LONDON METROPOLITAN UNIVERSITY

islington college
(इस्लिङ्टन कलेज)

Module Code & Module Title

CC5051NA Database Systems Development

Assessment Weightage & Type

50% Individual Coursework

Year and Semester

2019-20 Autumn

Student Name:Krishna budhathoki

London Met ID: 18029976

College ID: np01cp4a180172

Assignment Due Date: 2019-12-30

Assignment Submission Date: 2019-12-30

Title (Where Required): Patient Recording System

Word Count (Where Required):7237

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

Table of Tables

Introduction

Vayodha is well known hospital which is known for its multi-specialty hospital with vision of “cure with care”. In Sanskrit, Vayodha means having good health. All things are used for patient with a care and helpful manner. The hospital has showcasing quality care–24 hours a day. Where in Vayodha's symbol 'V' symbolizes loving hands and Life is symbolized by the Green droplet.

Vayodha Hospitals is a multi-specialty hospital with the dream of "Cure with Treatment" situated in the central orbit of the Kathmandu ring road. It not only offers a wide range of health services from primary care to state-of - the-art therapies for severe and unusual health problems, but also strives to become a national leader in patient-centered care committed to providing preventive and curative care in the world class.

With medical facilities with a total capacity of 50 beds, many of their services account for the overall clinical results. They are also the nation's first hospital to have a transition in the "Wheels to Wings" rescue mode with rooftop helipad. Our health services system requires not only the delivery of a range of health facilities in conjunction with developments but also extend medical evacuation from every corner of the nation.

The mission of Vayodha Hospitals is to become a national leader in patient-centered health care, medical research and policy reform for health care. At Vayodha, they ensure that patients are in safe hands as their health conditions are covered under the legion of seasoned doctors and specialists who have come of age and experience working together to fulfill each patient's health care needs.

Current Business Activities and Operations

Vayodha ensures that patients are well cared and feel comfortable in the hands of the hospital. The patients are first requested to provide their personal information such as name, age, date of birth, country. Province, city, street, street.no, fax, phone.no, cell.no which allows them to directly communicate with the hospital for the update of their health. It also allows a proper appointment confirmation for the treatment which allows the patient to know in advance with whom he/she is going to have treatment. When appointment is confirmed, the patient is notified about the appointment with appointment date and time and other information such as ward name, floor. Then the patient is requested to pay the bill for the appointment. Then the patient is shifted toward where he/she is treated by the specified hospital staffs.

Sales activities:

- Maximized quality of customer service
- Complaint management
- Preventive maintenance schedule
- Strengthened marketing and sales

Management activities:

- Patient care.
- Clean sanitation for hospital staffs
- Ward Sanitation and Provision of Therapeutic environment.
- Proper Supply and hospital equipment
- Interpretation of policies and procedures.
- Well managed budget
- Evaluation of each employees

Current Business Rules

Some of the business rules that I designed and formatted are listed as follows:

Rules for patients

- A patient can request many appointments as per needed.
- One appointment is specified for only one treatment.
- One bill is requested to be paid for one appointment
- The patient should provide all the necessary information required before the treatment

Rules of doctors

- At least one staff is required for one treatment
- When staff becomes patient, the certified staff is given free treatment whereas uncertified staff is required to pay for the treatment
- The staff is given a commission from each treatment of the patients

Assumption

- A specified reason is needed for appointment to be held by the patient.
- Only one treatment is possible for one appointment and one bill is generated.
- There can be many staffs attending for the treatment of one patient.
- There can be many contact details for one address
- For one treatment, one ward is used but many treatments can be held in one ward.
- Both patient and staff can have multiple addresses

Identification of Entities and Attributes.

An entity is an object or any item about which the data is taken and stored in the form of properties, tables. It is the key element in the relational database. Entities can be anything such as customer, email address, contact, patient, appointment. Each entity should have a specific name that defines the entity because it represents the table.

After proper understanding of coursework and researching, I was able to make following entities:

- Patient
- Appointment
- Staff

In database management system, an attribute is a component of database which carries a characteristic of the entity which helps in understanding the database. Attributes plays an important role in including the details carried by the entities in a initial ER diagram. Attributes describe the instances in the row of a database. Each entity holds its own data type.

The list of entities along with their respective attributes are shown in the below table

Entity	Attribute	Datatypes
Patient	Patient_id	VARCHAR2(6)
	Staff_id	VARCHAR2(6)
	P_name	VARCHAR2(25)
	age	NUMBER
	Country	VARCHAR2(25)
	Province	VARCHAR2(25)
	City	VARCHAR2(25)

	Street	VARCHAR2(25)
	Street_no	Number
	P_phone_no	Number
	Fax	NUMBER
	Cell.no	Number
	email	VARCHAR2(25)
	DOB	VARCHAR2(25)
	Status	VARCHAR2(9)
	Gender	VARCHAR2(6)

Table 1 patient entity

Entity	Attribute	Datatypes
Appointment	appointment_id	VARCHAR2(6)
	appointment_date	date
	time	VARCHAR2(8)
	Ward_no	VARCHAR2(6)
	ward name	VARCHAR2(10)
	floor	Number
	treatment_id	VARCHAR2(6)
	treatment	VARCHAR2(25)
	treatment type	VARCHAR2(25)
	bill_id	VARCHAR2(6)
	bill_Date	date
	paymentmethod	VARCHAR2(10)
	amount	Number

Table 2 Appointment entity

Entity	Attribute	Datatypes
	staff_id	VARCHAR2(6)
	s-name	VARCHAR2(25)
	S_DOB	date
	S_age	NUMBER (2)
	category	VARCHAR2(8)
	certified	Boolean
	staff_commission	Number

Staff	speciality	VARCHAR2(8)
	S_country	VARCHAR2(25)
	s_province	VARCHAR2(25)
	S_city	VARCHAR2(25)
	S_street	VARCHAR2(25)
	S_street_no	VARCHAR2(10)
	S_phone_no	Number (10)
	S_fax	NUMBER (10)
	S_cell_no	Number (10)
	S_email	VARCHAR2(25)

Table 3 Staff Entity

Initial er diagram

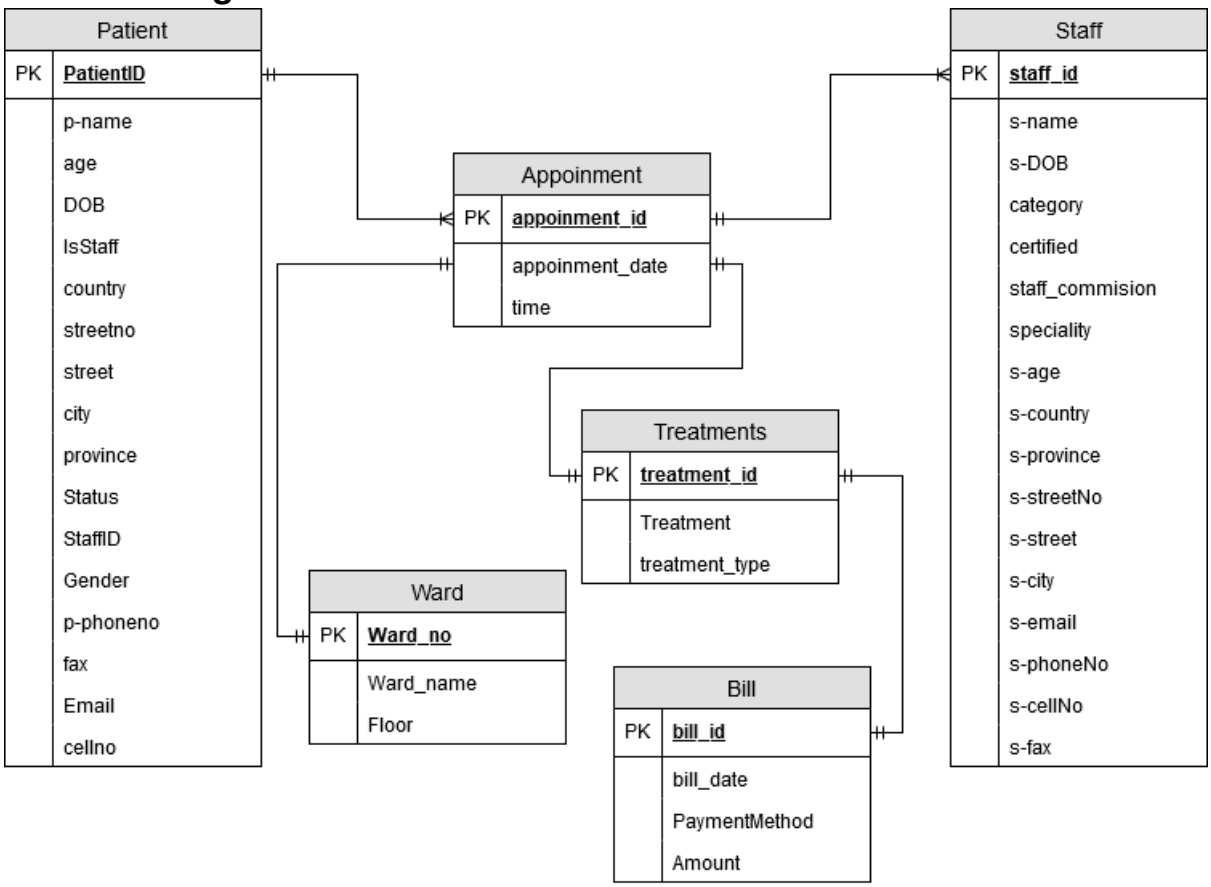

Figure 1 Initial ER diagram

Normalization

Normalization is a database design technique which is used for organizing the tables and minimizing the redundancy and dependency of data. It helps in dividing the larger table into smaller table and links them into different tables giving them relationships between each table. It is necessary to normalize the tables because it manages each table and relates the entities in a proper way without making mistakes in the entities and attributes.

The advantages of normalization are as follows:

- It eliminates the duplicate data. It reduces the size of the database and saves time and money
- It provides better performance. As the size gets smaller, flow of data becomes faster and shorter improving the response time and period
- There will be less errors as redundancy and dependency is minimized
- It also ensures faster creating, inserting and maintenance of tables
- It also gives the idea of joining only the tables which have relations and are needed.

Un-normalized form (UNF):

UNF is a simple database model which lacks the efficiency of flow of the data in database. An un-normalized form usually makes redundancy and dependency of data. It is also a single entity to define the table as all the attributes is stored in single entity.

I have shown list of entities with their respective attributes in the above table which is used for making the un-normalized form.

Patient (patient_id, Staff_id, p-name, age, DOB, isStaff, status, gender, {country, province, city, street, street.no {p-phone.no, fax, cell.no, email}} {appointment_id,Appointment_Date, time, ward.no, ward name, floor, treatment_id, treatment, treatment type, bill_id, bill_Date, paymentmethod, amount, {staff_id, s_name, s_DOB, category, certified, staff_commission, speciality {s_country, s_province, s_city, s_street, s_street.no, {s_phone.no, s_fax, s_cell.no, s_email}})

In the above UNF, the attributes which are data of patient information are not in repeating data except the address and mailing data. For the address of patient and staff the address is repeating, and contacts are repeating inside the address. From the scenario of patient, the patient can have multiple appointments, so the appointment entity is in repeating data. Same process is applied for staff as many staff is used for checking single patients.

First normal form:

The first normal form of normalization is the process of separating the repeating data from each group and sorting them in another form of table. Each repeating group is separated from its previous entity and a primary key is assigned for the specific entity which will represent the whole entity.

The repeating group is separated in the following stepwise process.

1st step:

Patient → (**patient_id**, staff_id , p-name, age, DOB, isStaff, status, gender, {country, province, city, street, street.no, {p-phone.no, fax, cell.no, e-mail})

Appointment → (**appointment_id, patient_id*,** appointment_date, time, ward.no, ward name, floor, treatment_id, treatment, treatment type, bill_id, bill_Date, ,paymentmethod, amount)

Staff → (**staff_id, appointment_id*,patient_id*,** s-name, s-DOB, category, certified, speciality, staff_commission ,s-age, {s-country, s-province, s-city, s-street, s-street.no, {s-phone.no, s-email, s-fax, s-cell.no})

In the above step the first entity is separated from its repeating data such as patient, appointment are separated into different entities. Patient_id is assigned foreign key in each separated entity as shown in above step. Then staff is sepearated from appointment.

