Project for Database Design

Phase IV. Documentation

Vinit Patel Madhav Patel
vnp130130@utdallas.edu mmp130630@utdallas.edu

(Week 17: Dec, 2 to Dec, 9)

0. Pre-Illumination

In this project report we will follow the requirement of Phase IV directly. In Section 1 we gave problem description copied from Web site; in Section 2 we answered 3 questions listed in the project and justified our solution; in Section 3 we exhibited EER diagram with all assumptions; in Section 4 we showed our relational schema after normalization; in Section 5 we gave all requested SQL statements for both views and queries; and in Section 6 we gave dependency diagram induced from relational schemas. Finally, a short summary is given at the end of this report.

1. Problem Description

Design, develop, and test Hospital Management System to maintain the records of various departments, staff, and patients in the hospital based on combined database. The project is in four parts: conceptual database design (Phase I), logical database design (Phase II), Oracle relational database implementation (Phase III), and final report &demo (Phase IV).

MM hospital is a full-service medical facility with numbers of staff including doctors, nurses, receptionists, pharmacists, and support staff. Patients having different kinds of ailments come to the hospital and get checkup done from the concerned doctors. They can also schedule treatment either online or by calling the receptionists. If required they are admitted as in-patients in the hospital and discharged after all services.

1) Each employee's information contains ID, Name, SSN, Gender, Age, Title, Department(s), Date of Join, Salary, Address (address line 1, address line 2, city, state, zipcode), Phone_Number (one individual may have more than one phone number) and Email.

Doctors write prescriptions (including RX#, Description for drug uses) for all patients. For each doctor, the system records Level (1-5), Medical School, Language(s), and Specialty (Cardiology, Family Practice, Nephrology, Infertility, etc.). One doctor may know multiple languages and have multiple specialties. The language that each nurse speaks is also
recorded respectively. Doctors and nurses are responsible for all patients' surgery and test treatments (see details in (4)).

Receptionists are responsible for calling treatment appointments for all patients.

Pharmacists are responsible for dispensing prescribed medications to patients. The system records the Degree for each pharmacist, and tracks the Dispense_Date for each filling service. Database Design 2 Support staff in the hospital are divided into part-time staff and full-time staff. For part-time staff, the number of work hours per week is recorded.
One support staff can also be a volunteer. For all volunteers, their available weekday and time-slots should be recorded. For safety, the age of each volunteer cannot be over 75.

2) Information describing the department of the hospital is recorded:
Department ID, Department Name, Locations, and Phone number. Each department has at least one manager who works in this department.

3) Patients of the hospital can be divided into in-patients and out-patients (or both). For each patient, the system keeps track of the following information:
ID (like "mxl000001", unique), Last Name, First Name, DOB, Gender, Chief Reason for Visit, Name of driver to transport home, Telephone Number(s), Address (address line 1, address line 2, city, state, zipcode), Email, Language(s), and Medical History. Particularly, SSNs are required for all in-patients.

The system will assign a unique ID to each patient generated by picking out the first letter of the first name and the last name with a randomly generated letter in the middle, then, putting a randomly generated integer with six digits at the end. For example, for the person named Mary Lee, the id can be "mxl000001", where x and 000001 are randomly generated.

Receptionists serve all patients then arrange the corresponding doctors or nurses for the patients in need. All treatments must be given by either doctors, nurses, or both. The system records the fee and the result for each treatment. A Patient may visit the hospital many times for different reasons, and the system tracks the corresponding receptionist and doctor
(or nurse, if any), the date and the time for each visit. The system also keeps track of the insurance information (Policy#, Company Name, Phone, and Expired Data) for all insured patients. It is assumed that for different companies the policy# may be the same, and one particular insurance record may cover different patients. Each patient can be insured by maximum 3 insurance companies.

4) For all patients, the hospital offers treatment scheduling either online or by calling the receptionists. For each calling appointment, the system keeps track of the receptionist information along with the calling date as Database Design 3 well. For each appointment, Name of ordering doctors (if known),
Treatment_type (can only be "surgery" or "test"), Scheduled_Date, and Scheduled_Time are recorded. The patient in need, the Scheduled_Date, and Scheduled_Time together determine one unique appointment.

5) Ambulances service is provided for all patients. Each ambulance has a unique License Number, a Stored-In Date, and a Location. A DL# of an ambulance driver who belongs to support staff is recorded. For each detachment, the system keeps track of Start-time, End-time, along with the carried patient (s) (no more than 5).

6) Medicines stored in the hospital can be tracked in the system also. Each kind of medicine's information contains ID, Name, Price, Stored-In Date, Quantity, and Type (Rx or Non-prescription). Pharmacists fill all prescriptions for the patients, and the corresponding dates are recorded.

7) Each in-patient is assigned one particular bed during unique time period. Each Bed has an ID_no, a Room_no (not unique). The value of ID_no is between “001” and “500”. The specified start time and end time for each bed assignment can be tracked.

8) After his/her visit, a patient may receive bills from several providers, such as prescription, treatment, ambulances, ward (Bed), and pharmacy.Besides the patient information in (3), the hospital also needs the following financial information for all billed patients: Account Number, Account Holder, Bank Name, Billing-Address, and Expired Date. The
system tracks the status (Paid, Unpaid) for all billing information.

9) The hospital often holds special events of different themes for its patients, like cancer support, childbirth class etc. Each event has a unique Name, Held time, and a Description. The event holders can be employees or volunteers. An event may be held in different cities. Thus, the system needs to record the holders, city (or cities) for each event. All patients
including their relatives can attend any event they are interested in, while a relative is an individual attendee. And the attendees need evaluate the events they attend. The evaluation score varies from 0 to 100. The system only records the Names of all relatives. For different patients their relatives’ names may be the same.

2. Three Questions

2.1 1.Is the ability to model super-class / subclass relationships likely to be important in such environment? Why or why not?

Answer:

Yes, the ability to model super-class/ sub class relationships is very important in this kind of environment. In the Hospital Management Database, using Super-class/ sub-class relationship, we can easily handle the inheritance problems that may cause without super class/sub class.
An advantage of using the subclass-table strategy for abstract base classes is that properties common to multiple persistent subclasses can be defined in the superclass without having to suffer the performance consequences and relational design restrictions inherent in other strategies. Persisting and modifying instances of subclasses is efficient, typically only requiring a single INSERT or UPDATE statement. Loading relations to these subclasses is also efficient. For example, we do not need to mention all the attributes of doctors and drivers which are common with an entity Employee in this environment. Hence, super class/ sub class also improves the redundancy criteria. Thus, super class/ subclass relationships play a vital role in creation of database management systems.

2.2 2.Can you think of 5 more rules (other than the one explicitly described above) that are likely to be used in a hospital environment? Add your rules to the above requirement to be implemented.

