Doctor Appointment System

MEDICO

(By Swetha Annavarapu – 16011A0537
Afreen Sultana T– 16011A0538
Yashaswi Veerepalli– 16011A0546)

Software Requirements Specification

Table of Contents

1. Abstract

Life is becoming too busy to get medical appointments in person and to maintain a proper health care. The main idea of this work is to provide ease and comfort to patients while taking appointment from doctors and it also resolves the problems that the patients has to face while making an appointment. The application Medico acts as a user friendly application where the patient can book appointments online at a nearby hospital, or chat with a verified doctor in case of small emergencies, or can order medicines online by specifying the address, or can book slots for tests and scans at a hospital laboratory.

2. Introduction

This document goes over the requirements that must be fulfilled before Medico can be delivered to the customer. The first section discusses the scope of the application, any definitions or acronyms used, along with the organization of the SRS document.

2.1Purpose of this document

The purpose of this document is to present a detailed description of a Personal Medical Record application. It explains the purpose and features of the application itself, the web interface through which a user can access a personal account and the server that hosts the user accounts.

2.2 Scope of this document

This document is meant to describe the entire product, its intended features and prescribed use. This application is used to provide the customer an easy way to book doctor appointments, order medicines, communication with doctor and book tests and scans. This Personal Medical Record application is used to store personal medical history. Doctors would be able to add information to the patient's Medico account at anytime, including at an appointment, with the patient's consent. This includes diagnoses, treatments, medications, allergies, and medical procedures. Medico would allow for uniform communication of information between the patient and their respective doctor(s). The software would label and organize information authored by health care providers separately from information authored by the patient to maintain authenticity.

2.3 Overview

Based on the results of the requirements elicitation, the system is described as: a patient appointment scheduling system whose only user is to be the secretary making the appointments. The system is to store appointment time, type, date, location and personnel data as well as patient data to include name, address and phone number/s. The user will be able to schedule appointments for any day in the future, and a mechanism by which corrections and updates to the data can be made, is to exist also. This information can be displayed by the user upon request. It also has the option of chatting with the verified doctor in case of small emergencies. We can also order the medicines online and track the delivery until the order is delivered. We can also book tests and scans.

2.4 Definitions, Acronyms and Abbreviations

Define all terms, acronyms, and abbreviations need to understand the SRS. If this section is extensive, then move to an appendix. It is also possible to provide a link to other resources for extensive terminology explanation.

SRS - Software Requirements Specification
DBMS- Database Management System
MySQL - a query language to interrogate the system
GUI - Graphical User Interface
JDBC- Java Database Connectivity
Medico- The Name of the Application
Patient- The User of Application who books appointments, tests or scans, chat with a doctor and order medicines.
Doctor- A certified doctor who treats patients and prescribes medicines or tests if necessary.
Appointment- Patients book appointments with the required doctor depending upon their problem or book slots for tests.
Online Application- The medium which enables the user to book appointments online for a doctor and runs on internet.
Hospital Scheduling- A schedule is maintained so that no two appointments collide with each other.
Medicine- Necessary drugs prescribed by a doctor can be ordered online and delivery is made to the specified address.
Test- Laboratory tests such as blood test, thyroid test etc. are performed at a hospital laboratory.
Medical Records- An account of patient's medical history.
Delivery Boy- The delivery boy delivers the medicines from the pharmacy to the specified address.
Login ID - a user identification number to enter the system
Password - a word that enables one to gain admission into the system

3. General Description

3.1 Product Functions

The product stores appointment and patient records, allows for addition, deletion and modification of those records and provides views of those records. It also allows the patients to chat with the doctor, chat with the doctor and book tests and scans.

3.2 User Characteristics

The user has standard experience with commercial office productivity software and standard PC operating systems and has heretofore accomplished this application with those products. Although the user does have an extensive background within the application's domain, this is to be the first software product specifically designed for this purpose that the user will encounter.

The Medico Office has sufficient, well skilled and highly experienced doctors. It keeps track of when doctors are available and allows clients to make appointments. The system shall provide a list of appointments for next business day so the secretary can contact clients to confirm their appointments. It should display, during regular business hours, clients and the rooms the clients are assigned to and which dentist each client is waiting to see, and the location of each dentist and each assistant.