2nd step:

Patient → (**patient_id,** staff_id , p-name, age, DOB, isStaff, status, gender)

Address → (**a_id, patient_id*,** country, province, city, street, street.no, {p-phone.no, fax, cell.no, e-mail})

Appointment → (**appointment_id, patient_id*,** appointment_date, time, ward.no, ward name, floor, treatment_id,treatment, treatment type, bill_id, bill_Date, ,paymentmethod, amount)

Staff → (**staff_id, appointment_id*, patient_id*,** s-name, s-DOB, category, certified, staff_commission, speciality, s-age)

s-address → (**s_a_id, appointment_id*, patient_id*, staff_id*,** s-country, s-province, s-city, s-street, s-street.no, {s-phone.no, s-email, s-fax, s-cell.no})

In the above step 2, the address entity is separated from the patient entity as patient has repeating address which is same case for staff

3rd step:

Patient (**patient_id,** staff_id , p-name, age, DOB, isStaff, status, gender)

Address (**a_id, patient_id*,** country, province, city, street, street.no)

Contact (**con_id, a_id*, patient_id*,** p-phone.no, fax, cell.no, e-mail)

Appointment → (**appointment_id, patient_id*,** appointment_date, time, ward.no, ward name, floor, treatment_id,treatment, treatment type, bill_id, bill_Date, ,paymentmethod, amount)

Staff (**staff_id, appointment_id*,patient_id*,** s-name, s-DOB, category, certified, staff_commission, speciality, s-age)

s-address (**s_a_id, appointment_id*,patient_id*, staff_id*,** s-country, s-province, s-city, s-street, s-street.no)

s-contact (**s_con_id, appointment_id*,s_a_id*, patient_id*, staff_id*,** s-phone.no, s-email, s-fax, s-cell.no)

in the above step 3, the contact for both patient and staff are repeating data for address so the contact entity is separated from address assigning the foreign key address, patient_id and staff_id for s-contact whereas for patient contact only a_id and patient_id is assigned foreign key.

Second normal form

The second normal form of normalization is the process of removing the partial dependencies in an entity. .

Step wise procedure

For the patient entity:

Patient (**patient_id**, staff_id , p-name, age, DOB, isStaff, status, gender)

In the patient entity there is only one composite key ignoring the staff_id because it does not hold any foreign key attributes. So the patient entity is already in 2nf.

For patient address entity:

Address (**a_id, patient_id*,** country, province, city, street, street.no)

In the address entity a_id and patient_id acts as two key attributes forming a composite key,so to remove the data redundancy, I removed unnecessary key tables which shows the nature of partial dependency.

Using the 2^n-1, we know that

2^2-1=4-1=3

So there will be three possible tables generated from patient address.

Now checking the partial dependency,

A_id → country, province, city, street, street.no

a_id, patient_id→

patient_id→

so the patient_id does not give any relation of patient address, so we remove the patient_id entity and use a_id entity and a_id, patient_id for constructing the table.

So the list of entities for patient address are as follows:

address (**a_id**, country, province, city, street, street.no)

address-bridge **(a_id*, patient_id***)

For patient contact entity:

Contact (**con_id, a_id*, patient_id*,** p-phone.no, fax, cell.no, e-mail)

In the contact entity con_id , patient_id, a_id acts as key attributes forming a composite key,so to remove the data redundancy, I removed unnecessary key tables which shows the nature of partial dependency.

For checking the list of possible entities

Using 2^n-1,

2^3-1=8-1=7

So there are seven possible entities to be made. Now checking the partial dependency.

Patient_id, a_id, con_id →

Patient_id, a_id →

A_id, con_id→

Patient_id, Con_id →

Patient_id →

A_id→

Con_id→p-phone.no, fax, cell.no, e-mail

So the patient_id with a_id and patient_id does not show any relation of contact of patient so all the entities with similar scenarios are eliminated. Therefore only two entities are left which are as follows:

Contact(Con_id, p-phone.no, fax, cell.no, e-mail)

patientContact-Bridge(**Patient_id*, a_id*, con_id***)

as shown in above entity the patient shows the relation with address and contact in patient contact-bridge so other table which consists of foreign key is not noted. The contact gives the data of patient contact.

For appointment entity:

Appointment (**appointment_id, patient_id*,** appointment_date, time, ward.no, ward name, floor, treatment_id,treatment, treatment type, bill_id, bill_Date, ,paymentmethod, amount)

In the appointment entity appointment_id, patient_id acts as two key attributes forming a composite key, so to remove the data redundancy, I removed unnecessary key tables which shows the nature of partial dependency.

For checking list of possible entities:

Using the 2^n-1, we know that

2^2-1=4-1=3

So there will be 3 possible entities taken from appointment entity. Now checking the partial dependency.

patient_id, appointment_id→ *

patient_id →

appointment_id→ appointment_date, time, ward no, ward name, floor, treatment_id,treatment, treatment type, bill_id, bill_Date, ,paymentmethod, amount

So patient_id alone does not show any relation of appointment of patient so all the entities with similar scenarios are eliminated. Therefore only two entities are left which are as follows:

Appointment (**appointment_id**, appointment_date, time, ward.no, ward name, floor, treatment_id,treatment, treatment type, bill_id, bill_Date, ,paymentmethod, amount)

Appointment-bridge (**patient_id*, appointment_id***)

as shown in above entity the patient shows the relation with appointment_id in Appointment-bridge so other table which consists of foreign key is not noted. The appointment entity gives the data of appointment.

For staff entity:

Staff (**staff_id, appointment_id*,patient_id*,** s-name, s-DOB, category, certified, staff_commission, speciality, s-age)

In the staff entity, staff_id, appointment_id, patient_id acts as key attributes forming a composite key, so to remove the data redundancy; I removed unnecessary key tables which shows the nature of partial dependency.

For checking list of possible entities:

Using the 2^n-1, we know that

2^3-1=8-1=7

So there will be 7 possible entities taken from staff entity. Now checking the partial dependency.

Patient_id→*

Appointment_id→*

Staff_id→ s-name, s-DOB, category, certified, staff_commission, speciality

Patient_id, staff_id→*

Appointment_id, staff_id→*

Patient_id, appointment_id→*

Patient_id, appointment_id, staff_id→*

So patient_id, appointment_id, patient_id with appointment_id does not show any relation of staff of patient so all the entities with similar scenarios are eliminated.The patient_id, appointment_id and staff_id gives relation of all entities so it used as bridge.

Therefore only two entities are left which are as follows:

Staff (**staff_id**, s-name, s-DOB, category, certified, staff_commission, speciality, s-age)

Staff-bridge (**Patient_id*, appointment_id*, staff_id***)

as shown in above entity the staff-bridge shows the relation with appointment_id, patient_id and staff_id so other table which consists of foreign key is not noted. The staff entity gives the data of staff.

For staff address entity:

s-address (**s_a_id, appointment_id*,patient_id*, staff_id*,** s-country, s-province, s-city, s-street, s-street.no)

In the staff address entity, s_a_id, appointment_id, patient_id and staff_id acts as key attributes forming a composite key, so to remove the data redundancy; I removed unnecessary key tables which shows the nature of partial dependency.

For checking list of possible entities:

Using the 2^n-1, we know that

2^4-1=16-1=15

So there will be 15 possible entities taken from staff address entity. Now checking the partial dependency.

patient_id→*

appointment_id→*

staff_id→*

s_a_id→ s-country, s-province, s-city, s-street, s-street.no

patient_id, appointment_id, staff_id, s_a_id →*

patient_id, appointment_id, staff_id →*

patient_id, appointment_id, s_a_id →*

patient_id, staff_id, s_a_id →*

appointment_id, staff_id, s_a_id →*

patient_id, appointment_id →*

patient_id, staff_id →*

patient_id, s_a_id →*

appointment_id, staff_id → *

appointment_id, s_a_id →*

staff_id, s_a_id →*

So patient_id, appointment_id does not show any relation of staff address so all the entities with similar scenarios are eliminated.The patient_id, appointment_id , staff_id and s_a_id gives relation of all entities so it used as bridge.

Therefore only three entities are left which are as follows:

s-address (**s_a_id,** s-country, s-province, s-city, s-street, s-street.no)

staffpatient-bridge (**patient_id*, appointment_id*, staff_id*, s_a_id***)

staffaddress-bridge (**staff_id*, s_a_id*)**

as shown in above entity the staffpatient-bridge and staffaddress-bridge shows relation with appointment_id, patient_id, staff_id and s_a_id so other table which consists of foreign key is not noted. S-address entity gives the data of staff address.

For staff contact entity:

s-contact (**s_con_id, appointment_id*, s_a_id*, patient_id*, staff_id*,** s-phone.no, s-email, s-fax, s-cell.no)

In the staff contact entity, s_con_id, appointment_id, s_a_id, patient_id and staff_id acts as key attributes forming a composite key, so to remove the data redundancy; I removed unnecessary key tables which shows the nature of partial dependency.

For checking list of possible entities:

Using the 2^n-1, we know that

2^5-1=32-1=31

So there will be 31 possible entities taken from staff contact entity. Now checking the partial dependency.

patient_id→*

appointment_id→*

staff_id→*

s_a_id→*

s_con_id→ s-phone.no, s-email, s-fax, s-cell.no

patient_id, appointment_id →*

patient_id, staff_id →*

patient_id, s_a_id →*

patient_id, s_con_id →*

appointment_id, staff_id →*

appointment_id, s_a_id →*

appointment_id, s_con_id → *

staff_id, s_a_id →*

staff_id, s_con_id →*

s_a_id, s_con_id →*

patient_id, appointment_id, staff_id → *

patient_id, appointment_id, s_a_id →*

patient_id, appointment_id, s_con_id → *

patient_id, staff_id, s_a_id →*

patient_id, staff_id, s_con_id →*

patient_id, s_a_id, s_con_id →*

appointment_id, staff_id, s_a_id →*

appointment_id, s_a_id, s_con_id →*

appointment_id, staff_id, s_con_id →*

staff_id, s_a_id, s_con_id →*

patient_id, appointment_id, staff_id, s_a_id, s_con_id → *

patient_id, appointment_id, staff_id, s_a_id →*

patient_id, appointment_id, staff_id, s_con_id →*

patient_id, appointment_id, s_a_id, s_con_id →*

patient_id, appointment_id, s_a_id, s_con_id →*

appointment_id, staff_id, s_a_id, s_con_id →*

So patient_id, appointment_id does not show any relation of staff contact so all the entities with similar scenarios are eliminated.The patient_id, appointment_id , staff_id , s_a_id and s_con_id gives relation of all entities so it used as bridge.

Therefore only three entities are left which are as follows:

s-contact (**s_con_id,** s-phone.no, s-email, s-fax, s-cell.no)

staffcontact-bridge (**staff_id*, s_a_id*, s_con_id***)

patientS-contact-bridge (**patient_id*, appointment_id*, staff_id*, s_a_id*, s_con_id***)

In the above entity staff contact, staff contact bridge and patient staff contact gives staff contact so other tables are discarded with their foreign keys and only 3 tables are noted. As shown in above entity the staffcontact-bridge and patientS-contact-bridge shows relation with appointment_id, patient_id, staff_id and s_a_id so other table which consists of foreign key is not noted. S-address entity gives the data of staff address

now for the final form of 2nf are as follows:

final form

Patient (**patient_id**, staff_id , p-name, age, DOB, isStaff, status, gender)

address→(**a_id**, country, province, city, street, street.no)

address-bridge→**(a_id*, patient_id***)

contact→(**Con_id**, p-phone.no, fax, cell.no, e-mail)

patient contact-bridge→ (**Patient_id*, a_id*, con_id***)

Appointment (**appointment_id**, appointment_date, time, ward.no, ward name, floor, treatment_id,treatment, treatment type, bill_id, bill_Date, ,paymentmethod, amount)

Appointment-bridge (**patient_id*, appointment_id***)

Staff (**staff_id**, s-name, s-DOB, category, certified, staff_commission, speciality, s-age)

Staff-bridge (Patient_**id*, appointment_id*, staff_id***)

s-address (**s_a_id,** s-country, s-province, s-city, s-street, s-street.no)

staffpatient-bridge (**patient_id*, appointment_id*, staff_id*, s_a_id***)

staffaddress-bridge (**staff_id*, s_a_id*)**

s-contact (**s_con_id,** s-phone.no, s-email, s-fax, s-cell.no)

staffcontact-bridge (**staff_id*, s_a_id*, s_con_id***)

patientS-contact-bridge (**patient_id*, appointment_id*, staff_id*, s_a_id*, s_con_id***)

Third normal form (3nf)**:**

Third normal form of normalization is a normal form that is used to reduce the duplication of data ensuring that entity is in second normal form and all the non-key attributes depends only on one candidate keys.