Answer:

1) ID of medicines named ‘m_id’ starts with m followed by indefinite unique integers. For example ‘m01’,’m122’ etc.
2) Ambulance should be identified uniquely from AMB_Reg_Licence number starting with AMB followed by three digit integer. For example ‘AMB020’.
3) For in_patient table, it is possible that patient may have visited hospital for TEST and need not be allocated a bed, so bed should be allowed NULL values.
4) The department id start with first two characters of department name followed by 3 digit integer. Example.’Ca123’
5) While taking appointment on call, time should be restricted between 10:00 to 22:00.

2.3 Justify using a Relational DBMS like Oracle for this project.

There are several advantages of using RDBMS for which we have used ORACLE RDBMS:
A Relational Database Management System (RDBMS) is a software system that provides access to a relational database. The software system is a collection of software applications that can be used to create, maintain, manage and use the database. A "relational database" is a database structured on the "relational" model. Data are stored and presented in a tabular format, organized in rows and columns with one record per row.

Data Structure

- The table format is simple and easy for database users to understand and use. RDBMSs provide data access using a natural structure and organization of the data. Database queries can search any column for matching entries.
  Multi-User Access
- RDBMSs allow multiple database users to access a database simultaneously. Built-in locking and transactions management functionality allow users to access data as it is being changed, prevents collisions between two users updating the data, and keeps users from accessing partially updated records.

Privileges

- Authorization and privilege control features in an RDBMS allow the database administrator to restrict access to authorized users, and grant privileges to individual users based on the types of database tasks they need to perform. Authorization can be defined based on the remote client IP address in combination with user authorization, restricting access to specific external computer systems.

Network Access

- RDBMSs provide access to the database through a server daemon, a specialized software program that listens for requests on a network, and allows database clients to connect to and use the database. Users do not need to be able to log in to the physical computer system to use the database, providing convenience for the users and a layer of security for the database. Network access allows developers to build desktop tools and Web applications to interact with databases.

Speed

- The relational database model is not the fastest data structure. RDBMS advantages, such as simplicity, make the slower speed a fair trade-off. Optimizations built into an RDBMS, and the design of the databases, enhance performance, allowing RDBMSs to perform more than fast enough for most applications and data sets. Improvements in technology, increasing processor speeds and decreasing memory and storage costs allow systems administrators to build incredibly fast systems that can overcome any database performance shortcomings.

Maintenance

- RDBMSs feature maintenance utilities that provide database administrators with tools to easily maintain, test, repair and back up the databases housed in the system. Many of the functions can be automated using built-in automation in the RDBMS, or automation tools available on the operating system.

Language

- RDBMSs support a generic language called "Structured Query Language" (SQL). The SQL syntax is simple, and the language uses standard English language keywords and phrasing, making it fairly intuitive and easy to learn. Many RDBMSs add non-SQL, database-specific keywords, functions and features to the SQL language.

We have used ORACLE database because of the advantages like:

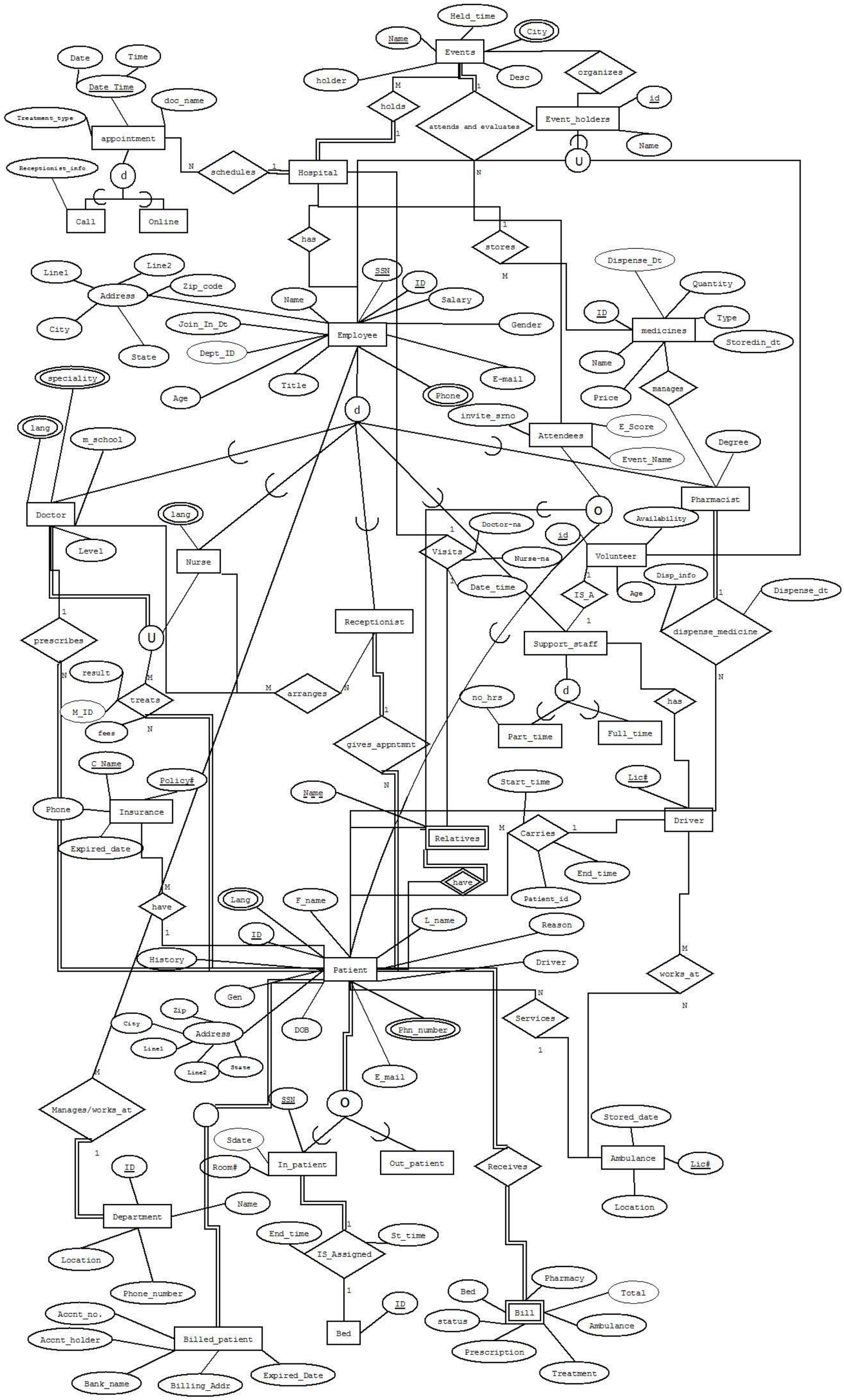
- Data Security
- Efficient Application Development
- Better Data Designing and Mining
- Simplified Data administration.