The user desires a system whereby the appointment scheduling and manipulation occur as described, but the system would also have the following capabilities: viewing of appointments by matching patient, viewing of appointments by doctors and staff, appointment reminder tracking (for use by the user).

3.3 General Constraints

- The system must be implemented on a standard Operating System.
- The Application can be accessed through Web and must be connected to the Internet.
- The User must have an account in the application in order to book appointments.
- Available hours are Monday through Friday: 9 AM to 5 PM.
- Available doctors, lab technicians are to be system parameters.
- The system will include the system parameter values of at least 5 Hospitals or laboratories nearby to the location.
- System parameters cannot be changed by the user.
- The system has no industry standard protocol constraints.
- The patient data stored by the system is secured.

4. Specific Requirements

4.1 Functional Requirements

1. Patient Management
Patient information and his/her area information will be registered. It is mandatory to Patient to provide his residential area during registration. When Patient will go online for health advice, application will prioritize his area's Doctors. If no Doctor is available in his area then system will suggest nearby Doctors.
Patient can update his Profile after registration.
Patients can see registered doctors list who are online at that moment.
Patients can also take advice from a doctor for a specific disease.
Patient can order medicines online.
Patient can also book slots for tests and scans.

2. Doctor Management
Doctor information and his/her area information will be registered. It is mandatory to Doctor to provide his residential area during registration. Registered doctors list can be seen by Patients.
Doctor can give advice & health tips to the patient about their disease and also consult him for a specific doctor if required.

3. Appointments
Patients can take online appointment for a specific doctor. For the appointment patient has to fill up a form, in the form the time of appointment for a specific doctor will be mentioned, Appointments will be given only if Doctor is free during that time. After submitting the form final report will be generated in which information about the clinic name, patient name, appointment time etc. will be included. Patient can print this report as well as report will be mailed to patient's registered mail address.

4. Request Management
Patient can put online request for text chat to the Doctor. Patient can communicate with doctor via live chat regarding their health problems & also get the health tips.

5. Feedback Management
A patient can enter his feedback about Doctor if any. Admin can see the list of feedback.

6. Health Tips
Patient can select his/her disease and get the health tips entered by expert doctors.

7. Delivery Boy

The delivery boy delivers the medicines ordered by a patient to the specified address and there is a tracking of the delivery.

4.2 Non - Functional Requirements

1.Security

Program requires log on, records must be secure.

2.Reliability

The system must provide all functionality whenever in use. System crashes are not to impair the data. System crashes are not to impair the system.

3.Portability

The application can be accessed either from a PC or a mobile phone.

4.Extensibility

The system must be modifiable, as described above, for future enhancements.

5.Reusability

The system components must be reusable for the current and future versions of this product.

4.3 External Interface Requirements

All the interactions of the software with patients, doctors, hardware and software are specified here.

1.User Interfaces

The user interface is designed in JDBC where SQL queries are implemented. The developer will have to study the designing of the product. The user of the product will get very user-friendly application which will be very easy to work with.

2.Hardware Interfaces

There are not direct hardware interfaces in the system. The operating systems on both the client and server side will be configured to handle the hardware interfaces.