3nf is used for solving following problems:

- To eliminate data anomalies
- Make the data model more informative
- make the data model neutral to different kinds of query statistics.

Lists of bridges used:

address-bridge **(a_id*, patient_id***)

patient contact-bridge (**Patient_id*, a_id*, con_id***)

Appointment-bridge (**patient_id*, appointment_id***)

Staff-bridge (Patient_**id*, appointment_id*, staff_id***)

staffpatient-bridge (**patient_id*, appointment_id*, staff_id*, s_a_id***)

staffaddress-bridge (**staff_id*, s_a_id*)**

staffcontact-bridge (**staff_id*, s_a_id*, s_con_id***)

patientS-contact-bridge (**patient_id*, appointment_id*, staff_id*, s_a_id*, s_con_id***)

The tables given above are already in 3nf because the table which has only single non-key attribute is already in 3nf. So, the final tables for the above tables are as it is.

Now checking the transitive dependencies for other tables:

Listing the tables that are required to check for transitive dependencies.

Patient (**patient_id**, staff_id, p-name, age, DOB, isStaff, status, gender)

address (**a_id**, country, province, city, street, street.no)

contact (**Con_id**, p-phone.no, fax, cell.no, e-mail)

Appointment (**appointment_id**, appointment_date, time, ward.no, ward name, floor, treatment_id,treatment, treatment type, bill_id, bill_Date, ,paymentmethod, amount)

Staff (**staff_id**, s-name, s-DOB, category, certified, staff_commission, speciality, s-age)

s-address (**s_a_id,** s-country, s-province, s-city, s-street, s-street.no)

s-contact (**s_con_id,** s-phone.no, s-email, s-fax, s-cell.no)

for the patient table:

Patient (**patient_id**, staff_id, p-name, age, DOB, isStaff, status, gender)

p-name→*

age→*

DOB→*

isStaff→*

status→*

gender→*

The patient table is already in 3nf because there is no transitive dependency. So, the patient table is

Patient (**patient_id**, staff_id, p-name, age, DOB, isStaff, status, gender)

For the patient address table:

Address (**a_id**, country, province, city, street, street.no)

Country→*

province→*

city →*

street→*

street.no→*

The patient address table is already in 3nf because there is no transitive dependency. So, the patient address table is

Address (**a_id**, country, province, city, street, street.no)

For patient contact

contact (**Con_id**, p-phone.no, fax, cell.no, e-mail)

p-phone.no →*

fax →*

e-mail →*

cell.no →*

The patient contact table is already in 3nf because there is no transitive dependency. So, the patient address table is

contact (**Con_id**, p-phone.no, fax, cell.no, e-mail)

for appointment table

Appointment (**appointment_id**, appointment_date, time, ward.no, ward name, floor, treatment_id,treatment, treatment type, bill_id, bill_Date, ,paymentmethod, amount)

appointment_date→*

time→*

ward no→ ward name, floor

treatment_id→ treatment, treatment type

bill_id→ bill_Date, paymentmethod, amount

the appointment table is in transitive dependency, so the final table constructed are as follows:

Appointment (**appointment_id**, appointment_date, time, **ward no*, treatment_id*, bill_id*)**

Ward (**ward.no**, ward name, floor)

Treatments (**treatment_id**, treatment, treatment type)

Bill (**bill_id**, bill_Date, paymentmethod, amount)

For staff table

Staff (**staff_id**, s-name, s-DOB, category, certified, staff_commission, speciality, s-age)

s-name→*

s-DOB→*

category→*

certified→*

staff_commission→*

speciality→*

s-age→*

the staff table is already in 3nf because there is no transitive dependency. So, the staff table is

Staff (**staff_id**, s-name, s-DOB, category, certified, staff_commission, speciality, s-age)

For staff address

s-address (**s_a_id,** s-country, s-province, s-city, s-street, s-street.no)

s-country→*

s-province→*

s-city→*

s-street→*

s-street.no→*

the staff address table is already in 3nf because there is no transitive dependency. So, the staff address table is

s-address (**s_a_id,** s-country, s-province, s-city, s-street, s-street.no)

For staff contact

s-contact (**s_con_id,** s-phone.no, s-email, s-fax, s-cell.no)

s-phone.no→*

s-email→*

s-fax→*

s-cell.no→*

the staff address table is already in 3nf because there is no transitive dependency. So, the staff address table is

s-contact (**s_con_id,** s-phone.no, s-email, s-fax, s-cell.no)

now for the final form of 3nf, the tables are constructed:

Patient (**patient_id**, staff_id, p-name, age, DOB, isStaff, status, gender)

address (**a_id**, country, province, city, street, street.no)

contact (**Con_id**, p-phone.no, fax, cell.no, e-mail)

patient contact-bridge (**Patient_id*, a_id*, con_id***)

Appointment (**appointment_id**, appointment_date, time, **ward no*, treatment_id*, bill_id*)**

Ward (**ward.no**, ward name, floor)

Treatments (**treatment_id**, treatment, treatment type)

Bill (**bill_id**, bill_Date, paymentmethod, amount)

Staff (**staff_id**, s-name, s-DOB, category, certified, staff_commission, speciality, s-age)

Staff-bridge (**Patient_id*, appointment_id*, staff_id***)

S_address (**s_a_id,** s_country, s_province, s-city, s_street, s_street_no)

S_contact (**s_con_id,** s_phone_no, s_email, s_fax, s_cell_no)

staffcontact-bridge (**staff_id*, s_a_id*, s_con_id***)

for minimizing the data redundancy, so I have removed address-bridge→**(a_id*, patient_id***), Appointment-bridge (**patient_id*, appointment_id***), staffaddress-bridge (**staff_id*, s_a_id*)**, patientS-contact-bridge (**patient_id*, appointment_id*, staff_id*, s_a_id*, s_con_id***).

Final E-R diagram

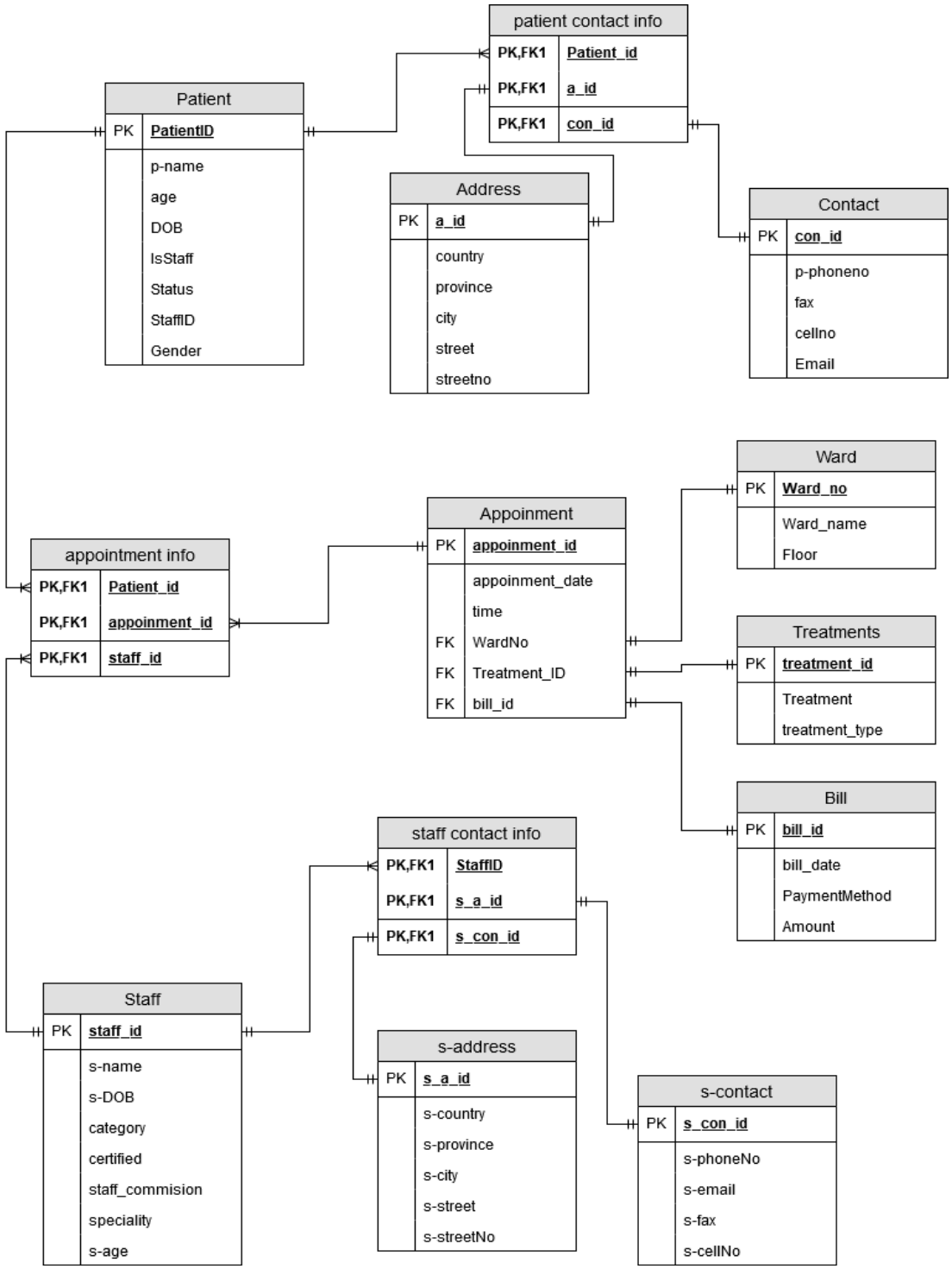

Figure 2 ER diagram

In the given figure 2, the final ER diagram is made after normalizing the initial ER diagram. In the figure 2, we can see that there are many entities which are connected to each other while giving the relationship between them. In the patient table we can see that patient can have multiple address and contact. The patient can

also have as many appointments as he/she wants. For staff, many staff can attend in one appointment in one specified ward for one treatment. The staff can attend many appointments but in different time as time is mentioned for specified appointment. Staff can also have many addresses and contact as same attributes are used when staff becomes patient.

In the given figure the patient is connected to address and contact through contact Info Bridge which gives relation between patient, address and contact. The patient is connected to appointment Info Bridge which gives relationship with staff and appointments which gives the detail to patient about staff, ward and payments before the treatment. The appointment table is connected to three tables which are obtained through 3nf. In the appointment table, there are 3 foreign keys which give the information of ward, treatment and bill payment. The staff table is similar to patient table which is connected to staff contact Info Bridge which gives a relationship with staff address and staff contact. Then staff contact Info Bridge is connected to staff address and staff contact respectively.

Data dictionary

Entity	Attributes	Data type	Description
patient	**Patient_id**	VARCHAR2(6)	It stores unique id of patient which cannot be empty
	P_NAME	VARCHAR2(25)	It stores name of patient which cannot be null
	DOB	DATE	It stores date of patient which cannot be empty
	ISSTAFF	VARCHAR2(3)	It tells if the patient is normal or doctor
	GENDER	VARCHAR2(6)	It tells the gender of the patient
	AGE	NUMBER	It tells the age of patient
	STAFF_ID	VARCHAR2(6)	It stores the id no of staff who is admitted
	type	VARCHAR2(8)	It tells if patient is regular or new

Table 4 patient table

Entity	Attributes	Data type	Description
address	**A_ID**	VARCHAR2(6)	It gives unique id for address
	COUNTRY	VARCHAR2(25)	It gives name of country
	PROVINCE	VARCHAR2(25)	It gives province number
	CITY	VARCHAR2(25)	It gives name of city
	STREET_NO	NUMBER	It gives street number
	STREET	VARCHAR2(25)	It gives name of street

Table 5 patient address

Entity	Attributes	Data type	Description
Contact	**CON_ID**	VARCHAR2(6)	It gives unique id for contact of patient
	P_PHONE_NO	NUMBER	It gives phone no of patient
	CELL_NO	NUMBER	It gives another cell number of

			patient
	EMAIL	VARCHAR2(25)	It gives unique email address
	FAX	NUMBER	It gives unique fax number

Table 6 patient contact

Entity	Attributes	Data type	Description
ward	**WARD_NO**	VARCHAR2(6)	It gives unique ward number
	WARD_NAME	VARCHAR2(10)	It gives ward name
	FLOOR	NUMBER	It give which floor ward is in.

