,'16-feb-2010');

1 row created.

SQL> insert into Staff values ('P502','UnCertified','Doctor',40000,'28-mar-2009');

1 row created.

SQL> insert into Staff values ('P501','UnCertified','Assistant',60000,'15-dec-2010');

1 row created.
```

Figure 32: Inserting data in table Staff.

9) Inserting data in Patient
insert into Patient values ('P500','New','O+');
insert into Patient values ('P504','Regular','A+');
insert into Patient values ('P513','Regular','B+');
insert into Patient values ('P505','Regular','O-');
insert into Patient values ('P508','New','AB+');
insert into Patient values ('P510','Regular','O+');
insert into Patient values ('P512','New','A+');
insert into Patient values ('P506','New','A+');
insert into Patient values ('P511','Regular','B+');

```
SQL> insert into Patient values ('P500','New','O+');

1 row created.

SQL> insert into Patient values ('P504','Regular','A+');

1 row created.

SQL> insert into Patient values ('P513','Regular','B+');

1 row created.

SQL> insert into Patient values ('P505','Regular','O-');

1 row created.

SQL> insert into Patient values ('P508','New','AB+');

1 row created.

SQL> insert into Patient values ('P510','Regular','O+');

1 row created.

SQL> insert into Patient values ('P512','New','A+');

1 row created.

SQL> insert into Patient values ('P506','New','A+');

1 row created.

SQL> insert into Patient values ('P511','Regular','B+');

1 row created.
```

Figure 33: Inserting data in table Patient.

10) Inserting data in table Ward

```
insert into Ward values ('WD1','Emergency',25);
insert into Ward values ('WD2','General',50);
insert into Ward values ('WD3','Maternity',30);
insert into Ward values ('WD4','Intensive Care',15);
insert into Ward values ('WD5','Regal Suite',5);
insert into Ward values ('WD6','Burns',20);
insert into Ward values ('WD7','Lab',10);
insert into Ward values ('WD8','Primary',35);
```

```
SQL> insert into Ward values ('WD1','Emergency',25);

1 row created.

SQL> insert into Ward values ('WD2','General',50);

1 row created.

SQL> insert into Ward values ('WD3','Maternity',30);

1 row created.

SQL> insert into Ward values ('WD4','Intensive Care',15);

1 row created.

SQL> insert into Ward values ('WD5','Regal Suite',5);

1 row created.

SQL> insert into Ward values ('WD6','Burns',20);

1 row created.

SQL> insert into Ward values ('WD7','Lab',10);

1 row created.

SQL> insert into Ward values ('WD8','Primary',35);

1 row created.
```

Figure 34: Inserting data in table Ward.

11) Inserting data in table Appointment.

```
insert into Appointment values ('AP10','02-feb-2019',5000,'P500','P505','WD1');
insert into Appointment values ('AP11','03-dec-2019',2500,'P504','P511','WD2');
insert into Appointment values ('AP12','13-nov-2019',0,'P503','P500','WD3');
insert into Appointment values ('AP13','03-nov-2019',0,'P509','P504','WD4');
insert into Appointment values ('AP14','15-aug-2019',2500,'P515','P508','WD5');
insert into Appointment values ('AP15','03-feb-2019',3000,'P501','P513','WD6');
insert into Appointment values ('AP16','23-jan-2019',5000,'P507','P510','WD7');
insert into Appointment values ('AP17','26-oct-2019',0,'P500','P504','WD8');
insert into Appointment values ('AP18','04-sep-2019',3000,'P502','P506','WD2');
insert into Appointment values ('AP19','05-oct-2019',2500,'P513','P512','WD1');
```

```
SQL> insert into Appointment values ('AP10','02-feb-2019',5000,'P500','P505','WD1');

1 row created.

SQL> insert into Appointment values ('AP11','03-dec-2019',2500,'P504','P511','WD2');

1 row created.

SQL> insert into Appointment values ('AP12','13-nov-2019',0,'P503','P500','WD3');

1 row created.

SQL> insert into Appointment values ('AP13','03-nov-2019',0,'P509','P504','WD4');

1 row created.