Detail description of which is provided by RDBMS advantages as discussed above.

3. EER diagram with all assumptions

- **Assumptions:**

- We think that all the patient contacts hospital directly with phone numbers they have and receptionist accepts their request.
- One doctor prescribes multiple patients. So we have cardinality as 1:N for relationship between doctor and patient.
- We assume that the doctor and nurses responsible for patient is same as they treats the patient.
- We have assumed that one receptionist makes appointment for multiple patients by calling and also that the receptionists have all the data of patients.
- We have assumed that age of each volunteer is less than 75.
- We assume that the system has already implemented algorithm to generate unique patient id.
- For cardinality of receptionist and patient, we assume that one receptionist communicates with multiple patients.
- We have assumed that appointment is scheduled just by call or online-website. But it is also possible that appointment is taken in person at hospital.
- Pharmacist has all information to write prescriptions.
- We have assumed that patient informs the relatives about the events organized by hospital and then even the individual relatives can attend the event.
- We assume notation M and N for showing multiple (Many) cardinality.
- All the transactions like bill payment, currency is in US Dollar.

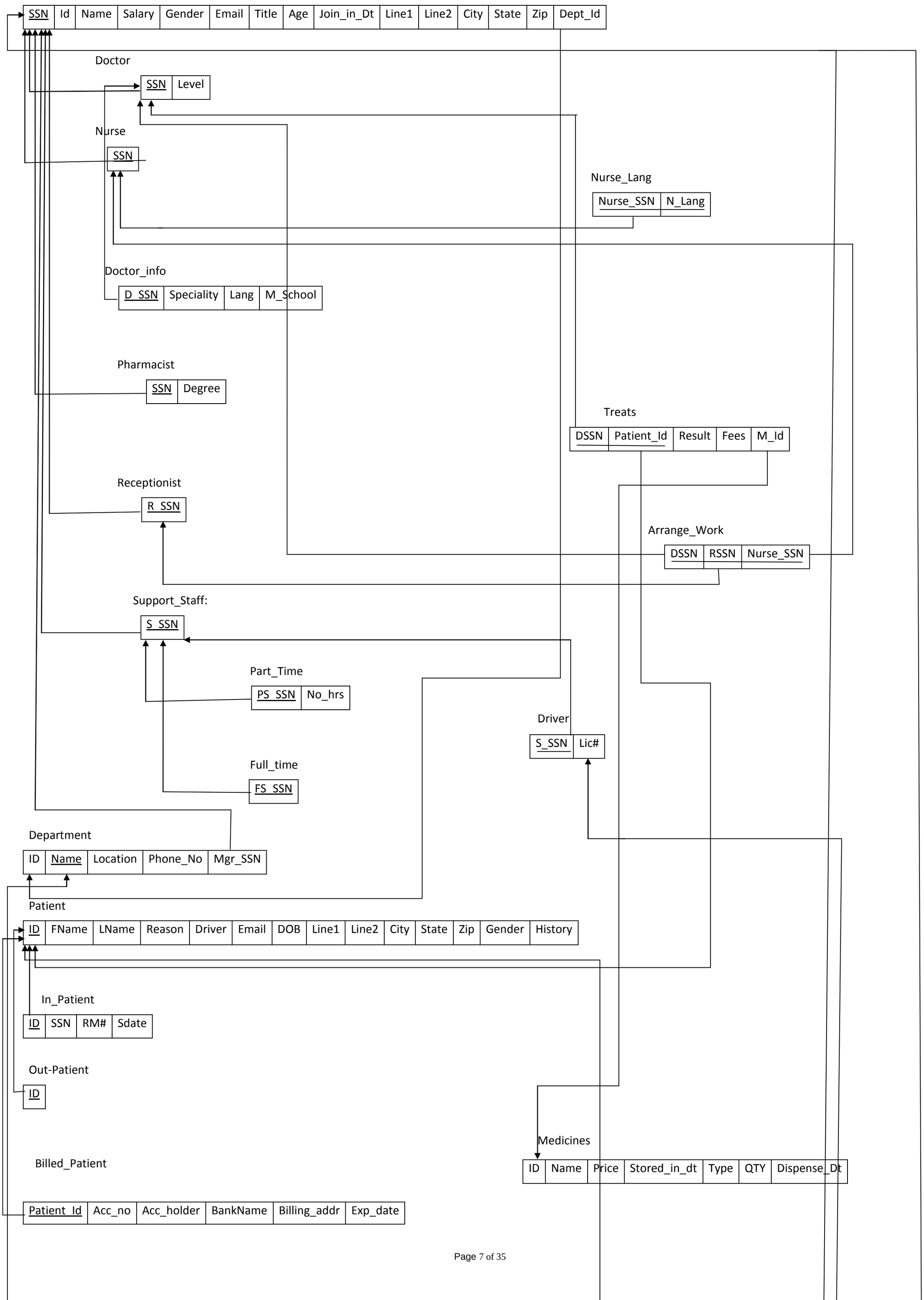

Events
Held_time
City
Name
Desc
holder
organizes
holds
attends and evaluates
Event_holders
id
Name
Date
Time
Date_Time
doc_name
Treatment_type
appointment
Receptionist_info
d
Call
Online
schedules
Hospital
U
has
stores
Dispense_Dt
Quantity
Line1
Line2
Address
Zip_code
City
State
SSN
ID
Salary
Name
Join_In_Dt
Employee
Gender
Dept_ID
Age
Title
E-mail
Phone
invite_srno
Attendees
E_Score
Event_Name
ID
medicines
Type
Storedin_dt
Name
Price
manages
speciality
lang
m_school
Degree
Pharmacist
Doctor
lang
Level
Nurse
Doctor-na
Nurse-na
Visits
Date_time
O
id
Volunteer
Availability
Disp_info
Age
IS_A
Dispense_dt
dispense_medicine
Support_staff
Receptionist
prescribes
U
result
M_ID
treats
fees
arranges
gives_appntmnt
no_hrs
d
Part_time
Full_time
has
C_Name
Policy#
Insurance
Phone
Expired_date
Name
Start_time
Lic#
Driver
Relatives
Carries
End_time
have
Patient_id
have
Lang
F_name
L_name
Reason
ID
History
Patient
Driver
works_at
Gen
Zip
City
Address
Line1
Line2
State
DOB
Phn_number
Services
E_mail
Manages/works_at
SSN
O
Stored_date
Sdate
Room#
In_patient
Out_patient
Receives
Ambulance
Lic#
ID
Department
Name
Location
Phone_number
End_time
IS_Assigned
St_time
Pharmacy
Bed
Total
ID
Bed
status
Bill
Ambulance
Accnt_no.
Accnt_holder
Billed_patient
Prescription
Treatment
Bank_name
Billing_Addr
Expired_Date