Table 7 ward

Entity	Attributes	Data type	Description
treatment	**TREATMENT_ID**	VARCHAR2(6)	It gives unique treatment id
	TREATMENT	VARCHAR2(25)	It gives name of treatment
	TREATMENT_TYPE	VARCHAR2(25)	It gives which type of treatment it is.

Table 8 Treatment

Entity	Attributes	Data type	Description
Bill	**BILL_ID**	VARCHAR2(6)	It gives unique id to bill
	Bill_date	date	It gives bill date
	amount	number	It gives the amount paid by the patient
	Payment method	VARCHAR2(10)	It gives the payment method

Table 9 Bill

Entity	Attributes	Data type	Description
appointment	**APPOINTMENT_ID**	VARCHAR2(6)	It gives the unique appointment id
	APPOINTMENT_DATE	DATE	It gives date of appointment
	TIME	VARCHAR2(8)	It gives time of appointment
	WARD_NO*	VARCHAR2(6)	It gives information of ward
	TREATMENT_ID*	VARCHAR2(6)	It gives information of treatment
	BILL_ID*	VARCHAR2(6)	It gives information of bill

Table 10 Appointment

Entity	Attributes	Data type	Description
Appointment info	**PATIENT_ID***	VARCHAR2(6)	It gives relation with staff and appointment
	STAFF_ID*	VARCHAR2(6)	It gives relation with patient and appointment
	APPOINTMENT_ID*	VARCHAR2(6)	It gives relation with staff and patient

Table 11 Appointment info

Entity	Attributes	Data type	Description
contact info	**PATIENT_ID***	VARCHAR2(6)	It gives relation with address and contact
	A_ID*	VARCHAR2(6)	It gives relation with patient and contact
	CON_ID*	VARCHAR2(6)	It gives relation with patient and address

Table 12 contact info

Entity	Attributes	Data type	Description
staff	**STAFF_ID**	VARCHAR2(6)	It gives unique id for staff
	S_NAME	VARCHAR2(25)	it gives name of staff
	S_DOB	DATE	It gives date of birth of staff
	CATEGORY	VARCHAR2(8)	it gives the category of staff
	STAFF_COMMISSION	NUMBER	It gives commission of staff from the appointment
	SPECIALITY	VARCHAR2(8)	It gives the specialty of staff
	CERTIFIED	VARCHAR2(8)	It gives the certification of staff

Table 13 staff

Entity	Attributes	Data type	Description
S_address	**S_A_ID**	VARCHAR2(6)	It gives unique id for address
	S_COUNTRY	VARCHAR2(25)	It gives name of country
	S_PROVINCE	VARCHAR2(25)	It gives province number
	S_CITY	VARCHAR2(25)	It gives name of city
	S_STREET_NO	NUMBER	It gives street number
	S_STREET	VARCHAR2(25)	It gives name of street

Table 14 staff address

Entity	Attributes	Data type	Description
S_Contact	**S_CON_ID**	VARCHAR2(6)	It gives unique id for contact of staff
	s_PHONE_NO	NUMBER	It gives phone no of patient
	S_CELL_NO	NUMBER	It gives another cell number of staff
	S_EMAIL	VARCHAR2(25)	It gives unique email address
	S_FAX	NUMBER	It gives unique fax number

Table 15 staff contact

Entity	Attributes	Data type	Description
Staff contact info	**STAFF_ID***	VARCHAR2(6)	It gives relation with address and contact
	S_A_ID*	VARCHAR2(6)	It gives relation with staff and contact
	S_CON_ID*	VARCHAR2(6)	It gives relation with staff and address

Table 16 staff contact info

Creating table

At first sql was connected to system. Then a user was created with name vayodha and permission was granted to connect to vayodha.

```
SQL> connect system
Enter password:
Connected.
SQL> CREATE USER vayodha IDENTIFIED BY vayodha;

User created.

SQL> grant connect, resource to vayodha;

Grant succeeded.

SQL> connect vayodha;
Enter password:
Connected.
SQL>
```

Figure 3 user creating

CREATE USER VAYODHA IDENTIFIED BY VAYODHA;

GRANT CONNECT, RESOURCE TO VAYODHA;

CONNECT VAYODHA;

VAYODHA

Table was created for each table with specified details for suitable data type.

```
SQL> create table Patient(Patient_id VARCHAR2(6),
  2  Constraint Patient_PK Primary Key(Patient_id),
  3  P_Name VARCHAR2(25) NOT NULL,
  4  DOB DATE,
  5  isStaff VARCHAR2(3),
  6  Gender VARCHAR2(6),
  7  Age NUMBER,
  8  Staff_id VARCHAR2(6),
  9  type VARCHAR2(8)
 10  );

Table created.
```

Figure 4 patient table

create table Patient(Patient_id VARCHAR2(6),

Constraint Patient_PK Primary Key(Patient_id),

P_Name VARCHAR2(25) NOT NULL,

DOB DATE,

isStaff VARCHAR2(3),

Gender VARCHAR2(6),

Age NUMBER,

Staff_id VARCHAR2(6),

type VARCHAR2(8));

```
SQL> describe patient;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 PATIENT_ID                                NOT NULL VARCHAR2(6)
 P_NAME                                    NOT NULL VARCHAR2(25)
 DOB                                                DATE
 ISSTAFF                                            VARCHAR2(3)
 GENDER                                             VARCHAR2(6)
 AGE                                                NUMBER
 STAFF_ID                                           VARCHAR2(6)
 TYPE                                               VARCHAR2(8)
```

Figure 5 describe patient table

Describe patient;

```
SQL> CREATE TABLE Address(A_id VARCHAR2(6),
  2  Constraint Address_PK Primary Key (A_id),
  3  Country VARCHAR2(25) NOT NULL,
  4  Province VARCHAR2(25) NOT NULL,
  5  City VARCHAR2(25) NOT NULL,
  6  Street_No Number NOT NULL,
  7  Street VARCHAR2(25) NOT NULL);

Table created.

SQL> describe address
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 A_ID                                      NOT NULL VARCHAR2(6)
 COUNTRY                                   NOT NULL VARCHAR2(25)
 PROVINCE                                  NOT NULL VARCHAR2(25)
 CITY                                      NOT NULL VARCHAR2(25)
 STREET_NO                                 NOT NULL NUMBER
 STREET                                    NOT NULL VARCHAR2(25)

SQL>
```

Figure 6 patient address

CREATE TABLE Address(A_id VARCHAR2(6),

Constraint Address_PK Primary Key (A_id),

Country VARCHAR2(25) NOT NULL,

Province VARCHAR2(25) NOT NULL,

City VARCHAR2(25) NOT NULL,

Street_No Number NOT NULL,

Street VARCHAR2(25) NOT NULL);

describe address

```
SQL> CREATE TABLE Contact(Con_ID VARCHAR2(6),
  2  Constraint Patient_Contact_ID_PK Primary Key (Con_ID),
  3  P_Phone_no NUMBER NOT NULL,
  4  Cell_no NUMBER NOT NULL,
  5  Email VARCHAR2(25) NOT NULL,
  6  Fax NUMBER NOT NULL);

Table created.

SQL> describe contact
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 CON_ID                                    NOT NULL VARCHAR2(6)
 P_PHONE_NO                                NOT NULL NUMBER
 CELL_NO                                   NOT NULL NUMBER
 EMAIL                                     NOT NULL VARCHAR2(25)
 FAX                                       NOT NULL NUMBER
```

Figure 7 patient contact

CREATE TABLE Contact(Con_ID VARCHAR2(6),

Constraint Patient_Contact_ID_PK Primary Key (Con_ID),

P_Phone_no NUMBER NOT NULL,

Cell_no NUMBER NOT NULL,

Email VARCHAR2(25) NOT NULL,

Fax NUMBER NOT NULL);

Describe contact

```
SQL> CREATE TABLE Contact_info(Patient_id VARCHAR2(6),
  2  CONSTRAINT Patient_FK FOREIGN KEY (Patient_id)
  3  REFERENCES Patient(Patient_id),
  4  A_id VARCHAR2(6),
  5  CONSTRAINT A_id_FK1 FOREIGN KEY(A_id)
  6  REFERENCES Address(A_id),
  7  Con_ID  VARCHAR2(6),
  8  CONSTRAINT Con_ID_FK1 FOREIGN KEY(Con_ID)
  9  REFERENCES Contact(Con_ID));

Table created.

SQL> describe contact_info
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 PATIENT_ID                                         VARCHAR2(6)
 A_ID                                               VARCHAR2(6)
 CON_ID                                             VARCHAR2(6)

SQL>
```

Figure 8 patient contact information

CREATE TABLE Contact_info(Patient_id VARCHAR2(6),

CONSTRAINT Patient_FK FOREIGN KEY (Patient_id)

REFERENCES Patient(Patient_id),

A_id VARCHAR2(6),

CONSTRAINT A_id_FK1 FOREIGN KEY(A_id)

REFERENCES Address(A_id),

Con_ID VARCHAR2(6),

CONSTRAINT Con_ID_FK1 FOREIGN KEY(Con_ID)

REFERENCES Contact(Con_ID));

Describe contact_info

```
SQL> CREATE TABLE Ward (Ward_no VARCHAR2(6),
  2  CONSTRAINT Ward_No_PK PRIMARY KEY (Ward_no),
  3  Ward_name VARCHAR2(10) NOT NULL,
  4  Floor Number  NOT NULL);

Table created.

SQL> describe ward
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 WARD_NO                                   NOT NULL VARCHAR2(6)
 WARD_NAME                                 NOT NULL VARCHAR2(10)
 FLOOR                                     NOT NULL NUMBER
```

Figure 9 ward

CREATE TABLE Ward (Ward_no VARCHAR2(6),

CONSTRAINT Ward_No_PK PRIMARY KEY (Ward_no),

Ward_name VARCHAR2(10) NOT NULL,

Floor Number NOT NULL);

Describe ward

```
SQL> CREATE TABLE Treatment (Treatment_id VARCHAR2(6),
  2  CONSTRAINT Treatment_PK PRIMARY KEY (Treatment_id),
  3  Treatment VARCHAR2(25) NOT NULL,
  4  treatment_type VARCHAR2(25) NOT NULL);

Table created.

SQL> describe treatment
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 TREATMENT_ID                              NOT NULL VARCHAR2(6)
 TREATMENT                                 NOT NULL VARCHAR2(25)
 TREATMENT_TYPE                            NOT NULL VARCHAR2(25)
```

Figure 10 Treatment

CREATE TABLE Treatment (Treatment_id VARCHAR2(6),

CONSTRAINT Treatment_PK PRIMARY KEY (Treatment_id),

Treatment VARCHAR2(25) NOT NULL,

treatment_type VARCHAR2(25) NOT NULL);

describe treatment

```
SQL> CREATE TABLE Bill (Bill_id VARCHAR2(6),
  2  CONSTRAINT Bill_No_PK PRIMARY KEY (Bill_id),
  3  Bill_date DATE NOT NULL,
  4  Amount NUMBER NOT NULL,
  5  Payment VARCHAR2(10));

Table created.

SQL> DESCRIBE Bill;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 BILL_ID                                   NOT NULL VARCHAR2(6)
 BILL_DATE                                 NOT NULL DATE
 AMOUNT                                    NOT NULL NUMBER
 PAYMENT                                            VARCHAR2(10)
```

Figure 11 Bill

CREATE TABLE Bill (Bill_id VARCHAR2(6),

CONSTRAINT Bill_No_PK PRIMARY KEY (Bill_id),

Bill_date DATE NOT NULL,

Amount NUMBER NOT NULL,

Payment VARCHAR2(10));

DESCRIBE Bill;

```
SQL> CREATE TABLE Appointment(Appointment_id  VARCHAR2(6),
  2  CONSTRAINT Appointment_id_PK PRIMARY KEY (Appointment_id),
  3  Appointment_date DATE NOT NULL,
  4  Time VARCHAR(8) NOT NULL,
  5  Ward_no VARCHAR2(6),
  6  CONSTRAINT Ward_no_FK2 FOREIGN KEY(Ward_no)
  7  REFERENCES Ward(Ward_no),
  8  Treatment_id VARCHAR2(6),
  9  CONSTRAINT Treatment_id_FK2 FOREIGN KEY(Treatment_id)
 10  REFERENCES Treatment(Treatment_id),
 11  Bill_id VARCHAR2(6),
 12  CONSTRAINT Bill_FK2 FOREIGN KEY(Bill_id)
 13  REFERENCES Bill(Bill_id));

Table created.

SQL> describe appointment
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 APPOINTMENT_ID                            NOT NULL VARCHAR2(6)
 APPOINTMENT_DATE                          NOT NULL DATE
 TIME                                      NOT NULL VARCHAR2(8)
 WARD_NO                                            VARCHAR2(6)
 TREATMENT_ID                                       VARCHAR2(6)
 BILL_ID                                            VARCHAR2(6)
```