SQL> insert into Appointment values ('AP14','15-aug-2019',2500,'P515','P508','WD5');

1 row created.

SQL> insert into Appointment values ('AP15','03-feb-2019',3000,'P501','P513','WD6');

1 row created.

SQL> insert into Appointment values ('AP16','23-jan-2019',5000,'P507','P510','WD7');

1 row created.

SQL> insert into Appointment values ('AP17','26-oct-2019',0,'P500','P504','WD8');

1 row created.

SQL> insert into Appointment values ('AP18','04-sep-2019',3000,'P502','P506','WD2');

1 row created.

SQL> insert into Appointment values ('AP19','05-oct-2019',2500,'P513','P512','WD1');

1 row created.
```

Figure 35: Inserting data in table Appointment.

3.5 Select Statements

- Address Table

```
SQL> select * from address;

ADDRE COUNTRY                   PROVINCE STATE                                    ZONE                                      CITY
                        STREET                                     STREET_NUMBER
----- ------------------------ --------- ------------------------------------------------ ------------------------------------------------ ------------------------
---------------------- ------------------------------------------------ -------------
A101  Nepal                            1 Purba                                        Koshi                                     Dharan
                        Fulbari                                                     1
A202  Nepal                            1 Purba                                        Koshi                                     Dharan
                        Bagarkot                                                    2
A303  Nepal                            1 Purba                                        Sunsari                                   Dharan
                        Kalopul                                                    10
A404  Nepal                            1 Purba                                        Sunsari                                   Itahari
                        Homes                                                       5
A505  Nepal                            1 Purba                                        Morang                                    Koshi
                        Dhulari                                                     3
A606  Nepal                            1 Central                                      Bagmati                                   Katmandu
                        Chabel                                                      9
A707  India                            4 Central                                      TamilNadu                                 Chennai
                        Madras                                                     19
A808  Nepal                            3 Central                                      Bagmati                                   Kathmandu
                        Sukedhara                                                   6

8 rows selected.
```

Figure 36: Showing data of table Address.

- Address_Email Table

```
SQL> select * from Address_Email;

ADDRESS_EMAIL                                                                      ADDRE
-------------------------------------------------------------------------------- -----
fulbari@gmail.com                                                                  A101
dharanful@gmail.com                                                                A101
bagarkot@gmail.com                                                                 A202
kalopul@gmail.com                                                                  A303
homes@gmail.com                                                                    A404
homes23@gmail.com                                                                  A404
dhulari@gmail.com                                                                  A505
chabel@gmail.com                                                                   A606
madras@gmail.com                                                                   A707
sukedhara@gmail.com                                                                A808
ktmsuk@gmail.com                                                                   A808

11 rows selected.
```

Figure 37: Showing data of table Adddress_Email.

- Address_Number Table

```
SQL> select * from Address_Number;

ADDRESS_NUMBER ADDRE
-------------- -----
    9812345678 A101
    9823456789 A101
    9811223344 A404
    9811780223 A606
    9822780223 A606
    9812332111 A707
    9811780224 A808
    9811780225 A808

8 rows selected.
```

Figure 38: Showing data of table Address_Number.

- Address_Fax Table

```
SQL> select * from Address_Fax;

ADDRESS_FAX ADDRE
----------- -----
   12345678 A202
   87654321 A303
   98765432 A303
   11223344 A505
   44332211 A505
   12131415 A707
   51413121 A707
   12344321 A808

8 rows selected.
```

Figure 39: Showing data of table Address_Fax.

- Person Table

```
SQL> select * from person;