3.Software Interfaces

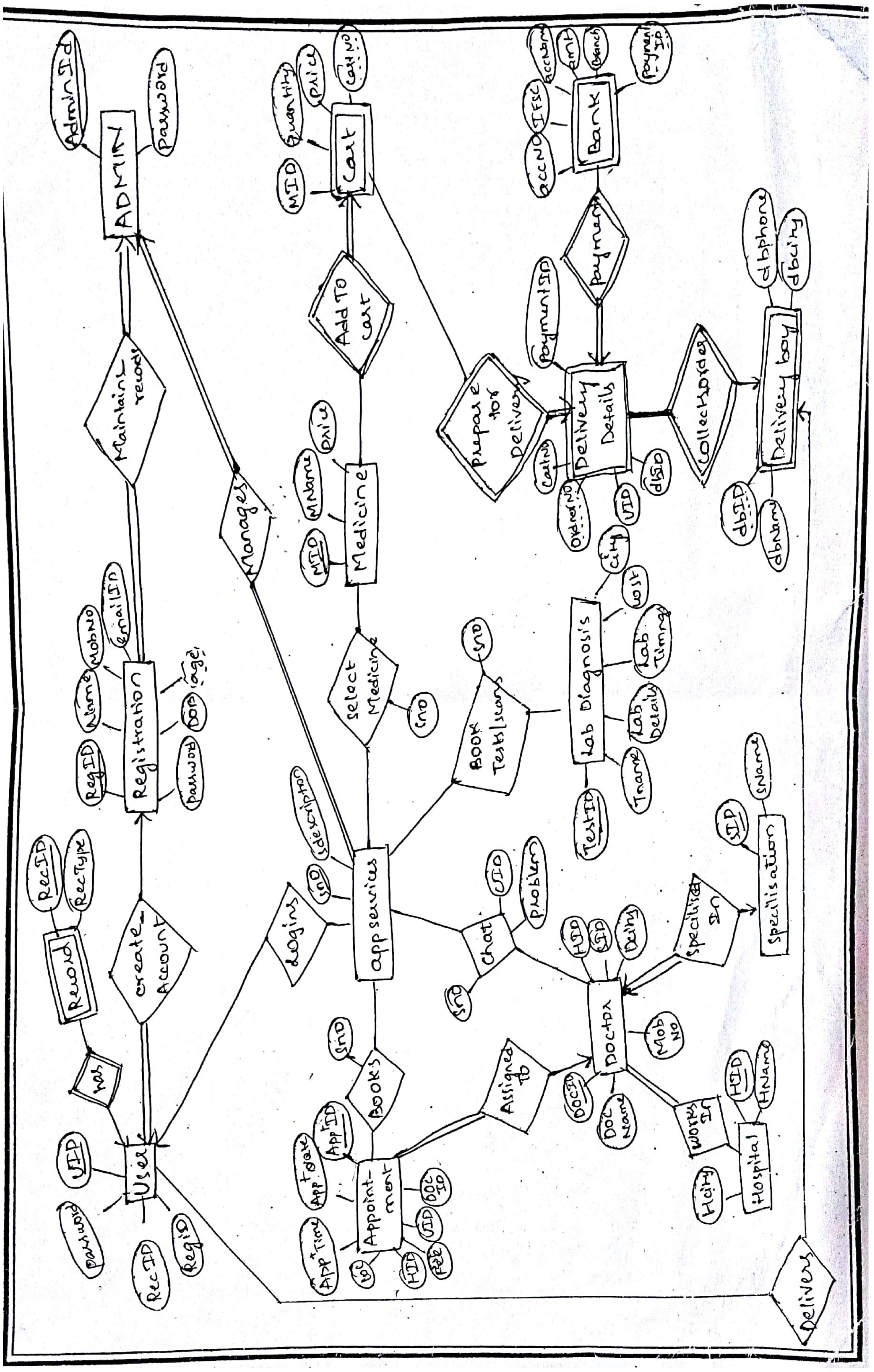
Operating System: Windows, Linux, Mac OS
Front End: JDBC
Back End: MySQL

4.Communication Interfaces

Communication is done through internet. Mails will be sent to the patient regarding the scheduled appointments or the track of the medicine delivery.

5.References

- https://www.practo.com/
- http://www.cse.msu.edu/~cse435/Projects/F09/Deliverables/Incoming/SRS-Rev1/iMedLife%20SRS.doc


ADMIN
AdminID
Password
Maintains records
Manages
Registration
RegID
Name
MobNO
EmailID
DOB
Password
create Account
Record
RecID
RecType
has
User
UID
Password
RecID
RegID
Logins
App services
SID
Sdescription
Cart
MID
Quantity
Price
CartNO
Add To cart
Medicine
MID
MName
Price
select Medicine
SNO
BOOK Tests/scans
SID
Lab Diagnosis
TestID
Tname
Lab Details
Lab Timings
cost
city
Chat
CID
problem
SNO
Doctor
HID
SID
Dcity
Mob No
DOCID
DOC Name
Specialised In
Specialisation
SID
SName
Works In
Hospital
HID
HName
Hcity
Assigned to
Books
SNO
Appointment
App ID
App Date
App Time
DOC ID
UID
HID
Fee
SNO
Prepare for Delivery
Delivery Details
CartNo
OrderNO
UID
dbID
paymentID
Payment
Bank
accNO
IFSC
accname
amt
Branch
Payment ID
Collect order
Delivery boy
dbID
dbname
dbphone
dbcity
Delivers

Schema Diagrams

Design of a database is called the schema. Schema is of three types: Physical schema, logical schema and view schema.