Figure 12 Appointment

CREATE TABLE Appointment(Appointment_id VARCHAR2(6),

CONSTRAINT Appointment_id_PK PRIMARY KEY (Appointment_id),

Appointment_date DATE NOT NULL,

Time VARCHAR(8) NOT NULL,

Ward_no VARCHAR2(6),

CONSTRAINT Ward_no_FK2 FOREIGN KEY(Ward_no)

REFERENCES Ward(Ward_no),

Treatment_id VARCHAR2(6),

CONSTRAINT Treatment_id_FK2 FOREIGN KEY(Treatment_id)

REFERENCES Treatment(Treatment_id),

Bill_id VARCHAR2(6),

CONSTRAINT Bill_FK2 FOREIGN KEY(Bill_id)

REFERENCES Bill(Bill_id));

Describe appointment

```
SQL> CREATE TABLE AppointmentInfo(Patient_id VARCHAR2(6),
  2  CONSTRAINT Patient_id_FK3 FOREIGN KEY (Patient_id)
  3  REFERENCES Patient(Patient_id),
  4  Staff_ID VARCHAR2(6),
  5  CONSTRAINT Staff_ID_FK3 FOREIGN KEY(Staff_ID)
  6  REFERENCES Staff(Staff_ID),
  7  Appointment_id VARCHAR2(6),
  8  CONSTRAINT Appointment_id_FK3 FOREIGN KEY(Appointment_id)
  9  REFERENCES Appointment(Appointment_id));

Table created.

SQL> describe appointmentinfo
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 PATIENT_ID                                         VARCHAR2(6)
 STAFF_ID                                           VARCHAR2(6)
 APPOINTMENT_ID                                     VARCHAR2(6)
```

Figure 13 Appointment information

CREATE TABLE AppointmentInfo(Patient_id VARCHAR2(6),

CONSTRAINT Patient_id_FK3 FOREIGN KEY (Patient_id)

REFERENCES Patient(Patient_id),

Staff_ID VARCHAR2(6),

CONSTRAINT Staff_ID_FK3 FOREIGN KEY(Staff_ID)

REFERENCES Staff(Staff_ID),

Appointment_id VARCHAR2(6),

CONSTRAINT Appointment_id_FK3 FOREIGN KEY(Appointment_id)

REFERENCES Appointment(Appointment_id));

Describe appointmentinfo

```
SQL> CREATE TABLE Staff(Staff_ID VARCHAR2(6),
  2  Constraint Staff_PK Primary Key (Staff_ID),
  3  s_name VARCHAR2(25) NOT NULL,
  4  s_DOB DATE,
  5  category VARCHAR2(8) NOT NULL,
  6  staff_commission number NOT NULL,
  7  Speciality VARCHAR2(8) NOT NULL,
  8  certified VARCHAR2(8) NOT NULL)
  9  ;

Table created.
```

Figure 14 Staff

CREATE TABLE Staff(Staff_ID VARCHAR2(6),

Constraint Staff_PK Primary Key (Staff_ID),

s_name VARCHAR2(25) NOT NULL,

s_DOB DATE,

category VARCHAR2(8) NOT NULL,

staff_commission number NOT NULL,

Speciality VARCHAR2(8) NOT NULL,

certified VARCHAR2(8) NOT NULL)

describe staff

```
SQL> CREATE TABLE S_Address(s_a_id VARCHAR2(6),
  2  CONSTRAINT S_A_PK PRIMARY KEY (s_a_id),
  3  S_Country VARCHAR2(25) NOT NULL,
  4  S_Province VARCHAR2(25) NOT NULL,
  5  S_City VARCHAR2(25) NOT NULL,
  6  S_Street_No VARCHAR2(10) NOT NULL,
  7  S_Street_Name VARCHAR2(25) NOT NULL);

Table created.

SQL> describe s_address
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 S_A_ID                                    NOT NULL VARCHAR2(6)
 S_COUNTRY                                 NOT NULL VARCHAR2(25)
 S_PROVINCE                                NOT NULL VARCHAR2(25)
 S_CITY                                    NOT NULL VARCHAR2(25)
 S_STREET_NO                               NOT NULL VARCHAR2(10)
 S_STREET_NAME                             NOT NULL VARCHAR2(25)
```

Figure 15 Staff address

CREATE TABLE S_Address(s_a_id VARCHAR2(6),

CONSTRAINT S_A_PK PRIMARY KEY (s_a_id),

S_Country VARCHAR2(25) NOT NULL,

S_Province VARCHAR2(25) NOT NULL,

S_City VARCHAR2(25) NOT NULL,

S_Street_No VARCHAR2(10) NOT NULL,

S_Street_Name VARCHAR2(25) NOT NULL);

Describe s_address

```
SQL> CREATE TABLE S_Contact (s_con_id VARCHAR2(6),
  2  CONSTRAINT S_Con_ID_PK PRIMARY KEY (s_con_id),
  3  s_phone_no Number (10) NOT NULL,
  4  s_cell_no Number (10) NOT NULL,
  5  s_email VARCHAR2(25) NOT NULL,
  6  s_fax NUMBER(10) NOT NULL);

Table created.

SQL> describe s_contact
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 S_CON_ID                                  NOT NULL VARCHAR2(6)
 S_PHONE_NO                                NOT NULL NUMBER(10)
 S_CELL_NO                                 NOT NULL NUMBER(10)
 S_EMAIL                                   NOT NULL VARCHAR2(25)
 S_FAX                                     NOT NULL NUMBER(10)
```

Figure 16 Staff contact

CREATE TABLE S_Contact (s_con_id VARCHAR2(6),

CONSTRAINT S_Con_ID_PK PRIMARY KEY (s_con_id),

s_phone_no Number (10) NOT NULL,

s_cell_no Number (10) NOT NULL,

s_email VARCHAR2(25) NOT NULL,

s_fax NUMBER(10) NOT NULL);

describe s_contact

```
SQL> CREATE TABLE StaffContactInfo(Staff_ID VARCHAR2(6),
  2  CONSTRAINT Staff_ID_FK2 FOREIGN KEY (Staff_ID)
  3  REFERENCES Staff(Staff_ID),
  4  s_a_id VARCHAR2(6),
  5  CONSTRAINT s_a_id_FK2 FOREIGN KEY(s_a_id)
  6  REFERENCES S_Address(s_a_id),
  7  s_con_id VARCHAR2(6),
  8  CONSTRAINT s_con_id_FK2 FOREIGN KEY(s_con_id)
  9  REFERENCES S_Contact(s_con_id));

Table created.

SQL> describe staffcontactinfo
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 STAFF_ID                                           VARCHAR2(6)
 S_A_ID                                             VARCHAR2(6)
 S_CON_ID                                           VARCHAR2(6)
```

Figure 17 Staff contact information

CREATE TABLE StaffContactInfo(Staff_ID VARCHAR2(6),

CONSTRAINT Staff_ID_FK2 FOREIGN KEY (Staff_ID)

REFERENCES Staff(Staff_ID),

s_a_id VARCHAR2(6),

CONSTRAINT s_a_id_FK2 FOREIGN KEY(s_a_id)

REFERENCES S_Address(s_a_id),

s_con_id VARCHAR2(6),

CONSTRAINT s_con_id_FK2 FOREIGN KEY(s_con_id)

REFERENCES S_Contact(s_con_id));

Describe staffcontactinfo

Inserting the data in all the tables

```
SQL> INSERT INTO Patient VALUES ('P1','John','06-JAN-00','no','Male', 19, '','new');
1 row created.
SQL> INSERT INTO Patient VALUES ('P2','Max','08-FEB-97' ,'no','Male',22, '','new');
1 row created.
SQL> INSERT INTO Patient VALUES ('P3','Tom','01-OCT-89' ,'no', 'Male', 30, '','new');
1 row created.
SQL> INSERT INTO Patient VALUES ('P4','Jim','01-SEP-01','no','Male' ,18, '','regular');
1 row created.
SQL> INSERT INTO Patient VALUES ('P5','Natalia','05-NOV-65','no','Female',54, '','new');
1 row created.
SQL> INSERT INTO Patient VALUES ('P6','Pam','06-AUG-79' ,'no','Female' ,40, '','new');
1 row created.
SQL> INSERT INTO Patient VALUES ('P7','Andy','18-JUN-92' ,'yes','Male',27, 'S11','regular');
1 row created.
```

Figure 18 Insert patient

INSERT INTO Patient VALUES ('P1','John','06-JAN-00','no','Male', 19, '','new');

INSERT INTO Patient VALUES ('P2','Max','08-FEB-97' ,'no','Male',22, '','new');

INSERT INTO Patient VALUES ('P3','Tom','01-OCT-89' ,'no', 'Male', 30, '','new');

INSERT INTO Patient VALUES ('P4','Jim','01-SEP-01','no','Male' ,18, '','regular');

INSERT INTO Patient VALUES ('P5','Natalia','05-NOV-65','no','Female',54, '','new');

INSERT INTO Patient VALUES ('P6','Pam','06-AUG-79' ,'no','Female' ,40, '','new');

INSERT INTO Patient VALUES ('P7','Andy','18-JUN-92' ,'yes','Male',27, 'S11','regular');

```
SQL> INSERT INTO  Address VALUES ('A1','Nepal','Province no.3','Kathmandu',139,'Jyatha Street');

1 row created.

SQL> INSERT INTO  Address VALUES ('A2','Nepal','Province no.3','Hetauda',4,'Hetauda City College');

1 row created.

SQL> INSERT INTO  Address VALUES ('A3','Nepal','Province no.3','Dolakha',21,'Kalinchok');

1 row created.

SQL> INSERT INTO  Address VALUES ('A4','Nepal','Province no.2','Birgunj',45,'Shankaracharya Gate');

1 row created.

SQL> INSERT INTO  Address VALUES ('A5','Nepal','Province no.1','Damak',65,'Damak Hospital');

1 row created.

SQL> INSERT INTO  Address VALUES ('A6','Nepal','Province no.5','Butwal',39,'Siddha Baba Temple');

1 row created.

SQL> INSERT INTO  Address VALUES ('A7','Nepal','Gandaki Pradesh','Pokhara',150,'Phewa Tal');

1 row created.
```

Figure 19 Inserting Address

INSERT INTO Address VALUES ('A1','Nepal','Province no.3','Kathmandu',139,'Jyatha Street');

INSERT INTO Address VALUES ('A2','Nepal','Province no.3','Hetauda',4,'Hetauda City College');

INSERT INTO Address VALUES ('A3','Nepal','Province no.3','Dolakha',21,'Kalinchok');

INSERT INTO Address VALUES ('A4','Nepal','Province no.2','Birgunj',45,'Shankaracharya Gate');

INSERT INTO Address VALUES ('A5','Nepal','Province no.1','Damak',65,'Damak Hospital');

INSERT INTO Address VALUES ('A6','Nepal','Province no.5','Butwal',39,'Siddha Baba Temple');

INSERT INTO Address VALUES ('A7','Nepal','Gandaki Pradesh','Pokhara',150,'Phewa Tal');

```
SQL> INSERT INTO Contact VALUES ('C1',9813778955,981378755,'john@gmail.com',22228888);

1 row created.

SQL> INSERT INTO Contact VALUES ('C2',9814798755,984178757,'max@gmail.com',12345868);

1 row created.

SQL> INSERT INTO Contact VALUES ('C3',9841768955,981778455,'tom@gmail.com',45225868);

1 row created.

SQL> INSERT INTO Contact VALUES ('C4',9840978955,982378755,'Jim@gmail.com',78225868);

1 row created.

SQL> INSERT INTO Contact VALUES ('C5',9815678555,981568755,'Natalia@gmail.com',22228888);

1 row created.

SQL> INSERT INTO Contact VALUES ('C6',9818756955,981678555,'Pam@gmail.com',56228888);

1 row created.

SQL> INSERT INTO Contact VALUES ('C7',9819778855,986378755,'Andy@gmail.com',67228888);

1 row created.
```

Figure 20 Insert Contact

INSERT INTO Contact VALUES ('C1',9813778955,981378755,'john@gmail.com',22228888);