PERSO NAME                                                        AGE GENDER     ADDRE
----- ------------------------------------------------------------ --- ---------- -----
P500  Girija Tamang                                                 20 Male       A101
P501  Cezal Gautam                                                  20 Female     A101
P502  Karuna Gurung                                                 22 Female     A303
P503  Amir Roy                                                      32 Male       A707
P504  Nilu Kapoor                                                   25 Female     A707
P505  Nil Gurung                                                    22 Male       A202
P506  Sofiya Rana                                                   20 Female     A404
P507  Raju Lama                                                     23 Male       A505
P508  Aruna Tamang                                                  29 Female     A606
P509  Rohan Magar                                                   30 Male       A808
P510  Sneha Regmi                                                   22 Female     A404
P511  Pujan Parsai                                                  26 Male       A606
P512  Reshma Giri                                                   27 Female     A505
P513  Sudip Rai                                                     28 Male       A303
P514  Amrit Rai                                                     29 Male       A202
P515  Jishu Gurung                                                  24 Female     A808

16 rows selected.
```

Figure 40: Showing data of table Person.

- Person_Number Table

```
SQL> select * from Person_Number;

CELL_NUMBER PERSO
----------- -----
 9812345678 P500
 9812345679 P501
 9812345670 P502
 9812345671 P503
 9812345672 P504
 9812345673 P509
 9812345674 P500
 9812345675 P503
 9812345676 P505

9 rows selected.
```

Figure 41: Showing data of table Person_Number

- Person_Email Table

```
SQL> select * from Person_Email;

EMAIL                                                                    PERSO
------------------------------------------------------------------------ -----
girija@gmail.com                                                         P500
cezal@gmail.com                                                          P501
karuna@gmail.com                                                         P502
amir@gmail.com                                                           P503
nilu@gmail.com                                                           P504
nil@gmail.com                                                            P505
sofiya@gmail.com                                                         P506
raju@gmail.com                                                           P507
aruna@gmail.com                                                          P508
rohan@gmail.com                                                          P509
sneha@gmail.com                                                          P510
pujan@gmail.com                                                          P511
reshma@gmail.com                                                         P512
sudip@gmail.com                                                          P513
amrit@gmail.com                                                          P514
jishu@gmail.com                                                          P515

16 rows selected.
```

Figure 42: Showing data of table Person_Email.

- Staff Table

```
SQL> select * from staff;

STAFF STAFF_TYPE         POST                SALARY JOIN_DATE
----- ----------------- --------------- ---------- ---------
P503  Certified         Doctor               90000 02-JAN-09
P500  Certified         Doctor               90000 01-FEB-10
P504  Certified         Nurse                50000 25-MAY-10
P515  Certified         Nurse                50000 10-DEC-10
P509  Certified         Assistant            70000 01-NOV-09
P513  UnCertified       Doctor               70000 10-NOV-10
P508  UnCertified       Nurse                40000 23-AUG-10
P507  UnCertified       Assistant            30000 16-FEB-10
P502  UnCertified       Doctor               40000 28-MAR-09
P501  UnCertified       Assistant            60000 15-DEC-10

10 rows selected.
```

Figure 43: Showing data of table staff.

- Patient Table

```
SQL> select * from patient;

PATIE PATIENT_TYPE    BLOOD
----- --------------- -----
P500  New             O+
P504  Regular         A+
P513  Regular         B+
P505  Regular         O-
P508  New             AB+
P510  Regular         O+
P512  New             A+
P506  New             A+
P511  Regular         B+

9 rows selected.
```

Figure 44: Showing data of table Patient.

- Ward Table

```
SQL> select * from ward;

WARD_ WARD_TYPE              CAPACITY
----- -------------------- ----------
WD1   Emergency                    25
WD2   General                      50
WD3   Maternity                    30
WD4   Intensive Care               15
WD5   Regal Suite                   5
WD6   Burns                        20
WD7   Lab                          10
WD8   Primary                      35

8 rows selected.
```

Figure 45: Showing data of table Ward.

- Appointment Table

```
SQL> select * from Appointment;

APPOI APPOINT_D TREATMENT_FEE STAFF PATIE WARD_
----- --------- ------------- ----- ----- -----
AP10  02-FEB-19          5000 P500  P505  WD1
AP11  03-DEC-19          2500 P504  P511  WD2
AP12  13-NOV-19             0 P503  P500  WD3
AP13  03-NOV-19             0 P509  P504  WD4
AP14  15-AUG-19          2500 P515  P508  WD5
AP15  03-FEB-19          3000 P501  P513  WD6
AP16  23-JAN-19          5000 P507  P510  WD7
AP17  26-OCT-19             0 P500  P504  WD8
AP18  04-SEP-19          3000 P502  P506  WD2
AP19  05-OCT-19          2500 P513  P512  WD1