The design of a database at physical level is called **physical schema**, how the data stored in blocks of storage is described at this level.

Design of database at logical level is called **logical schema**, programmers and database administrators work at this level, at this level data can be described as certain types of data records gets stored in data structures, however the internal details such as implementation of data structure is hidden at this level (available at physical level).

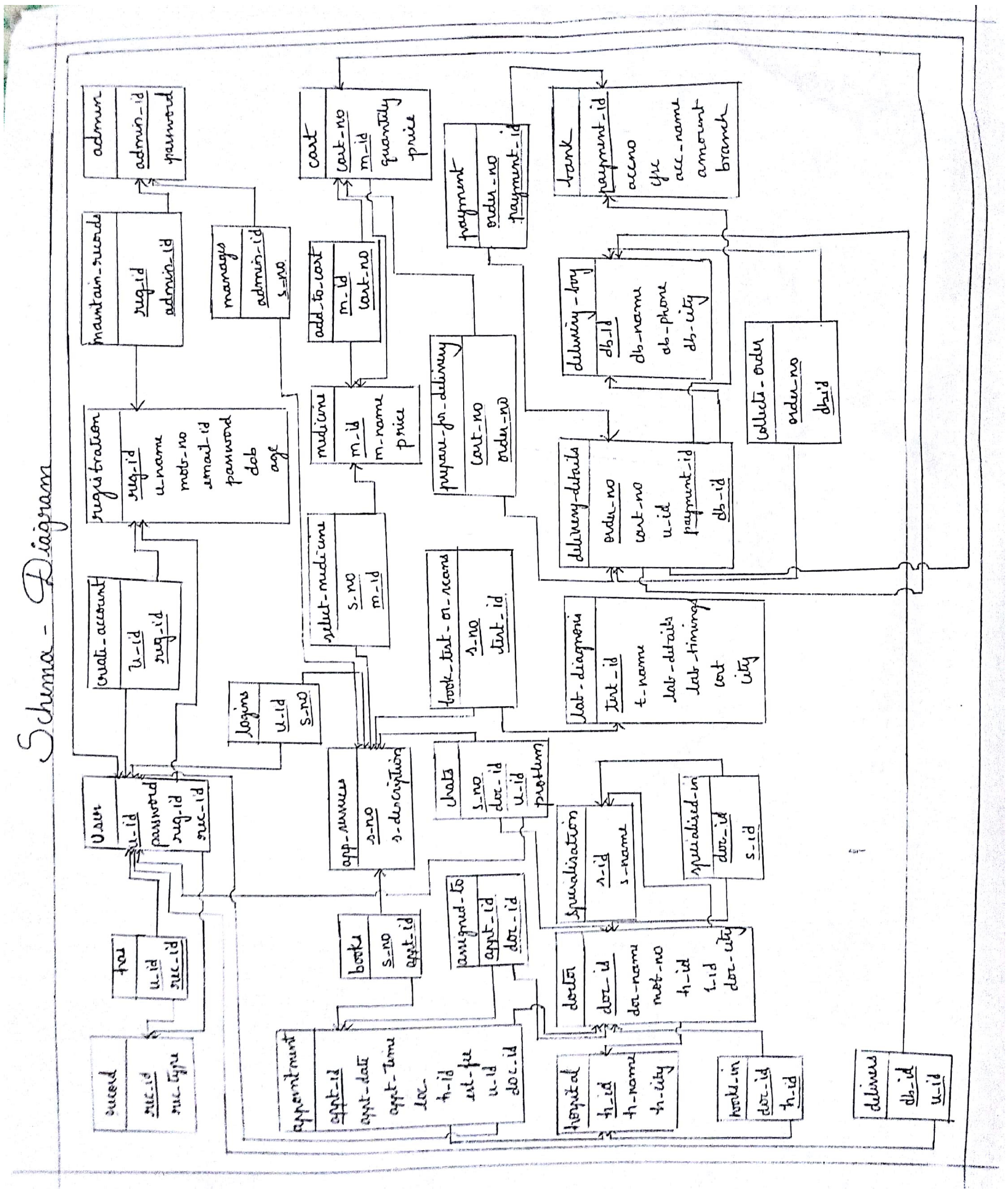
Design of database at view level is called **view schema.** This generally describes end user interaction with database systems.