INSERT INTO Contact VALUES ('C2',9814798755,984178757,'max@gmail.com',12345868);

INSERT INTO Contact VALUES ('C3',9841768955,981778455,'tom@gmail.com',45225868);

INSERT INTO Contact VALUES ('C4',9840978955,982378755,'Jim@gmail.com',78225868);

INSERT INTO Contact VALUES ('C5',9815678555,981568755,'Natalia@gmail.com',22228888);

INSERT INTO Contact VALUES ('C6',9818756955,981678555,'Pam@gmail.com',56228888);

INSERT INTO Contact VALUES
('C7',9819778855,986378755,'Andy@gmail.com',67228888);

```
SQL> INSERT INTO Contact_info VALUES('P1','A1','C1');

1 row created.

SQL> INSERT INTO Contact_info VALUES('P2','A2','C2');

1 row created.

SQL> INSERT INTO Contact_info VALUES('P3','A3','C3');

1 row created.

SQL> INSERT INTO Contact_info VALUES('P4','A4','C4');

1 row created.

SQL> INSERT INTO Contact_info VALUES('P5','A5','C5');

1 row created.

SQL> INSERT INTO Contact_info VALUES('P6','A6','C6');

1 row created.

SQL> INSERT INTO Contact_info VALUES('P7','A7','C7');

1 row created.
```

Figure 21 Insert Contact_Info

INSERT INTO Contact_info VALUES('P1','A1','C1');

INSERT INTO Contact_info VALUES('P2','A2','C2');

INSERT INTO Contact_info VALUES('P3','A3','C3');

INSERT INTO Contact_info VALUES('P4','A4','C4');

INSERT INTO Contact_info VALUES('P5','A5','C5');

INSERT INTO Contact_info VALUES('P6','A6','C6');

INSERT INTO Contact_info VALUES('P7','A7','C7');

```
SQL> INSERT INTO Ward VALUES('W1','Pediatrics',1);

1 row created.

SQL> INSERT INTO Ward VALUES('W2','Geriatrics',2);

1 row created.

SQL> INSERT INTO Ward VALUES('W3','Maternity',5);

1 row created.

SQL> INSERT INTO Ward VALUES('W4','Endoscopy',4);

1 row created.

SQL> INSERT INTO Ward VALUES('W5','Therapy',2);

1 row created.

SQL> INSERT INTO Ward VALUES('W6','Cardiology',2);

1 row created.

SQL> INSERT INTO Ward VALUES('W7','ENT',1);

1 row created.
```

Figure 22 Insert Ward

```
INSERT INTO Ward VALUES('W1','Pediatrics',1);
INSERT INTO Ward VALUES('W2','Geriatrics',2);
INSERT INTO Ward VALUES('W3','Maternity',5);
INSERT INTO Ward VALUES('W4','Endoscopy',4);
INSERT INTO Ward VALUES('W5','Therapy',2);
INSERT INTO Ward VALUES('W6','Cardiology',2);
INSERT INTO Ward VALUES('W7','ENT',1);
```

```
SQL> INSERT INTO treatment VALUES('T1','Acute sinusitis','Vaccine therapy');

1 row created.

SQL> INSERT INTO treatment VALUES('T2','Weight Loss','health regime');

1 row created.

SQL> INSERT INTO treatment VALUES('T3','child healthcare','consultant');

1 row created.

SQL> INSERT INTO treatment VALUES('T4','Gastrointestinal tract','stomach');

1 row created.

SQL> INSERT INTO treatment VALUES('T5','exercise routine','heart disease');

1 row created.

SQL> INSERT INTO treatment VALUES('T6','heart','artery block');

1 row created.

SQL> INSERT INTO treatment VALUES('T7','ear','ear wax');

1 row created.
```

Figure 23 Insert Treatment

```
INSERT INTO treatment VALUES('T1','Acute sinusitis','Vaccine therapy');
INSERT INTO treatment VALUES('T2','Weight Loss','health regime');
INSERT INTO treatment VALUES('T3','child healthcare','consultant');
INSERT INTO treatment VALUES('T4','Gastrointestinal tract','stomach');
INSERT INTO treatment VALUES('T5','exercise routine','heart disease');
INSERT INTO treatment VALUES('T6','heart','artery block');
INSERT INTO treatment VALUES('T7','ear','ear wax');
```

```
SQL> INSERT INTO bill VALUES('B1','16-JAN-19',2000,'Cash');

1 row created.

SQL> INSERT INTO bill VALUES('B2','9-SEP-19',1000,'Cash');

1 row created.

SQL> INSERT INTO bill VALUES('B3','14-OCT-19',1000,'Cash');

1 row created.

SQL> INSERT INTO bill VALUES('B4','8-SEP-19',2000,'Cash');

1 row created.

SQL> INSERT INTO bill VALUES('B5','16-OCT-19',2000,'Cash');

1 row created.

SQL> INSERT INTO bill VALUES('B6','17-FEB-19',2000,'Cash');

1 row created.

SQL> INSERT INTO bill VALUES('B7','11-MAR-19',2000,'E-Cash');

1 row created.
```

Figure 24 Insert Bill

INSERT INTO bill VALUES('B1','16-JAN-19',2000,'Cash');

INSERT INTO bill VALUES('B2','9-SEP-19',1000,'Cash');

INSERT INTO bill VALUES('B3','14-OCT-19',1000,'Cash');

INSERT INTO bill VALUES('B4','8-SEP-19',2000,'Cash');

INSERT INTO bill VALUES('B5','16-OCT-19',2000,'Cash');

INSERT INTO bill VALUES('B6','17-FEB-19',2000,'Cash');

INSERT INTO bill VALUES('B7','11-MAR-19',2000,'E-Cash');

```
SQL> INSERT INTO appointment VALUES('A1','15-JAN-19','10:30 AM','W1','T1','B1');

1 row created.

SQL> INSERT INTO appointment VALUES('A2','8-SEP-19','11:30 AM','W2','T2','B2');

1 row created.

SQL> INSERT INTO appointment VALUES('A3','13-OCT-19','12:30 AM','W3','T3','B3');

1 row created.

SQL> INSERT INTO appointment VALUES('A4','7-SEP-19','10:00 AM','W4','T4','B4');

1 row created.

SQL> INSERT INTO appointment VALUES('A5','15-OCT-19','1:30 PM','W5','T5','B5');

1 row created.

SQL> INSERT INTO appointment VALUES('A6','16-FEB-19','10:30 AM','W6','T6','B6');

1 row created.

SQL> INSERT INTO appointment VALUES('A7','10-MAR-19','10:30 AM','W7','T7','B7');

1 row created.
```

Figure 25 Insert Appointment

```
INSERT INTO appointment VALUES('A1','15-JAN-19','10:30 AM','W1','T1','B1');
INSERT INTO appointment VALUES('A2','8-SEP-19','11:30 AM','W2','T2','B2');
INSERT INTO appointment VALUES('A3','13-OCT-19','12:30 AM','W3','T3','B3');
INSERT INTO appointment VALUES('A4','7-SEP-19','10:00 AM','W4','T4','B4');
INSERT INTO appointment VALUES('A5','15-OCT-19','1:30 PM','W5','T5','B5');
INSERT INTO appointment VALUES('A6','16-FEB-19','10:30 AM','W6','T6','B6');
INSERT INTO appointment VALUES('A7','10-MAR-19','10:30 AM','W7','T7','B7');
```

```
SQL> INSERT INTO staff VALUES ('S1','Prashant','03-JAN-93','Doctor', 2000,'children','yes');

1 row created.

SQL> INSERT INTO staff VALUES ('S2','Kiran','04-FEB-92','Doctor', 3000,'Old-aged','yes');

1 row created.

SQL> INSERT INTO staff VALUES ('S8','Preeti','15-FEB-97','Nurse', 1000,'Old-aged','yes');

1 row created.

SQL> INSERT INTO staff VALUES ('S3','Ayuesh','17-JAN-90','Doctor', 2000,'Birth','no');

1 row created.

SQL> INSERT INTO staff VALUES ('S9','Priya','05-AUG-97','Nurse', 1000,'Birth','yes');

1 row created.

SQL> INSERT INTO staff VALUES ('S4','Krishna','28-MAR-87','Doctor', 4000,'Stomach','yes');

1 row created.

SQL> INSERT INTO staff VALUES ('S5','Jimmy','03-JAN-93','Doctor', 2500,'Therapy','yes');

1 row created.

SQL> INSERT INTO staff VALUES ('S6','Kurt','15-OCT-91','Doctor', 4000,'Heart','yes');

1 row created.

SQL> INSERT INTO staff VALUES ('S10','Prashant','19-JUN-93','Doctor', 2000,'ENT','yes');

1 row created.

SQL> INSERT INTO staff VALUES ('S11','Andy','18-JUN-92','Doctor', 2000,'brain','yes');

1 row created.
```

Figure 26 Insert Staff

INSERT INTO staff VALUES ('S1','Prashant','03-JAN-93','Doctor', 2000,'children','yes');

INSERT INTO staff VALUES ('S2','Kiran','04-FEB-92','Doctor', 3000,'Old-aged','yes');

INSERT INTO staff VALUES ('S8','Preeti','15-FEB-97','Nurse', 1000,'Old-aged','yes');

INSERT INTO staff VALUES ('S3','Ayuesh','17-JAN-90','Doctor', 2000,'Birth','no');

INSERT INTO staff VALUES ('S9','Priya','05-AUG-97','Nurse', 1000,'Birth','yes');

INSERT INTO staff VALUES ('S4','Krishna','28-MAR-87','Doctor', 4000,'Stomach','yes');

INSERT INTO staff VALUES ('S5','Jimmy','03-JAN-93','Doctor', 2500,'Therapy','yes');

INSERT INTO staff VALUES ('S6','Kurt','15-OCT-91','Doctor', 4000,'Heart','yes');

INSERT INTO staff VALUES ('S10','Prashant','19-JUN-93','Doctor', 2000,'ENT','yes');

INSERT INTO staff VALUES ('S11','Andy','18-JUN-92','Doctor', 2000,'brain','yes');

```
SQL> INSERT INTO  S_Address VALUES ('SA1','Nepal','Province no.3','Kathmandu',39,'Thamel Street');

1 row created.

SQL> INSERT INTO  S_Address VALUES ('SA2','Nepal','Province no.3','Hetauda',14,'Hetauda chowk');

1 row created.

SQL> INSERT INTO  S_Address VALUES ('SA3','Nepal','Province no.3','Sindhuli',11,'Dada');

1 row created.

SQL> INSERT INTO  S_Address VALUES ('SA4','Nepal','Province no.2','Birgunj',145,'Ghadiarwa');

1 row created.

SQL> INSERT INTO  S_Address VALUES ('SA5','Nepal','Province no.1','Biratnagar',25,'Birat Chowk');

1 row created.

SQL> INSERT INTO  S_Address VALUES ('SA6','Nepal','Province no.5','Butwal',29,'Traffic Chowk');

1 row created.

SQL> INSERT INTO  S_Address VALUES ('SA7','Nepal','Gandaki Pradesh','Pokhara',50,'Pardi');

1 row created.

SQL> INSERT INTO  S_Address VALUES ('SA8','Nepal','Province no.3','Kathmandu',13,'Kalkhu Chowk');

1 row created.

SQL> INSERT INTO  S_Address VALUES ('SA9','Nepal','Province no.3','Kathmandu',46,'Star hospital');

1 row created.

SQL> INSERT INTO  S_Address VALUES ('SA10','Nepal','Gandaki Pradesh','Pokhara',150,'Phewa Tal');

1 row created.
```

Figure 27 Insert Staff Address

INSERT INTO S_Address VALUES ('SA1','Nepal','Province no.3','Kathmandu',39,'Thamel Street');

INSERT INTO S_Address VALUES ('SA2','Nepal','Province no.3','Hetauda',14,'Hetauda chowk');

INSERT INTO S_Address VALUES ('SA3','Nepal','Province no.3','Sindhuli',11,'Dada');

INSERT INTO S_Address VALUES ('SA4','Nepal','Province no.2','Birgunj',145,'Ghadiarwa');

INSERT INTO S_Address VALUES ('SA5','Nepal','Province no.1','Biratnagar',25,'Birat Chowk');

INSERT INTO S_Address VALUES ('SA6','Nepal','Province no.5','Butwal',29,'Traffic Chowk');

INSERT INTO S_Address VALUES ('SA7','Nepal','Gandaki Pradesh','Pokhara',50,'Pardi');