10 rows selected.
```

Figure 46: Showing data of table Appointment.

3.6 Information Queries

1. List all patients, regular and new

   Select Patient_Id, Patient_Type, Name, Gender from Patient,Person where Patient.Patient_Id = Person.Person_Id;

```
SQL> select Patient_Id,Patient_Type,Name,Gender from Patient,Person where
  2  Patient.Patient_Id =Person.Person_Id;

PATIE PATIENT_TYPE    NAME                                                       GENDER
----- --------------- ---------------------------------------------------------- ----------
P500  New             Girija Tamang                                              Male
P504  Regular         Nilu Kapoor                                                Female
P505  Regular         Nil Gurung                                                 Male
P506  New             Sofiya Rana                                                Female
P508  New             Aruna Tamang                                               Female
P510  Regular         Sneha Regmi                                                Female
P511  Regular         Pujan Parsai                                               Male
P512  New             Reshma Giri                                                Female
P513  Regular         Sudip Rai                                                  Male

9 rows selected.
```

Figure 47: List all patients, regular and new

2. List all patients with all their addresses.
   Select Patient.Patient_Id, Person.Name, Address. * from Patient inner join Person on Person.Person_Id = Patient.Patient_Id join Address on Address.Address_Id = Person.Address_Id;

```
SQL> Select Patient.Patient_Id, Person.Name, Address. *  from Patient inner join Person
  2  on Person.Person_Id = Patient.Patient_Id join Address on Address.Address_Id =
  3  Person.Address_Id;

PATIE NAME                                                 ADDRE COUNTRY                   PROVINCE STATE                                                 ZONE
                                       CITY                                     STREET                                          STREET_NUMBER
----- ---------------------------------------------------- ----- ------------------------- ---------- ---------------------------------------------- ----------
-------------------------------------- ---------------------------------------------------- ---------------------------------------------------- -------------
P500  Girija Tamang                                        A101  Nepal                                1 Purba                                                Koshi
                                       Dharan                                   Fulbari                                                      1
P505  Nil Gurung                                           A202  Nepal                                1 Purba                                                Koshi
                                       Dharan                                   Bagarkot                                                     2
P513  Sudip Rai                                            A303  Nepal                                1 Purba                                                Sunsari
                                       Dharan                                   Kalopul                                                     10
P510  Sneha Regmi                                          A404  Nepal                                1 Purba                                                Sunsari
                                       Itahari                                  Homes                                                        5
P506  Sofiya Rana                                          A404  Nepal                                1 Purba                                                Sunsari
                                       Itahari                                  Homes                                                        5
P512  Reshma Giri                                          A505  Nepal                                1 Purba                                                Morang
                                       Koshi                                    Dhulari                                                      3
P511  Pujan Parsai                                         A606  Nepal                                1 Central                                              Bagmati
                                       Katmandu                                 Chabel                                                       9
P508  Aruna Tamang                                         A606  Nepal                                1 Central                                              Bagmati
                                       Katmandu                                 Chabel                                                       9
P504  Nilu Kapoor                                          A707  India                                4 Central                                              TamilNadu
                                       Chennai                                  Madras                                                      19

9 rows selected.
```

Figure 48: List all patients with all their addresses.

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

   Select St. Staff_Id, Pr. Name, Ap. Appointment_Id, Ap. Appoint_Date, Ap. Treatment_Fee as "Amount" from Staff St join Person Pr on St.Staff_Id = Pr. Person_Id join Appointment Ap on St.Staff_Id=Ap.Staff_Id where St. Staff_Id = 'P500' ;

```
SQL>  Select St.Staff_Id,Pr.Name,Ap.Appointment_Id,Ap.Appoint_Date,Ap.Treatment_Fee as "Amount"
  2    from Staff St join Person Pr on St.Staff_Id = Pr.Person_Id
  3    join Appointment Ap on St.Staff_Id=Ap.Staff_Id
  4    where St.Staff_Id='P500';