A database schema, along with primary key and foreign key dependencies, can be depicted by *schema diagrams.*

The following shows Schema diagram for the **Online Doctor Appointment System.** Each relation appears as a box, with the relation name at the top and attributes listed inside the box.

Primary key attributes are shown underlined. Foreign key dependencies appear as arrows from foreign key attributes of the referring relation to the primary key of the referencing relation.

Referential integrity constraints other than foreign key constraints shown explicitly in schema diagram.


Schema - Diagram
admin
admin_id
password
maintain_records
reg_id
admin_id
manages
admin_id
s_no
registration
reg_id
u_name
mob_no
email_id
password
dob
age
create_account
u_id
reg_id
logins
u_id
s_no
User
u_id
password
reg_id
rec_id
has
u_id
rec_id
record
rec_id
rec_type
cart
cart_no
m_id
quantity
price
add_to_cart
m_id
cart_no
medicine
m_id
m_name
price
select_medicine
s_no
m_id
app_services
s_no
s_description
books
s_no
appt_id
appointment
appt_id
appt_date
appt_time
loc
h_id
est_fee
u_id
doc_id
payment
order_no
payment_id
prepare_for_delivery
cart_no
order_no
book_test_or_scans
s_no
test_id
chats
s_no
doc_id
u_id
problem
assigned_to
appt_id
doc_id
bank
payment_id
accno
ifsc
acc_name
amount
branch
delivery_boy
db_id
db_name
db_phone
db_city
delivery_details
order_no
cart_no
u_id
payment_id
db_id
collects_order
order_no
db_id
lab_diagnosis
test_id
t_name
lab_details
lab_timings
cost
city
specialisation
s_id
s_name
specialised_in
doc_id
s_id
doctor
doc_id
doc_name
mob_no
h_id
l_id
doc_city
hospital
h_id
h_name
h_city
works_in
doc_id
h_id
delivers
db_id
u_id

SCHEMA REFINEMENT - NORMALIZATION

FIRST NORMAL FORM:

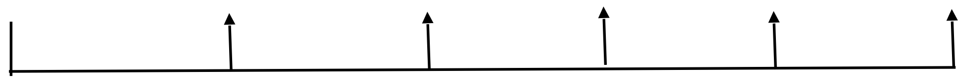
Let us consider the doctor table to verify First Normal Form(1NF)

TABLE: doctor (doc_id, doc_name, mob_no, h_id, s_id,doc_city)

FUNCTIONAL DEPENDENCIES:

The functional dependencies are: -

I	N	M	H	S	C
doc_id	doc_name	mob_no	h_id	s_id	doc_city



So, the functional dependencies of table are: -
FD1: I -> N,M,H,S,C

ATTRIBUTE CLOSURE:

1. Consider attribute I whose attribute closure is represented by I+
   Initially I+ = {I}
   From I -> N,M,H,S,C, I+ = {I, N, M,H,S,C}
2. Consider attribute N whose attribute closure is represented by N+
   Initially N+ = {N}
   Since there are no FDs of the form N -> y
   N+ = {N}
3. Consider attribute M whose attribute closure is represented by M+

   Initially M+ = {M}
   Since there are no FDs of the form M -> y
   M+ = {M}
4. Consider attribute H whose attribute closure is represented by H+

   Initially H+ = {H}
   Since there are no FDs of the form H -> y
   H+ = {H}

5. Consider attribute S whose attribute closure is represented by S+
   Initially S+ = {S}
   Since there are no FDs of the form S -> y
   S+ = {S}
6. Consider attribute C whose attribute closure is represented by C+
   Initially C+ = {C}

Since there are no FDs of the form C -> y
C+ = {C}

KEY IDENTIFICATION:

In the above, since I+ contains all the attributes, hence I is the primary key.