4. Relational Schema in Third Normal Form

4.1 Relational Schema

See next page for diagram:

8A.Employee:


SSN | Id | Name | Salary | Gender | Email | Title | Age | Join_in_Dt | Line1 | Line2 | City | State | Zip | Dept_Id
Doctor
SSN | Level
Nurse
SSN
Nurse_Lang
Nurse_SSN | N_Lang
Doctor_info
D_SSN | Speciality | Lang | M_School
Pharmacist
SSN | Degree
Treats
DSSN | Patient_Id | Result | Fees | M_Id
Receptionist
R_SSN
Arrange_Work
DSSN | RSSN | Nurse_SSN
Support_Staff:
S_SSN
Part_Time
PS_SSN | No_hrs
Driver
S_SSN | Lic#
Full_time
FS_SSN
Department
ID | Name | Location | Phone_No | Mgr_SSN
Patient
ID | FName | LName | Reason | Driver | Email | DOB | Line1 | Line2 | City | State | Zip | Gender | History
In_Patient
ID | SSN | RM# | Sdate
Out-Patient
ID
Medicines
ID | Name | Price | Stored_in_dt | Type | QTY | Dispense_Dt
Billed_Patient
Patient_Id | Acc_no | Acc_holder | BankName | Billing_addr | Exp_date

Events

Name	Held_Time	Description

8C.Event_hol_Emp

Name	E_SSN

Event_hold_Vol

Name	V_Id

Volunteer

ID	Age	Availability

Attendees

Invite_sr_no	PID	Relative_Name	E_Score	Event_Name

Appointment

Date	Time	PID

Medicines

ID	Name	Price	Stored_in_dt	Type	Qty

Works_at

D_LIC#	A_LIC#

Insurance

Cpmpany_Name	Policy#	PID	Phone	Exp_dt

Ambulance

Lic#	Stored_dt	Location

Call

Date	Time	PID	Treatment_type	Doc_name	Receptionist_info

Online

Date	Time	PID	Treatment_type	Doc_Name

Relatives

P_ID	Rel_Name

Bill

B_No	P_ID	Status	Bed	Prescription	Ambulance	Pharmacy	Total

Event_City

Event_Name	E_City

E_DepName

ESSN	Dept_Name

PATIENT_PH_NO

P_ID	PH_NO

4.2 Format for Every Relation

Table 3. Format for Each Attribute

Relation Names	Attributes	Date Type
Employee	E_SSN	xxxxxxxxx Integer=10
	ID	String <= 11 chars
	Name	String <= 20 chars
	Salary	Long integer
	Gender	String
	Email	String<=20 chars
	Title	String
	Age	Integer<=100
	Join_in_Date	Date
	Line	String<=20 chars
	Line2	string <= 20 chars
	Zip	String <= 10 chars
	City	String<=20 chars
	State	String<=20 chars
	Dept_Id	String<=11 chars

Relation Names	Attributes	Data Type
Department	ID	String<=11 chars
	Name	String<=20 chars
	Location	String<=20 chars
	Phone_no	xxxxxxxxxx integer=10 chars

Relation Names	Attributes	Data Type
Patient	ID	String<=11 chars
	FName	String<=20 chars
	LName	String<=20 chars
	Phone_no	xxxxxxxxxx integer=10 chars
	Reason	String<=30 chars
	Driver	String<=20 chars
	Email	String<=20 chars
	DOB	Date
	Line	String<=20 chars
	Line2	String<=20 chars
	City	String<=20 chars
	State	String<=20 chars
	Zip	String<=5 chars
	Gender	String
	History	String<=25 chars
	Ins_Policy	String<=10 chars
	D_SSN	xxxxxxxxx Integer=10
	A_LIC#	Integer<=10

Relation Names	Attributes	Data Type
Events	Name	String<=20 chars
	Holder	String<=20 chars
	Held_time	Date
	Description	String<=30 chars

Relation Names	Attributes	Data Type
Appointment	Date	Date
	Time	Date
	Doc_Name	String<=20 chars
	Treat_type	String<=20 chars

Relation Names	Attributes	Data Type
Medicine	ID	Integer<=10
	Name	String<=20 chars
	Price	Integer
	Stored_in_Date	Date
	Type	String<=20 chars
	Qty	Integer
	Dispense_Dt	Date

Relation Names	Attributes	Data Type
Insurance	Comp_Name	String<=20 chars
	Policy#	Integer<=10
	Phone#	XXXXXXXXX integer=10 chars
	Exp_Date	Date

Relation Names	Attributes	Data Type
Ambulance	Lic#	Integer<=10
	Stored_Date	Date
	Location	String<=20 chars

Relation Names	Attributes	Data Type
Volunteer	ID	String<=11 chars
	Age	Integer
	Availability	Boolean

Relation Names	Attributes	Data Type
Relatives	P_ID	Integer<=10
	Rel_Name	String<=20 chars

Relation Names	Attributes	Data Type
Bill	Bill#	Long Integer<=10
	P_SSN	xxxxxxxxx String=10 chars
	Bed	Long Integer<=10
	Status	Boolean
	Prescription	String<=30 chars
	Ambulance	Long Integer<=10
	Pharmacy	Long Integer<=10
	Total	Integer

Relation Names	Attributes	Data Type
Doctor	D_SSN	xxxxxxxxx Integer=10
	Lang	String<=10 chars
	M_School	String<=20 chars
	Level	String<=5 chars

Relation Names	Attributes	Data Type
Nurse	N_SSN	xxxxxxxxx Integer=10
	Lang	String<=10 chars

Relation Names	Attributes	Date Type
Manager	Mgr_SSN	XXXXXXXXX, String = 10 chars
	ID	String <= 11 chars
	Name	String <= 20 chars

Relation Names	Attributes	Date Type
Pharmacist	PH_SSN	XXXXXXXXX, String = 10 chars
	ID	String <= 11 chars
	Name	String <= 20 chars

Relation Names	Attributes	Date Type
Receptionist	R_SSN	XXXXXXXXX, String = 10 chars
	ID	String <= 11 chars
	Name	String <= 20 chars

Relation Names	Attributes	Date Type
Support Staff	SS_SSN	XXXXXXXXX, String = 10 chars
	ID	String <= 11 chars
	Name	String <= 20 chars

Relation Names	Attributes	Date Type
Part-Time	PS_SSN	XXXXXXXXX, String = 10 chars
	ID	String <= 11 chars
	Name	String <= 20 chars