STAFF NAME                                                       APPOI APPOINT_D     Amount
----- ---------------------------------------------------------- ----- --------- ----------
P500  Girija Tamang                                              AP10  02-FEB-19       5000
P500  Girija Tamang                                              AP17  26-OCT-19          0
```

Figure 49: For a given certified doctor, find all the appointments he/she have conducted and the amount he/she got for conducting the appointment.

4. List all staffs that are also a patient.

   Select staf.staff_Id, Prsn.Name from Staff Staf join Patient Pt on Staf.Staff_Id = Pt.Patient_Id join Person Prsn on Staf.Staff_Id = Prsn.Person_Id;

```
SQL> Select staf.staff_Id, Prsn.Name from Staff Staf join Patient Pt on Staf.Staff_Id =
  2  Pt.Patient_Id join Person Prsn on Staf.Staff_Id = Prsn.Person_Id;

STAFF NAME
----- ------------------------------------------------------------
P500  Girija Tamang
P504  Nilu Kapoor
P508  Aruna Tamang
P513  Sudip Rai
```

Figure 50:List all staffs that are also a patient.

3.7 Transaction Queries

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

   Select St.Staff_Id, Pr.Name, Ap.Appointment_Id ,Ap.Appoint_Date, Ap.Treatment_Fee as "Charge" from Staff St join person Pr on St.Staff_Id = Pr.Person_Id join Appointment Ap on St.Staff_Id = Ap.Patient_Id where St.Staff_Type = ’UnCertified’ and St.Post = 'Doctor' ;

```
SQL> Select St.Staff_Id,Pr.Name,Ap.Appointment_Id,Ap.Appoint_Date,Ap.Treatment_Fee as "Charge"
  2    from Staff St join person Pr on St.Staff_Id=Pr.Person_Id
  3    join Appointment Ap on St.Staff_Id=Ap.Patient_Id
  4    where St.Staff_Type=’UnCertified’ and St.Post='Doctor';

STAFF NAME                                                     APPOI APPOINT_D     Charge
----- ------------------------------------------------------- ----- --------- ----------
P513  Sudip Rai                                                AP15  03-FEB-19       3000
```

Figure 51: List all uncertified doctors who have been attended an appointment for a treatment and the amount he/she have paid.

2. List the appointments that have been conducted in an emergency ward.

   Select Pt.Patient_Id,Ap.Appointment_Id,Pr.Name,Wd.Ward_Type from Patient Pt join Person Pr on Patient_Id = Pr.Person_Id join Appointment Ap on Pt.Patient_Id = Ap.Patient_Id join Ward Wd on Ap.Ward_Id = Wd.Ward_Id where Wd.Ward_Type = 'Emergency';

```
SQL> Select Pt.Patient_Id,Ap.Appointment_Id,Pr.Name,Wd.Ward_Type
  2  from Patient Pt join Person Pr on Patient_Id = Pr.Person_Id
  3  join Appointment Ap on Pt.Patient_Id=Ap.Patient_Id
  4  join Ward Wd on Ap.Ward_Id=Wd.Ward_Id
  5  where Wd.Ward_Type='Emergency';

PATIE APPOI NAME                                                    WARD_TYPE
----- ----- ------------------------------------------------------- --------------------
P505  AP10  Nil Gurung                                              Emergency
P512  AP19  Reshma Giri                                             Emergency
```

Figure 52:List the appointments that have been conducted in an emergency ward.

3. List all staffs (certified and uncertified) who have conducted or will conduct an appointment on a given date.

   Select Person. Name, Staff. Staff_Id, Staff. Staff_Type, Staff. Post, Appointment. Appoint_Date from Person Person join staff on Person. Person_Id = Staff. Staff_id join Appointment on Staff. Staff_Id = Appointment. Staff_Id where Appointment. Appoint_Date = ’02-feb-2019’;