CHECK FOR 1NF:

In doctor table, attribute mob_no is multivalued i.e., it is not atomic which is against 1NF rule. So we add an additional mob_type. And then we make the primary key as doc_id, mob_type.

Hence, the primary key for this table is- doc_id, mob_type

SECOND NORMAL FORM:

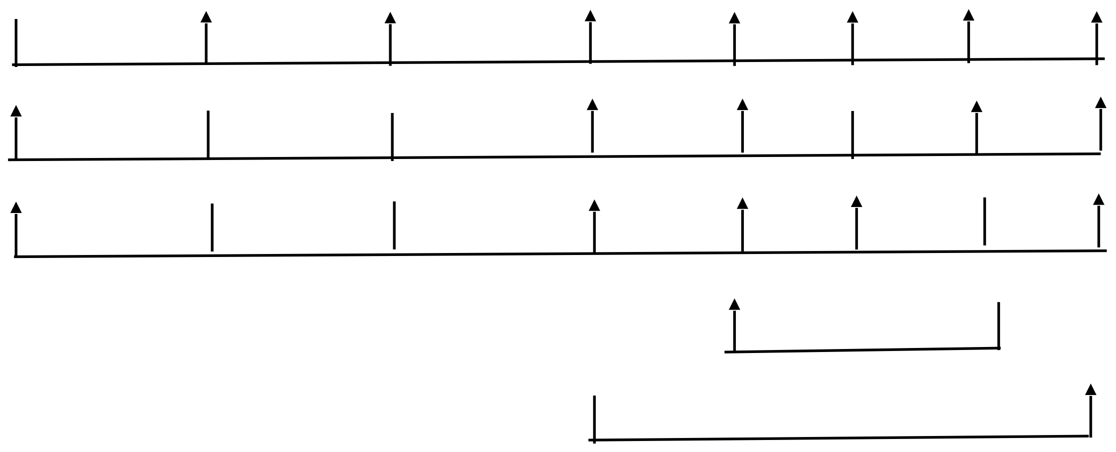
Let us consider the appointment table to verify Second Normal Form(2NF)

TABLE: appointment (appt_id, appt_date, appt_time, loc, h_id, est_fee, u_id, doc_id)

FUNCTIONAL DEPENDENCIES:

The functional dependencies are: -

I	D	T	H	E	U	C	L
appt_id	appt_date	appt_time	h_id	est_fee	u_id	doc_id	loc



So, the functional dependencies of table are: -

FD1: I -> D,T,H,E,U,C,L

FD2: DTU -> I,H,E,C,L

FD3: DTC -> I,H,E,U,L

FD4: C -> E

FD5: H -> L

ATTRIBUTE CLOSURE:

1. Consider attribute I whose attribute closure is represented by I+
   Initially I+ = {I}
   From I -> D,T,H,E,U,C,L, V+ = {I,D,T,H,E,U,C,L }
2. Consider attribute D whose attribute closure is represented by D+
   Initially D+ = {D}
   Since there are no FDs of the form D -> y
   D+ = {D}
3. Consider attribute T whose attribute closure is represented by T+
   Initially T+ = {T}
   Since there are no FDs of the form T -> y
   T+ = {T}
4. Consider attribute H whose attribute closure is represented by H+
   Initially H+ = {H}
   From H -> L, H+ = {H, L}
5. Consider attribute E whose attribute closure is represented by E+
   Initially E+ = {E}
   Since there are no FDs of the form E -> y
   E+ = {E}
6. Consider attribute U whose attribute closure is represented by U+
   Initially U+ = {U}
   Since there are no FDs of the form U -> y
   U+ = {U}
7. Consider attribute C whose attribute closure is represented by C+
   Initially C+ = {C}
   From C -> E, C+ = {C, E}

CHECK FOR 2NF:

appointment table is in 1NF since all the attributes are atomic. So, checking for 2NF.