Relation Names	Attributes	Date Type
Full-TIme	FS_SSN	XXXXXXXXX, String = 10 chars
	ID	String <= 11 chars
	Name	String <= 20 chars

Relation Names	Attributes	Data Type
In_Patient	P_ID	String<=11 chars
	RM#	String<=20 chars
	IP_SSN	xxxxxxxxx Integer=10

Relation Names	Attributes	Data Type
Out_Patient	P_ID	String<=11 chars
	FName	String<=20 chars
	LName	String<=20 chars

Relation Names	Attributes	Data Type
Call	Date	Date
	Time	Date
	P_ID	String<=11 chars
	Treatment_Type	String<=20 chars
	Doc_Name	String<=20 chars
	Receptionist_info	String<=20 chars

Relation Names	Attributes	Data Type
Online	Date	Date
	Time	Date
	P_ID	String<=11 chars
	Treatment_Type	String<=20 chars
	Doc_Name	String<=20 chars

Relation Names	Attributes	Data Type
Event Holders	ESSN	xxxxxxxxx Integer=10
	Vol_ID	String

Relation Names	Attributes	Data Type
Attendees	Invite_sr_no	String<=11 chars
	P_ID	String<=20 chars
	Relative_Name	String<=20 chars
	Event_Name	String<=20 chars

Relation Names	Attributes	Data Type
Billed Patient	Patient_ID	String
	Acc#	Long Integer<=10
	Acc_Holder	String<=20 chars
	Bank_Name	String<=20 chars
	Billing_Addr	String<=30 chars
	Exp_Date	Date

Relation Names	Attributes	Data Type
Treats	DSSN	xxxxxxxxx Integer=10
	Patient_Id	string<=11 chars
	Result	String<=50 chars
	Fees	String<=20 chars
	M_Id	Integer

Relation Names	Attributes	Data Type
Arrange_work	DSSN	xxxxxxxxx Integer=10
	RSSN	xxxxxxxxx Integer=10
	Nurse_SSN	xxxxxxxxx Integer=10

Relation Names	Attributes	Data Type
Works_At	D_LIC#	String<10chars
	A_LIC#	string<=10 chars

Relation Names	Attributes	Data Type
Driver	S_SSN	xxxxxxxxx Integer=10
	LIC#	string<=10 chars

5. All Requested SQL Statements

5.1 Creation of Database with SQL Statements

5.1.1 Table Creation

- **EMPLOYEE**

```
CREATE TABLE EMPLOYEE
(SSN        VARCHAR(10)     NOT NULL,
ID          VARCHAR(11)     NOT NULL,
NAME        VARCHAR(15)     NOT NULL,
SALARY      INT             NOT NULL ,
GENDER      VARCHAR(8),
EMAIL       VARCHAR(20),
TITLE       VARCHAR(3),
JOIN_IN_DATE        DATE        ,
LINE        VARCHAR (20),
LINE2       VARCHAR (20),
CITY        VARCHAR (20),
STATE       VARCHAR (20),
ZIP         VARCHAR (10),
Dept_ID     VARCHAR (11),
PRIMARY KEY(SSN)
FOREIGN KEY(Dept_ID) REFERENCES DEPT(ID));
```

- **DOCTOR**

```
CREATE TABLE DOCTOR
(SSN        VARCHAR(10)    NOT NULL,
D_L         INT,
PRIMARY KEY(SSN),
FOREIGN KEY(SSN) REFERENCES EMPLOYEE(SSN)
CHECK(D_L>0 AND D_L<5));
```

- **NURSE_LANG**

```
CREATE TABLE NURSE_LANG
(N_SSN      VARCHAR(10)    NOT NULL,
N_LANG      VARCHAR(20),
PRIMARY KEY(N_SSN),
FOREIGN KEY(N_SSN) REFERENCES NURSE(SSN));
```

- **NURSE**

```
CREATE TABLE NURSE
(SSN        VARCHAR(10)        NOT NULL,
PRIMARY KEY(SSN),
FOREIGN KEY(SSN) REFERENCES EMPLOYEE(SSN));
```

- **DOCTOR_INFO**

```
CREATE TABLE DOCTOR_INFO
(D_SSN          VARCHAR(10)    NOT NULL,
SPECIALITY      VARCHAR(10)    NOT NULL,
LANG            VARCHAR(10)    NOT NULL,
M_SCHOOL        VARCHAR(15)    NOT NULL,
PRIMARY KEY(D_SSN),
FOREIGN KEY(D_SSN) REFERENCES DOCTOR(SSN));
```

- **PHARMACIST**

```
CREATE TABLE PHARMACIST
(P_SSN          VARCHAR(10)    NOT NULL,
DEGREE          VARCHAR(10)    NOT NULL,
PRIMARY KEY(P_SSN),
FOREIGN KEY(P_SSN) REFERENCES EMPLOYEE(SSN));
```

- **TREATS**

```
CREATE TABLE TREATS
(D_SSN          VARCHAR (10)   NOT NULL,
P_ID            VARCHAR (11)   NOT NULL,
RESULT          VARCHAR (10),
FEES            VARCHAR (10),
M_ID            VARCHAR (11),
PRIMARY KEY (D_SSN),
FOREIGN KEY (D_SSN) REFERENCES DOCTOR (SSN),
FOREIGN KEY (M_ID) REFERENCES MEDICINE (ID),
FOREIGN KEY(P_ID) REFERENCES PATIENT(P_ID));
```

- **RECEPTIONIST**

```
CREATE TABLE RECEPTIONIST
(R_SSN        VARCHAR(10)    NOT NULL,
PRIMARY KEY(R_SSN),
FOREIGN KEY(R_SSN) REFERENCES EMPLOYEE(SSN));
```

- **ARRANGE_WORK**

```
CREATE TABLE ARRANGE_WORK
(D_SSN        VARCHAR(10)    NOT NULL,
R_SSN         VARCHAR(10)    NOT NULL,
N_SSN         VARCHAR(10)    NOT NULL,
PRIMARY KEY (D_SSN,R_SSN,N_SSN),
FOREIGN KEY(D_SSN)   REFERENCES DOCTOR(SSN),
FOREIGN KEY(R_SSN) REFERNCES RECEPTIONIST(R_SSN),
FOREIGN KEY(N_SSN) REFERENCES NURSE(SSN));
```

- **SUPPORT_STAFF**

```
CREATE TABLE SUPPORT_STAFF
(S_SSN            VARCHAR(10)    NOT NULL,
PRIMARY KEY(S_SSN),
FOREIGN KEY (S_SSN) REFERENCES EMPLOYEE(SSN));
```

- **PART_TIME**

```
CREATE   TABLE PART_TIME
(PS_SSN           VARCHAR(10)    NOT NULL,
NO_HOURS          VARCHAR(5),
PRIMARY KEY(PS_SSN),
FOREIGN KEY(PS_SSN) REFERENCES SUPPORT_STAFF(S_SSN));
```