INSERT INTO S_Address VALUES ('SA8','Nepal','Province no.3','Kathmandu',13,'Kalkhu Chowk');

INSERT INTO S_Address VALUES ('SA9','Nepal','Province no.3','Kathmandu',46,'Star hospital');

```
SQL> INSERT INTO S_Contact VALUES ('SC1',9843678955,9863887552,'Prashant@gmail.com',22128888);
1 row created.
SQL> INSERT INTO S_Contact VALUES ('SC2',9844798755,9831897573,'Kiran@gmail.com',22345868);
1 row created.
SQL> INSERT INTO S_Contact VALUES ('SC3',9840758955,9818764554,'Preeti@gmail.com',25227898);
1 row created.
SQL> INSERT INTO S_Contact VALUES ('SC4',9823778955,9841787555,'Ayuesh@gmail.com',28225786);
1 row created.
SQL> INSERT INTO S_Contact VALUES ('SC5',9828978555,9811687556,'Priya@gmail.com',25628899);
1 row created.
SQL> INSERT INTO S_Contact VALUES ('SC6',9845656955,9851785557,'Krishna@gmail.com',21228814);
1 row created.
SQL> INSERT INTO S_Contact VALUES ('SC7',9827878855,9841687558,'Jimmy@gmail.com',21828457);
1 row created.
SQL> INSERT INTO S_Contact VALUES ('SC8',9851077895,9851073358,'Kurt@gmail.com',21428253);
1 row created.
SQL> INSERT INTO S_Contact VALUES ('SC9',9801033795,9801076358,'Prashant@gmail.com',23128654);
1 row created.
SQL> INSERT INTO S_Contact VALUES ('SC10',9819778855,986378755,'Andy@gmail.com',67228888);
1 row created.
```

Figure 28 Insert Staff Contact

INSERT INTO S_Address VALUES ('SA10','Nepal','Gandaki Pradesh','Pokhara',150,'Phewa Tal');

INSERT INTO S_Contact VALUES ('SC1',9843678955,9863887552,'Prashant@gmail.com',22128888);

INSERT INTO S_Contact VALUES ('SC2',9844798755,9831897573,'Kiran@gmail.com',22345868);

INSERT INTO S_Contact VALUES ('SC3',9840758955,9818764554,'Preeti@gmail.com',25227898);

INSERT INTO S_Contact VALUES ('SC4',9823778955,9841787555,'Ayuesh@gmail.com',28225786);

INSERT INTO S_Contact VALUES ('SC5',9828978555,9811687556,'Priya@gmail.com',25628899);

INSERT INTO S_Contact VALUES ('SC6',9845656955,9851785557,'Krishna@gmail.com',21228814);

INSERT INTO S_Contact VALUES ('SC7',9827878855,9841687558,'Jimmy@gmail.com',21828457);

INSERT INTO S_Contact VALUES ('SC8',9851077895,9851073358,'Kurt@gmail.com',21428253);

INSERT INTO S_Contact VALUES ('SC9',9801033795,9801076358,'Prashant@gmail.com',23128654);

INSERT INTO S_Contact VALUES ('SC10',9819778855,986378755,'Andy@gmail.com',67228888);

```
SQL> INSERT INTO appointmentinfo VALUES('P1','S1','A1');
1 row created.
SQL> INSERT INTO appointmentinfo VALUES('P2','S2','A2');
1 row created.
SQL> INSERT INTO appointmentinfo VALUES('P2','S8','A2');
1 row created.
SQL> INSERT INTO appointmentinfo VALUES('P3','S3','A3');
1 row created.
SQL> INSERT INTO appointmentinfo VALUES('P3','S9','A3');
1 row created.
SQL> INSERT INTO appointmentinfo VALUES('P4','S4','A4');
1 row created.
SQL> INSERT INTO appointmentinfo VALUES('P5','S5','A5');
1 row created.
SQL> INSERT INTO appointmentinfo VALUES('P6','S6','A6');
1 row created.
SQL> INSERT INTO appointmentinfo VALUES('P7','S10','A7');
1 row created.
```

Figure 29 Insert Appointment Info

INSERT INTO appointmentinfo VALUES('P1','S1','A1');

INSERT INTO appointmentinfo VALUES('P2','S2','A2');

INSERT INTO appointmentinfo VALUES('P2','S8','A2');

INSERT INTO appointmentinfo VALUES('P3','S3','A3');

INSERT INTO appointmentinfo VALUES('P3','S9','A3');

INSERT INTO appointmentinfo VALUES('P4','S4','A4');

INSERT INTO appointmentinfo VALUES('P5','S5','A5');

INSERT INTO appointmentinfo VALUES('P6','S6','A6');

INSERT INTO appointmentinfo VALUES('P7','S10','A7');

```
SQL> INSERT INTO staffcontactinfo VALUES('S1','SA1','SC1');
1 row created.
SQL> INSERT INTO staffcontactinfo VALUES('S2','SA2','SC2');
1 row created.
SQL> INSERT INTO staffcontactinfo VALUES('S8','SA3','SC3');
1 row created.
SQL> INSERT INTO staffcontactinfo VALUES('S3','SA4','SC4');
1 row created.
SQL> INSERT INTO staffcontactinfo VALUES('S9','SA5','SC5');
1 row created.
SQL> INSERT INTO staffcontactinfo VALUES('S4','SA6','SC6');
1 row created.
SQL> INSERT INTO staffcontactinfo VALUES('S5','SA7','SC7');
1 row created.
SQL> INSERT INTO staffcontactinfo VALUES('S6','SA8','SC8');
1 row created.
SQL> INSERT INTO staffcontactinfo VALUES('S10','SA9','SC9');
1 row created.
SQL> INSERT INTO staffcontactinfo VALUES('S11','SA10','SC10');
1 row created.
```

Figure 30 insert staff contact info

```
INSERT INTO staffcontactinfo VALUES('S1','SA1','SC1');
INSERT INTO staffcontactinfo VALUES('S2','SA2','SC2');
INSERT INTO staffcontactinfo VALUES('S8','SA3','SC3');
INSERT INTO staffcontactinfo VALUES('S3','SA4','SC4');
INSERT INTO staffcontactinfo VALUES('S9','SA5','SC5');
INSERT INTO staffcontactinfo VALUES('S4','SA6','SC6');
INSERT INTO staffcontactinfo VALUES('S5','SA7','SC7');
INSERT INTO staffcontactinfo VALUES('S6','SA8','SC8');
INSERT INTO staffcontactinfo VALUES('S10','SA9','SC9');
```

INSERT INTO staffcontactinfo VALUES('S11','SA10','SC10');

Showing the data using select command

```
SQL> select * from patient;

PATIEN P_NAME                    DOB       ISS GENDER        AGE STAFF_ TYPE
------ ------------------------- --------- --- ------ ---------- ------ --------
P1     John                      06-JAN-00 no  Male           19        new
P2     Max                       08-FEB-97 no  Male           22        new
P3     Tom                       01-OCT-89 no  Male           30        new
P4     Jim                       01-SEP-01 no  Male           18        regular
P5     Natalia                   05-NOV-65 no  Female         54        new
P6     Pam                       06-AUG-79 no  Female         40        new
P7     Andy                      18-JUN-92 yes Male           27 S11    regular

7 rows selected.
```

Figure 31 Select Patient

select * from patient;

```
SQL> set linesize 150
SQL> select * from address;

A_ID   COUNTRY                  PROVINCE                 CITY                      STREET_NO STREET
------ ------------------------ ------------------------ ------------------------- --------- -------------------------
A1     Nepal                    Province no.3            Kathmandu                       139 Jyatha Street
A2     Nepal                    Province no.3            Hetauda                           4 Hetauda City College
A3     Nepal                    Province no.3            Dolakha                          21 Kalinchok
A4     Nepal                    Province no.2            Birgunj                          45 Shankaracharya Gate
A5     Nepal                    Province no.1            Damak                            65 Damak Hospital
A6     Nepal                    Province no.5            Butwal                           39 Siddha Baba Temple
A7     Nepal                    Gandaki Pradesh          Pokhara                         150 Phewa Tal

7 rows selected.
```

Figure 32 Select Address

select * from address;

```
SQL> select * from contact;

CON_ID P_PHONE_NO    CELL_NO EMAIL                          FAX
------ ---------- ---------- ------------------------- ----------
C1     9813778955  981378755 john@gmail.com              22228888
C2     9814798755  984178757 max@gmail.com               12345868
C3     9841768955  981778455 tom@gmail.com               45225868
C4     9840978955  982378755 Jim@gmail.com               78225868
C5     9815678555  981568755 Natalia@gmail.com           22228888
C6     9818756955  981678555 Pam@gmail.com               56228888
C7     9819778855  986378755 Andy@gmail.com              67228888

7 rows selected.
```

Figure 33 Select Contact

Select * from contact;

```
SQL> select * from contact_info;

PATIEN A_ID   CON_ID
------ ------ ------
P1     A1     C1
P2     A2     C2
P3     A3     C3
P4     A4     C4
P5     A5     C5
P6     A6     C6
P7     A7     C7

7 rows selected.

SQL>
```

Figure 34 Select Contact Info

Select * from contact_info;

```
SQL> select * from ward
  2  ;

WARD_N WARD_NAME       FLOOR
------ ---------- ----------
W1     Pediatrics          1
W2     Geriatrics          2
W3     Maternity           5
W4     Endoscopy           4
W5     Therapy             2
W6     Cardiology          2
W7     ENT                 1

7 rows selected.
```

Figure 35 Select Ward

Select * from ward

```
SQL> select * from treatment
  2  ;

TREATM TREATMENT                 TREATMENT_TYPE
------ ------------------------- -------------------------
T1     Acute sinusitis           Vaccine therapy
T2     Weight Loss               health regime
T3     child healthcare          consultant
T4     Gastrointestinal tract    stomach
T5     exercise routine          heart disease
T6     heart                     artery block
T7     ear                       ear wax

7 rows selected.
```

Figure 36 Select Treatment

Select * from treatment;

```
SQL> select * from bill;

BILL_I BILL_DATE     AMOUNT PAYMENTMET
------ --------- ---------- ----------
B1     16-JAN-19       2000 Cash
B2     09-SEP-19       1000 Cash
B3     14-OCT-19       1000 Cash
B4     08-SEP-19       2000 Cash
B5     16-OCT-19       2000 Cash
B6     17-FEB-19       2000 Cash
B7     11-MAR-19       2000 E-Cash

7 rows selected.
```

Figure 37 Select Bill

Select * from bill;

```
SQL> select * from appointment;

APPOIN APPOINTME TIME     WARD_N TREATM BILL_I
------ --------- -------- ------ ------ ------
A1     15-JAN-19 10:30 AM W1     T1     B1
A2     08-SEP-19 11:30 AM W2     T2     B2
A3     13-OCT-19 12:30 AM W3     T3     B3
A4     07-SEP-19 10:00 AM W4     T4     B4
A5     15-OCT-19 1:30 PM  W5     T5     B5
A6     16-FEB-19 10:30 AM W6     T6     B6
A7     10-MAR-19 10:30 AM W7     T7     B7

7 rows selected.
```

Figure 38 Select Appointment

Select * from appointment;

```
SQL> select * from appointmentinfo;

PATIEN STAFF_ APPOIN
------ ------ ------
P1     S1     A1
P2     S2     A2
P2     S8     A2
P3     S3     A3
P3     S9     A3
P4     S4     A4
P5     S5     A5
P6     S6     A6
P7     S10    A7

9 rows selected.
```

Figure 39 Select AppointmentInfo

Select * from appointmentinfo;

```
SQL> set linesize 144;
SQL> select * from staff;

STAFF_ S_NAME                    S_DOB     CATEGORY STAFF_COMMISSION SPECIALI CERTIFIE
------ ------------------------- --------- -------- ---------------- -------- --------
S1     Prashant                  03-JAN-93 Doctor               2000 children yes
S2     Kiran                     04-FEB-92 Doctor               3000 Old-aged yes
S8     Preeti                    15-FEB-97 Nurse                1000 Old-aged yes
S3     Ayuesh                    17-JAN-90 Doctor               2000 Birth    no
S9     Priya                     05-AUG-97 Nurse                1000 Birth    yes
S4     Krishna                   28-MAR-87 Doctor               4000 Stomach  yes
S5     Jimmy                     03-JAN-93 Doctor               2500 Therapy  yes
S6     Kurt                      15-OCT-91 Doctor               4000 Heart    yes
S10    Prashant                  19-JUN-93 Doctor               2000 ENT      yes
S11    Andy                      18-JUN-92 Doctor               2000 brain    yes

10 rows selected.
```