```
SQL> Select Person.Name, Staff.Staff_Id, Staff.Staff_Type, Staff.Post,
  2  Appointment.Appoint_Date from Person Person join staff on Person.Person_Id =
  3  Staff.Staff_id join Appointment on Staff.Staff_Id = Appointment.Staff_Id where
  4  Appointment.Appoint_Date = ’02-feb-2019’;

NAME                                                      STAFF STAFF_TYPE         POST            APPOINT_D
--------------------------------------------------------- ----- ------------------ --------------- ---------
Girija Tamang                                             P500  Certified          Doctor          02-FEB-19
```

Figure 53:List all staffs (certified) who have conducted or will conduct an appointment on a given date.

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

   Select Pt. Patient_Id, Ap. Appointment_Id, Pr. Name, Ap. Appoint_Date from Patient Pt join Person Pr on Pt. Patient_Id = Pr. Person_Id join Appointment Ap on Pt. Patient_Id = Ap. Patient_Id where Ap. Appoint_Date = '13-Nov-2019';

```
SQL> select Pt.Patient_Id,Ap.Appointment_Id,Pr.Name,Ap.Appoint_Date from Patient Pt
  2   join Person Pr on Pt.Patient_Id = Pr.Person_Id
  3   join Appointment Ap on Pt.Patient_Id = Ap.Patient_Id
  4   where Ap.Appoint_Date ='13-Nov-2019';

PATIE APPOI NAME                                                      APPOINT_D
----- ----- ------------------------------------------------------- ---------
P500  AP12  Girija Tamang                                             13-NOV-19
```

Figure 54: List all patients booked for an appointment on a given date.

3.8 Creating Dump File

```
Select Administrator: Command Prompt
(c) 2019 Microsoft Corporation. All rights reserved.

C:\WINDOWS\system32>d:

D:\> Exp GirijaTamang_18030995/girija file = databasecw.dmp

Export: Release 11.2.0.2.0 - Production on Sun Dec 29 13:13:55 2019

Copyright (c) 1982, 2009, Oracle and/or its affiliates.  All rights reserved.


Connected to: Oracle Database 11g Express Edition Release 11.2.0.2.0 - Production
Export done in WE8MSWIN1252 character set and AL16UTF16 NCHAR character set
server uses AL32UTF8 character set (possible charset conversion)
. exporting pre-schema procedural objects and actions
. exporting foreign function library names for user GIRIJATAMANG_18030995
. exporting PUBLIC type synonyms
. exporting private type synonyms
. exporting object type definitions for user GIRIJATAMANG_18030995
About to export GIRIJATAMANG_18030995's objects ...
. exporting database links
. exporting sequence numbers
. exporting cluster definitions
. about to export GIRIJATAMANG_18030995's tables via Conventional Path ...
. . exporting table                       ADDRESS          8 rows exported
EXP-00091: Exporting questionable statistics.
EXP-00091: Exporting questionable statistics.
. . exporting table                 ADDRESS_EMAIL         11 rows exported
EXP-00091: Exporting questionable statistics.
EXP-00091: Exporting questionable statistics.
. . exporting table                  ADDRESS_FAX          8 rows exported
EXP-00091: Exporting questionable statistics.
EXP-00091: Exporting questionable statistics.
. . exporting table                ADDRESS_NUMBER          8 rows exported
EXP-00091: Exporting questionable statistics.
EXP-00091: Exporting questionable statistics.
. . exporting table                  APPOINTMENT         10 rows exported
EXP-00091: Exporting questionable statistics.
EXP-00091: Exporting questionable statistics.
. . exporting table                       PATIENT          9 rows exported
```