- **FULL_TIME**

```
CREATE TABLE FULL_TIME
(FS_SSN           VARCHAR(10)    NOT NULL,
PRIMARY KEY(FS_SSN),
FOREIGN KEY(FS_SSN) REFERENCES SUPPORT_STAFF(S_SSN));
```

- **DRIVER**

```
CREATE TABLE DRIVER
(S_SSN            VARCHAR(10)    NOT NULL,
LIC#              VARCHAR(10)    NOT NULL,
PRIMARY KEY(S_SSN),
FOREIGN KEY(S_SSN) REFERENCES SUPPORT_STAFF(S_SSN),
FOREIGN KEY(LIC#) REFERENCES WORKS_AT(D_LIC#));
```

- **DEPT**

```
CREATE TABLE DEPT
(ID          VARCHAR(10)     NOT NULL,
NAME         VARCHAR(15)     NOT NULL,
LOC          VARCHAR(20)     NOT NULL,
PHONE_NO     VARCHAR(10),
MGR_SSN      VARCHAR(10)     NOT NULL,
PRIMARY KEY(ID),
FOREIGN KEY (MGR_SSN) REFERENCES EMPLOYEE(SSN));
```

- **IN_PATIENT**

```
CREATE TABLE IN_PATIENT
(P_ID        VARCHAR(10)     NOT NULL,
SSN          VARCHAR(10)     NOT NULL,
RM#          VARCHAR(5),
SDATE        DATE
PRIMARY KEY(P_ID),
FOREIGN KEY(P_ID) REFERENCES PATIENT(P_ID));
```

- **PATIENT**

```
CREATE TABLE PATIENT
(P_ID        VARCHAR(11)     NOT NULL,
FNAME        VARCHAR(10)     NOT NULL,
LNAME        VARCHAR(10),
REASON       VARCHAR(25),
DRIVER       VARCHAR(10),
EMAIL_ID     VARCHAR(20),
DOB          DATE,
LINE         VARCHAR(20),
LINE2        VARCHAR(20),
CITY         VARCHAR(15),
STATE        VARCHAR(15),
GENDER       VARCHAR(5),
HISTORY      VARCHAR(25),
ZIP          VARCHAR(10),
PRIMARY KEY(P_ID));
```

- **OUT_PATIENT**

```
CREATE TABLE OUT_PATIENT
(P_ID        VARCHAR(10)     NOT NULL,
PRIMARY KEY(P_ID),
FOREIGN KEY(P_ID) REFERENCES PATIENT(P_ID));
```

- **BILLED_PATIENT**

```
CREATE TABLE BILLED_PATIENT
(P_ID        VARCHAR(11)     NOT NULL,
```

```
ACC_NO        VARCHAR(10),
ACC_HOLDER    VARCHAR(15),
BANK_NAME     VARCHAR(20),
BILLINH_ADDR  VARCHAR(25),
EXP_DATE      DATE,
PRIMARY KEY(PID),
FOREIGN KEY(P_ID) REFERENCES PATIENT(P_ID));
```

- EVENTS

```
CREATE TABLE EVENTS
(NAME         VARCHAR(10)   NOT NULL,
HELD_TIME     DATE          NOT NULL,
DESCRIPTION   VARCHAR(40) ,
PRIMARY KEY(NAME));
```

- EVENT_HOLD_EMP

```
CREATE TABLE EVENT_HOLD_EMP
(NAME         VARCHAR (10)  NOT NULL,
E_SSN         VARCHAR (10)  NOT NULL,
PRIMARY KEY(NAME),
FOREIGN KEY(NAME) REFERENCES EVENTS(NAME));
```

- EVENT_HOLD_VOL

```
CREATE TABLE EVENT_HOLD_VOL
(NAME         VARCHAR(10)   NOT NULL,
V_ID          VARCHAR(11)   NOT NULL,
PRIMARY KEY(NAME),
FOREIGN KEY(NAME) REFERENCES EVENTS(NAME),
FOREIGN KEY(V_ID) REFERENCES VOLUNTEER(ID));
```

- VOLUNTEER

```
CREATE TABLE VOLUNTEER
(ID           VARCHAR(11)   NOT NULL,
AGE           VARCHAR(3),
AVAIL         VARCHAR(3),
PRIMARY KEY(ID),
CHECK(AGE>0 AND AGE<75));
```

- ATTENDEES

```
CREATE TABLE ATTENDEES
(INVITE_SR_NO     VARCHAR (10)   NOT NULL,
P_ID              VARCHAR (11)   NOT NULL,
RELATIVE_NAME     VARCHAR (20),
E_SCORE           VARCHAR (3)    NOT NULL,
EVENT_NAME        VARCHAR(20)
PRIMARY KEY(INVITE_SR_NO),
FOREIGN KEY (P_ID) REFERENCES PATIENT (P_ID),
```

```
FOREIGN KEY (EVENT_NAME) REFERENCES EVENTS (NAME),
CHECK (E_SCORE>0 AND E_SCORE<100));
```

- **APPOINTMENT**

```
CREATE TABLE APPOINTMENT
(DT          DATE          NOT NULL,
TIME         VARCHAR(10)   NOT NULL,
P_ID         VARCHAR(11)   NOT NULL,
PRIMARY KEY(DT,TIME),
FOREIGN KEY(P_ID) REFERENCES PATIENT(P_ID));
```

- **WORKS_AT**

```
CREATE TABLE WORKS_AT
(D_LIC#      VARCHAR(10)   NOT NULL,
A_LIC#       VARCHAR(10)   NOT NULL,
PRIMARY KEY(D_LIC#,A_LIC#),
FOREIGN KEY(D_LIC#) REFERENCES DRIVER(S_SSN),
FOREIGN KEY(A_LIC#) REFERENCES AMBULANCE(LIC_NO));
```

- **AMBULANCE**

```
CREATE TABLE AMBULANCE
(LIC_NO      VARCHAR(10)   NOT NULL,
STORED_DT    DATE,
LOC          VARCHAR(20),
PRIMARY KEY(LIC_NO));
```

- **MEDICINES**

```
CREATE TABLE MEDICINES
(ID          VARCHAR (10)  NOT NULL,
NAME         VARCHAR (15)  NOT NULL,
PRICE        VARCHAR (4)   NOT NULL,
STORED_IN_DT DATE          ,
TYPE         VARCHAR (10),
QTY          VARCHAR (4),
DISPENSE_DT  DATE
PRIMARY KEY (ID));
```