Figure 40 Select Staff

Select * from staff;

```
SQL> select * from s_address;

S_A_ID S_COUNTRY                 S_PROVINCE                S_CITY                    S_STREET_N S_STREET_NAME
------ ------------------------- ------------------------- ------------------------- ---------- -------------------------
SA1    Nepal                     Province no.3             Kathmandu                 39         Thamel Street
SA2    Nepal                     Province no.3             Hetauda                   14         Hetauda chowk
SA3    Nepal                     Province no.3             Sindhuli                  11         Dada
SA4    Nepal                     Province no.2             Birgunj                   145        Ghadiarwa
SA5    Nepal                     Province no.1             Biratnagar                25         Birat Chowk
SA6    Nepal                     Province no.5             Butwal                    29         Traffic Chowk
SA7    Nepal                     Gandaki Pradesh           Pokhara                   50         Pardi
SA8    Nepal                     Province no.3             Kathmandu                 13         Kalkhu Chowk
SA9    Nepal                     Province no.3             Kathmandu                 46         Star hospital
SA10   Nepal                     Gandaki Pradesh           Pokhara                   150        Phewa Tal

10 rows selected.
```

Figure 41 select Staff Address

Select * from s_address;

```
SQL> select * from s_contact;

S_CON_ S_PHONE_NO  S_CELL_NO S_EMAIL                        S_FAX
------ ---------- ---------- ------------------------- ----------
SC1    9843678955 9863887552 Prashant@gmail.com          22128888
SC2    9844798755 9831897573 Kiran@gmail.com             22345868
SC3    9840758955 9818764554 Preeti@gmail.com            25227898
SC4    9823778955 9841787555 Ayuesh@gmail.com            28225786
SC5    9828978555 9811687556 Priya@gmail.com             25628899
SC6    9845656955 9851785557 Krishna@gmail.com           21228814
SC7    9827878855 9841687558 Jimmy@gmail.com             21828457
SC8    9851077895 9851073358 Kurt@gmail.com              21428253
SC9    9801033795 9801076358 Prashant@gmail.com          23128654
SC10   9819778855  986378755 Andy@gmail.com              67228888

10 rows selected.
```

Figure 42 Select Staff Contact

Select * from s_contact;

```
SQL> select * from staffcontactinfo;

STAFF_ S_A_ID S_CON_
------ ------ ------
S1     SA1    SC1
S2     SA2    SC2
S8     SA3    SC3
S3     SA4    SC4
S9     SA5    SC5
S4     SA6    SC6
S5     SA7    SC7
S6     SA8    SC8
S10    SA9    SC9
S11    SA10   SC10

10 rows selected.
```

Figure 43 Select Staff Contact Info

Select * from staffcontactinfo

Database querying

Information Queries:

```
SQL> select patient_id, p_name, type from patient where type='new' or type='regular';

PATIEN P_NAME                    TYPE
------ ------------------------- --------
P1     John                      new
P2     Max                       new
P3     Tom                       new
P4     Jim                       regular
P5     Natalia                   new
P6     Pam                       new
P7     Andy                      regular

7 rows selected.

SQL>
```

Figure 44 Regular or new patient query

In the figure 44, i simply showed all the list of patients who are regular or new.

Code:

select patient_id, p_name, type from patient where type='new' or type='regular';

```
SQL> select patient.patient_id, patient.p_name, address.country, address.city, address.street,
  2  address.street_no, address.province from patient join
  3  contact_info on patient.patient_id= contact_info.patient_id join address on
  4  contact_info.a_id=address.a_id order by patient_id;

PATIEN P_NAME                    COUNTRY                  CITY                     STREET                   STREET_NO PROVINCE
------ ------------------------- ------------------------ ------------------------- ------------------------- ---------- -------------------------
P1     John                      Nepal                    Kathmandu                Jyatha Street                    139 Province no.3
P2     Max                       Nepal                    Hetauda                  Hetauda City College               4 Province no.3
P3     Tom                       Nepal                    Dolakha                  Kalinchok                         21 Province no.3
P4     Jim                       Nepal                    Birgunj                  Shankaracharya Gate               45 Province no.2
P5     Natalia                   Nepal                    Damak                    Damak Hospital                    65 Province no.1
P6     Pam                       Nepal                    Butwal                   Siddha Baba Temple                39 Province no.5
P7     Andy                      Nepal                    Pokhara                  Phewa Tal                        150 Gandaki Pradesh

7 rows selected.
```

Figure 45 patient and address join

In the figure 45, I have used select query to join two tables which are patient and address.

select patient.patient_id, patient.p_name, address.country, address.city, address.street, address.street_no, address.province from patient join contact_info on patient.patient_id= contact_info.patient_id join address on contact_info.a_id=address.a_id order by patient_id;

```
SQL> select staff.staff_id, staff.s_name, staff.category,staff.certified, appointment.Appointment_id , appointment.appointment_date,
  2  staff.staff_commission from staff join appointmentinfo
  3  on staff.staff_id =appointmentinfo.staff_id join appointment on appointmentinfo.Appointment_id = appointment.Appointment_id  where category='Doctor' and certified ='yes';

STAFF_ S_NAME                    CATEGORY CERTIFIE APPOIN APPOINTME STAFF_COMMISSION
------ ------------------------- -------- -------- ------ --------- ----------------
S1     Prashant                  Doctor   yes      A1     15-JAN-19             2000
S2     Kiran                     Doctor   yes      A2     08-SEP-19             3000
S4     Krishna                   Doctor   yes      A4     07-SEP-19             4000
S5     Jimmy                     Doctor   yes      A5     15-OCT-19             2500
S6     Kurt                      Doctor   yes      A6     16-FEB-19             4000
S10    Prashant                  Doctor   yes      A7     10-MAR-19             2000

6 rows selected.
```

Figure 46 certified staff appointment

In the figure 46, I have joined staff with appointment where staff is doctor and doctor is certified.

select staff.staff_id, staff.s_name, staff.category,staff.certified, appointment.Appointment_id , appointment.appointment_date,

staff.staff_commission from staff join appointmentinfo

on staff.staff_id =appointmentinfo.staff_id join appointment on appointmentinfo.Appointment_id = appointment.Appointment_id where category='Doctor' and certified ='yes';

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

PATIEN P_NAME                    DOB       ISS GENDER        AGE STAFF_ TYPE
------ ------------------------- --------- --- ------ ---------- ------ --------
P7     Andy                      18-JUN-92 yes Male           27 S11    regular

SQL>
```

Figure 47 staff patient

In the figure 47, I have used simple select query where patient is staff.

Select * from patient where isstaff='yes';

Transaction Queries:

```
SQL> SELECT staff.staff_id, staff.s_name, staff.category, staff.certified, appointment.appointment_id, appointment.appointment_date,
  2  staff.staff_commission from staff join appointmentinfo
  3  on staff.staff_id = appointmentinfo.staff_id join appointment on appointmentinfo.appointment_id = appointment.appointment_id
  4  where category= 'Doctor' and certified='no';

STAFF_ S_NAME                    CATEGORY CERTIFIE APPOIN APPOINTME STAFF_COMMISSION
------ ------------------------- -------- -------- ------ --------- ----------------
S3     Ayuesh                    Doctor   no       A3     13-OCT-19             2000

SQL>
```

Figure 48 uncertified doctor appointment

In the given figure I have join staff and appointment where staff is uncertified.

SELECT staff.staff_id, staff.s_name, staff.category, staff.certified, appointment.appointment_id, appointment.appointment_date,

staff.staff_commission from staff join appointmentinfo

on staff.staff_id = appointmentinfo.staff_id join appointment on appointmentinfo.appointment_id = appointment.appointment_id

where category= 'Doctor' and certified='no';

```
SQL> select appointment.appointment_id, appointment.appointment_date, appointment.Time, ward .ward_name from appointment join ward on appointment.ward_no=ward.ward_no
  2  where ward.ward_name='Emergency';

APPOIN APPOINTME TIME     WARD_NAME
------ --------- -------- ----------
A8     11-MAR-19 10:35 AM Emergency

SQL>
```

Figure 49 emergency ward

In the figure 49, the appointment and ward is joined where ward name is emergency.

select appointment.appointment_id, appointment.appointment_date, appointment.Time, ward .ward_name from appointment join ward on appointment.ward_no=ward.ward_no

where ward.ward_name='Emergency';

```
SQL> select staff.staff_id, staff.s_name, appointment.Appointment_id , appointment.appointment_date
  2  from staff join appointmentinfo
  3  on staff.staff_id =appointmentinfo.staff_id join appointment on appointmentinfo.Appointment_id = appointment.Appointment_id where appointment_date='08-SEP-19';

STAFF_ S_NAME                    APPOIN APPOINTME
------ ------------------------- ------ ---------
S2     Kiran                     A2     08-SEP-19
S8     Preeti                    A2     08-SEP-19
```

Figure 50 Appointment with date

in the figure 50, the appointment and staff is joined such that specified date is given.

select staff.staff_id, staff.s_name, appointment.Appointment_id , appointment.appointment_date

from staff join appointmentinfo

on staff.staff_id =appointmentinfo.staff_id join appointment on appointmentinfo.Appointment_id = appointment.Appointment_id where appointment_date='08-SEP-19';

```
SQL> select patient.patient_id, patient.p_name,appointment.Appointment_id, appointment.appointment_date
  2  from patient join appointmentinfo
  3  on patient.patient_id =appointmentinfo.patient_id join appointment on appointmentinfo.Appointment_id = appointment.Appointment_id where appointment_date='15-JAN-19';

PATIEN P_NAME                    APPOIN APPOINTME
------ ------------------------- ------ ---------
P1     John                      A1     15-JAN-19
```

Figure 51 patient appointment with date

In the figure 51, the appointment and patient is joined with specified date.

select patient.patient_id, patient.p_name,appointment.Appointment_id, appointment.appointment_date

from patient join appointmentinfo

on patient.patient_id =appointmentinfo.patient_id join appointment on appointmentinfo.Appointment_id = appointment.Appointment_id where appointment_date='15-JAN-19';

Critical evaluation

This coursework was very hard for me at first. It was very hard to understand about the scenarios of this coursework. The coursework was about hospital management system which consisted mainly of patient, appointment, staff, ward and treatment. There were some scenarios for patient as regular or normal patient should be noted. Then for doctor in staff, a certified or uncertified doctor should be noted and free treatment is being given to the certified doctor and uncertified doctor would not get that kind of privilege.

At first I could not understand the scenario, so I asked my module leader and other related teachers for guidance. I learned many things from the teachers about the scenarios which helped me to start the coursework. I started making attributes for each entity which were patient, staff and appointment. I had the idea of making another entity which were treatment, ward and prescription but I did not took the idea because my teacher suggested me to only make three entities which would make my coursework more easier. Then I stored treatment, ward and bill attributes inside the appointment entity which was better idea which would help me in getting better results in 3nf. I reviewed my initial er diagram to our related teachers who confirmed me about my initial er diagram just one week before submission. Then I immediately started normalization process in my copy reviewing to my teachers. I also took the idea using youtube videos which helped me in better understanding normalization about anomalies, partial and transitive dependency. Then I finished my normalization in two days and I started my documentation and final ER diagram. Then I started creating, inserting and queries for my coursework.

In the end, I was able to finish my coursework through hard work and perseverance with the support of my teachers and friends who helped me in each problem.

Critical assessment

This research has been one of the hardest tasks of the semester. Since it was the very first time, I used the oracle database, I found it somewhat difficult to complete the mission, but during the process I gained knowledge of various aspects of the database. As a computer student, I need to work with huge amounts of data, so I need to have adequate knowledge about handling and retrieving the stored date. I think the coursework will enable me to compute not only in my academics but also in my professional life, because expertise is what is required in professional life to be a good professional in any area.

To order to overcome the problems that occurred during the course work, I consulted with my module leader that I also studied and updated the lecture slides to overcome the difficulties that I was experiencing. I will now be able to implement the same knowledge in other modules after acquiring all the necessary knowledge on the Oracle database. If I had this information before doing the assignment, I would have put it into practice in my previous assignments where we had to deal with different organizations in the module where basic database development was to be done.

Therefore, I will now be able to implement this knowledge on other coursework after acquiring practical knowledge on the Oracle database. I agree that this course work has helped me develop my skills on different aspects of the database, which will be useful in my other course work as well as in the future.