From the above functional dependencies, we see that FD1, FD2, FD3 are total functional dependencies.
Consider FD5.
We see that est_fee(E) is functionally dependant on appt_date(D), appt_time(T) and doc_id(C).
But ext_fee(E) is only dependant on doc_id(C).
We see that it is a partially dependant function.
Hence, we decompose the table appointment as
appointment1 – appt_id, appt_date, appt_time, h_id, u_id, doc_id, loc
appointment2 – doc_id, est_fee
Therefore, tables are now 2NF.

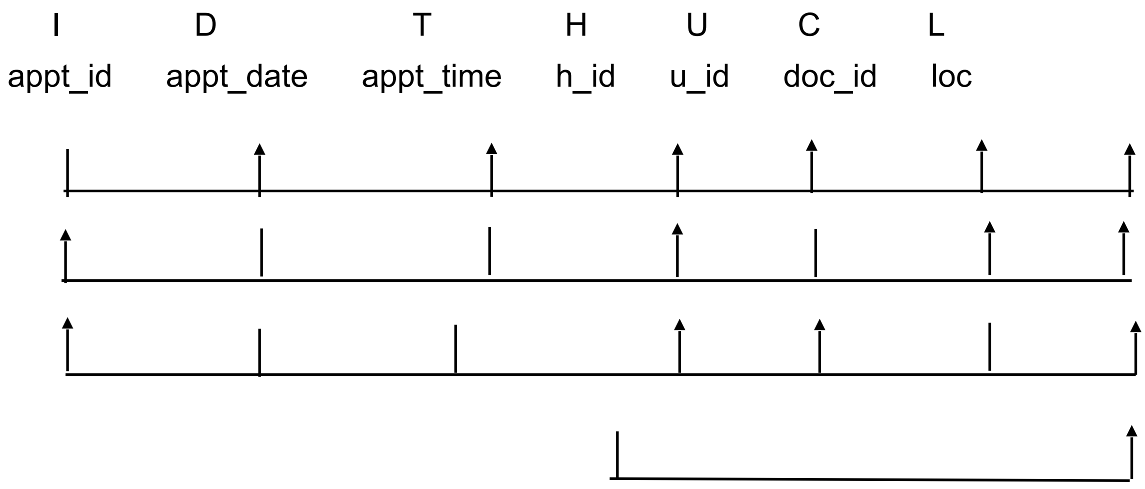
THIRD NORMAL FORM:

Let us consider the appointment1 table to verify Third Normal Form(3NF)

TABLE: appoinment1 (appt_id, appt_date, appt_time, h_id, u_id, doc_id, loc)

FUNCTIONAL DEPENDENCIES:

The functional dependencies are: -



So, the functional dependencies of table are: -
FD1: I -> D,T,H,U,C,L
FD2: DTU -> I,H,C,L
FD3: DTC -> I,H,U,L
FD4: H -> L

ATTRIBUTE CLOSURE:

1. Consider attribute I whose attribute closure is represented by I+
   Initially I+ = {I}
   From I -> D,T,H,U,C,L, I+ = {I, D,T,H,U,C,L }
2. Consider attribute D whose attribute closure is represented by D+
   Initially D+ = {D}
   Since there are no FDs of the form D -> y
   D+ = {D}
3. Consider attribute T whose attribute closure is represented by T+
   Initially T+ = {T}
   Since there are no FDs of the form T -> y
   T+ = {T}
4. Consider attribute I whose attribute closure is represented by H+
   Initially H+ = {H}
   From H -> L, H+ = {H,L }

5. Consider attribute U whose attribute closure is represented by U+
   Initially   U+ = {U}
   Since there are no FDs of the form U -> y
   U+ = {U}
6. Consider attribute C whose attribute closure is represented by C+
   Initially   C+ = {C}
   Since there are no FDs of the form C -> y
   C+ = {C}

CHECK FOR 3NF:

Appointment1 table is in 1NF and 2NF already since all the attributes are atomic and also there are no partial functional dependancies.
So checking for 3NF.
Consider FD4-
It doesnot follow any condition of 3NF
(

1. It is a trivial FD(X -> Y)
2. X is a superkey
3. Y is part of a key

)
So we decompose-
appointment1A - appt_id, appt_date, appt_time, h_id, u_id, doc_id
appointment1B – h_id,loc