- **INSURANCE**

```
CREATE TABLE INSURANCE
(C_NAME      VARCHAR(10)   NOT NULL,
POLICY_NO    VARCHAR(10)   NOT NULL,
P_ID         VARCHAR(11)   NOT NULL,
PHONE        VARCHAR (10),
EXP_DT       DATE,
PRIMARY KEY (C_NAME, POLICY_NO),
FOREIGN KEY (P_ID) REFERENCES PATIENT (P_ID));
```

- **CALL**

```
CREATE TABLE CALL
(DT          DATE          NOT NULL,
TIME         DATE          NOT NULL,
P_ID         VARCHAR(11)   NOT NULL,
TREAT_TYPE   VARCHAR(10),
DOC_NAME     VARCHAR(20)   NOT NULL,
REC_INFO     VARCHAR(20)   NOT NULL,
PRIMARY KEY(DT,TIME),
FOREIGN KEY (DT,TIME) REFERENCES APPOINTMENT(DT,TIME),
FOREIGN KEY(P_ID) REFERENCES PATIENT(P_ID));
```

- **ONL**

```
CREATE TABLE ONL
(DT          DATE          NOT NULL,
TIME         DATE          NOT NULL,
P_ID         VARCHAR(11)   NOT NULL,
TREAT_TYPE   VARCHAR(10),
DOC_NAME     VARCHAR(20)   NOT NULL,
PRIMARY KEY(DT,TIME),
FOREIGN KEY (DT,TIME) REFERENCES APPOINTMENT(DT,TIME),
FOREIGN KEY(P_ID) REFERENCES PATIENT(P_ID));
```

- **RELATIVES**

```
CREATE TABLE RELATIVES
(P_ID        VARCHAR(11)   NOT NULL,
REL_NAME     VARCHAR(20)   NOT NULL,
PRIMARY KEY(P_ID,REL_NAME),
FOREIGN KEY(P_ID) REFERENCES PATIENT(P_ID));
```

- **BILL**

```
CREATE TABLE BILL
(B_NO        VARCHAR(10)   NOT NULL,
P_ID         VARCHAR(11)   NOT NULL,
STATUS       VARCHAR(10),
BED          VARCHAR(10),
PRESCRIPTION VARCHAR(10),
AMBULANCE    VARCHAR(10),
PHARMACY     VARCHAR(5),
TOTAL        INT,          NOT NULL,
PRIMARY KEY(B_NO),
FOREIGN KEY(P_ID) REFERENCES PATIENT(P_ID));
```

- **EVENT_CITY**

```
CREATE TABLE EVENT_CITY
(EVENT_NAME  VARCHAR(20)   NOT NULL,
EVENT_CITY   VARCHAR(20)   NOT NULL,
```

PRIMARY KEY(EVENT_NAME,EVENT_CITY),

FOREIGN KEY(EVENT_NAME) REFERENCES EVENTS(NAME));

- **E_DEPTNAME**

CREATE TABLE E_DEPTNAME

(E_SSN VARCHAR(10) NOT NULL,

DEPT_NAME VARCHAR(20) NOT NULL,

PRIMARY KEY(E_SSN,DEPT_NAME),

FOREIGN KEY(E_SSN) REFERENCES EMPLOYEE(SSN),

FOREIGN KEY(DEPT_NAME) REFERENCES DEPT(NAME));

- **PATIENT_PH_NO**

  CREATE TABLE PATIENT_PH_NO

  (P_ID VARCHAR(11) NOT NULL,

  PH_NO VARCHAR(10) NOT NULL,

  PRIMARY KEY(P_ID,PH_NO),

  FOREIGN KEY(P_ID) REFERENCES PATIENT(P_ID));

5.1.2 A Database State

- INSERTION OF TABLE EMPLOYEE

---

1. insert into Employee

values('123674356',’01’,'Rohan',30,000,'Male',‘r@gmail.com',’Doctor’,’24’,’24-May-10’,’546XYZApt,’Courts mccallum’,’Dallas’,’Texas’,’75252’, ’CA123’);

2. insert into Employee

values('123674389',’02’,'Balika',35,000,'Female',‘b@gmail.com',’Nurse’,’27’,’12-Aug-08’,’532ABCAppt’,’Glennmccallum’,’Dallas’,’Texas’,’75252’,’CA123’);

3. insert into Employee

values('123674334',’03’,‘Vinit',50,000,'Male',‘v@hotmail.com',’Receptionist’,’20’,’29-Jan-11’,’4578Meadows’,’Driveway’,’Chicago’,’Illinois’,’68133’,’CA123’);

4. insert into Employee

values('123672389',’04’,'Jem',37,000,'Male',‘j@gmail.com',’Pharmacist’,’30’,’06-Jul-05’,’4135DownwardsAppt’,’Dentour’,’Sanantanio’,’California’,’43167’, CA123);

5. insert into Employee

values('489957634',’05’,'Maddy',25,000,'Male',‘m@gmail.com',’Driver’,’52’,’14-Jul-10’,’7777McCallumBlvd’,’Ashwood’,’Detroit’,’Michigan’,’48188’,’CA123’);

6. insert into Employee

values('378959208',’06’,'Ashtha',21,000,'Female',‘a@yahoo.com',’Driver’,’24’,’29-Sep-11’,’4315SherwoodCIR’,’Akshar’,’LosAngelos’,’California’,’43986’,’CA 123’);

```
SQL> select * from employee;

SSN        ID         NAME           SALARY   GENDER   EMAIL                TITLE                AGE JOIN_IN_D LINE                    LINE2                CITY
---------- ---------- -------------- -------- -------- -------------------- -------------------- --- -------- -------------------- -------------------- --------
STATE                ZIP        DEPT_ID
-------------------- ---------- ----------
123674356  01          Rohan          30000    Male     r@gmail.com          Dr.                   24 24-MAY-10 546xyz Apt      Courts of McCallum   Dallas
TX                   75252      ca123

123674334  03          Vinit          50000    Male     v@hotmail.com        Receptionist          20 29-JAN-11 4578 Meadows            Drive Way            Chicago
Illionis             68133      ca123

123674389  02          Balika         35000    Female   b@gmail.com          Nurse                 27 12-AUG-08 532abc Apt      Glen McCallum        Dallas
TX                   75252      ca123

123672389  04          Jem            37000    Male     j@gmail.com          Pharmacist            30 06-JUL-05 4135 Downwards Apt   Dentour              San Anta
California           43167      ca123

489957634  05          Maddy          25000    Male     m@gmail.com          Driver                52 14-JUL-10 7777 McCallum Blvd., Ashwood              Detroit
Michigan             48188      ca123

378959208  06          Astha          21000    Female   a@yahoo.com          driver                24 29-SEP-11 4315 Sherwood Cir    Akshar               Los Ange
California           43986      ca123

6 rows selected.

SQL>
```