```
EXP-00091: Exporting questionable statistics.
EXP-00091: Exporting questionable statistics.
. . exporting table                         PERSON         16 rows exported
EXP-00091: Exporting questionable statistics.
EXP-00091: Exporting questionable statistics.
. . exporting table                    PERSON_EMAIL         16 rows exported
EXP-00091: Exporting questionable statistics.
EXP-00091: Exporting questionable statistics.
. . exporting table                   PERSON_NUMBER          9 rows exported
EXP-00091: Exporting questionable statistics.
EXP-00091: Exporting questionable statistics.
. . exporting table                          STAFF         10 rows exported
EXP-00091: Exporting questionable statistics.
EXP-00091: Exporting questionable statistics.
. . exporting table                           WARD          8 rows exported
EXP-00091: Exporting questionable statistics.
EXP-00091: Exporting questionable statistics.
. exporting synonyms
. exporting views
. exporting stored procedures
. exporting operators
. exporting referential integrity constraints
. exporting triggers
. exporting indextypes
. exporting bitmap, functional and extensible indexes
. exporting posttables actions
. exporting materialized views
. exporting snapshot logs
. exporting job queues
. exporting refresh groups and children
. exporting dimensions
. exporting post-schema procedural objects and actions
. exporting statistics
Export terminated successfully with warnings.
```

Figure 55: Creating dump file.

4. Critical Evaluation

The project has been successfully completed due to enough preparation, adequate design, careful attention and a lot of research. If I look back at the creation journey of Patient Recording System database for Syangbo International Hospital, it was not easy enough.

While developing this project, I had tackled the case scenario and identify the entities and attributes for developing patient recording system database. I had done research on hospital management system for identifying entities and attributes that can develop a database as per the scenario. After identifying the entities for database, I got stuck on creating a relationship between the entities. After consulting model teacher, I had developed an initial relationship diagram. The main problem while doing this course work was normalization. While normalizing the table I had faced problems like identifying unique key, problem in finding partial functional dependency and checking transitive dependency. After doing research and taking help form model leader, I was able to tackle the problems of normalization. While solving sql queries I had to face problem like no showing data and error commands. After getting concept of executing queries from lecture slide and web pages I was able to produce sql queries.

After completing this coursework with a lot of research and help from model teacher and friends I have gained knowledge on relational database system. I got chance to learn about types of attributes and entities, entity relationship diagram, concept of normalization, and sql queries. I had gained knowledge to remove data anomalies and its important in database system.

5. Critical Assessment of coursework

As we all know this is the era of technology where database is an important content for every people as well as companies and organizations. It plays a vital role for recording, managing, updating data. All kinds of organization and companies uses different kind of database for recoding their valuable data and information. The project that I had developed also related with database. I found my course work very interesting, challenging and useful related to database.

This database module is helpful to another model because it provides the basic concept of handling data process and its importance. This course has shown me to create a system that effectively deals with an entity with a lesser measure of peculiarities imaginable in the control of information. This module helps to develop framework which helps in developing application and software and ideas for handling information which helps in model like software engineering, java for developing software that deals with storing information.

This coursework is related about database and creating a database that stores the record of patient. The skill and knowledge gained while doing this coursework is creating or making a framework that deals with an organization effectively with lesser measure of oddities conceivable in information control. It has given me the aptitude to break down the case in appropriate way. I have gained knowledge on normalization, creating database and executing queries which helps in choosing career like data analyst, information system analysist etc in future.

6. References

Ahlawat, A., 2009. *Student Tonignt.* [Online] Available at: https://www.studytonight.com/dbms/database-normalization.php [Accessed 26 Dcember 2019].

Gupta, S. & Gupta, G., 2012. *Database Management System.* 4th ed. Delhi: Khanna Book Publishing Co.(P) Ltd..

Saxon, C., 2018. *How to Create Users, Grant Them Privileges, and Remove Them in Oracle Database.* [Online] Available at: https://blogs.oracle.com/sql/how-to-create-users-grant-them-privileges-and-remove-them-in-oracle-database
[Accessed 25 December 2019].

Singh, K. & Mandeep, B. S., 2008. *Advance Database.* 3rd ed. Delhi: Khanna Book publishing Co.(P) Ltd.

Vaibhav, B. & Ashima, B. B., 2019. *Database Management System.* 2nd ed. Rohini: NarosaPublishingHouse Pvt.Ltd..

W3school.com, 1999. *w3school.com.* [Online] Available at: https://www.w3schools.com/sql/default.asp [Accessed 29 December 2019].