- INSERTION OF TABLE Doctor

---

1. insert into Doctor
Values ('123674356','3');

```
SQL> insert into doctor values('123674356','3');

1 row created.

SQL> select * from doctor;

SSN              D_L
---------- ----------
123674356           3

SQL>
```

- INSERTION OF TABLE Nurse_Lang

---

1. insert into Nurse_Lang
Values ('123674389','English');

```
SQL> insert into nurse_lang values('123674389','English');

1 row created.

SQL> select * from nurse_lang;

N_SSN      N_LANG
---------- --------------------
123674389  English

SQL>
```

- INSERTION OF TABLE Nurse

---

1. insert into Nurse
Values('123674389');

```
SQL> insert into nurse values('123674389');

1 row created.

SQL> select * from nurse;

SSN
--------------------
123674389

SQL>
```

- INSERTION OF TABLE Doctor_info

---

1. insert into Doctor_info
Values ('123674356','Cardiologist','English','UT Dallas');

```
SQL> insert into doctor_info values('123674356','Cardiolgst','English','UTDallas');

1 row created.

SQL> select * from doctor_info;

D_SSN      SPECIALITY LANG       M_SCHOOL
---------- ---------- ---------- ---------------
123674356  Cardiolgst English    UTDallas

SQL>
```

- INSERTION OF TABLE Pharmacist

---

1. insert into Pharmacist
   Values ('123672389','Pharm.D.');

```
SQL> insert into pharmacist values('123672389','Pharm.D.');

1 row created.

SQL> select * from pharmacist;

P_SSN      DEGREE
---------- ----------
123672389  Pharm.D.

SQL>
```

- INSERTION OF TABLE Treats

---

1. insert into Treats
   Values ('123674356','100','Positive','1200','**M01**');

```
SQL> select * from treats;

D_SSN      P_ID       RESULT     FEES       M_ID
---------- ---------- ---------- ---------- ----------
123674356  100        Positive   1200       m01

SQL>
```

- INSERTION OF TABLE Receptionist

---

1. insert into Receptionist
   Values ('123674334');

```
SQL> insert into receptionist values('123674334');

1 row created.

SQL> select * from receptionist;

R_SSN
----------
123674334

SQL>
```

- INSERTION OF TABLE Arrange_work

---

1. insert into Arrange_work
   Values ('123674356', '123674334','123674389');

```
SQL> insert into arrange_work values('123674356','123674334','123674389');

1 row created.

SQL> select * from arrange_work;

D_SSN      R_SSN      N_SSN
---------- ---------- ----------
123674356  123674334  123674389

SQL>
```

- INSERTION OF TABLE Support_staff

---

1. insert into Support_staff
   Values ('489957634');

2. insert into Support_staff
   Values ('378959208');

```
SQL> insert into support_staff values('489957634');

1 row created.

SQL> insert into support_staff values('378959208');

1 row created.

SQL> select * from support_staff;

S_SSN
----------
378959208
489957634

SQL>
```

- INSERTION OF TABLE Part_time

---

1. insert into Part_time
   Values ('378959208','45');

```
SQL> insert into part_time values('378959208','45');
1 row created.
SQL> select * from part_time;
PS_SSN     NO_HO
---------- -----
378959208  45
SQL>
```

- INSERTION OF TABLE Full_time

---

1. insert into Full_time
   Values ('489957634');

```
SQL> insert into full_time values('489957634');
1 row created.
SQL> select * from full_time;
FS_SSN
----------
489957634
SQL>
```

- INSERTION OF TABLE Driver

---

1. insert into Driver
   Values ('489957634','376AP189X8');

2. insert into Driver
   Values ('378959208','961MD375W8');

```
SQL> insert into driver values('489957634','376AP189X8');
1 row created.
SQL> insert into driver values('378959208','961MD375W8');
1 row created.
SQL> select * from driver;
S_SSN      LIC#
---------- ----------
489957634  376AP189X8
378959208  961MD375W8
SQL>
```

- INSERTION OF TABLE Department

---

1. insert into Dept
   Values ('CA123','Cardiology','Hill Street','9728830023', '123674356');

```
SQL> select * from dept;
ID         NAME           LOC                PHONE_NO   MGR_SSN
---------- -------------- ------------------ ---------- ----------
CA123      Cardiology     HillStreet         9728830023 123674356
SQL>
```

- INSERTION OF TABLE Patient

---

1. insert into Patient

Values('100','Arpit','Patel','BackPain','Maddy','arp@gmail.com','28-Feb-71','2214HillCrestDrive','NorthernBound','Dallas','Texas','75254','Male','Admitted on 11-12-2012.');

2. insert into Patient
   Values ('101','Hema','Dowson','Lever Pain','Ashtha','hem@yahoo.com','21-Aug-68', '2498 Coit road', 'Bishop Ave', 'Las Vegas','Califonia','37965','Female','Admitted on
   1st October 2013.');

```
SQL> select * from patient;

P_ID       FNAME      LNAME      REASON                 DRIVER     EMAIL_ID           DOB       LINE                 LINE2                CITY           STATE
---------- ---------- ---------- ---------------------- ---------- ------------------- --------- -------------------- -------------------- -------------- ----------
HISTORY                  ZIP
------------------------ ----------
100        Arpit      Patel      BackPain               Maddy      arp@gmail.com      28-FEB-71 2214 HillCrest Drive Northern Bound       DALLAS         Texas
Admitted on 12 Nov-2012   75254

101        Hema       Dowson     LiverPain              AAstha     hem@yahoo.com      21-AUG-68 2498 Coit road       Bisop Ave            Las Vegas      California
Admitted on 1 october13   37965

mxl123456  Robert     Downey     headache               Iron-man   robert@gmail.com   12-JUN-91 7740 MCCallum Blvd,  Apt 343             Dallas         Texas
Admitted on 12-12-95      75252

SQL>
```

- INSERTION OF TABLE In_Patient

---

1. insert into In_Patient
   Values (‘100’,’2479063589’,’ ‘, ’14-Feb-13’);
2. insert into In_patient
   Values(‘101’,’5632489632’,’20’,’ ‘);

```
SQL> select * from in_patient;

P_ID       SSN        RM#   SDATE
---------- ---------- ----- ---------
100        2479063589 20    14-FEB-13
101        5632489632 20

SQL>
```

- INSERTION OF TABLE Out_Patient

---

1. insert into Out_Patient
   Values (‘101’);

```
SQL> insert into out_patient values('101');

1 row created.

SQL> select * from out_patient;

P_ID
-----------
101

SQL>
```

- INSERTION OF TABLE Billed_Patient

---

1. insert into Billed_patient
   Values (‘100’,’1234567890’,’Maya’,’Bank of America’,’ 2214 HillCrest Dallas TX’,’04-Mar-14’);

```
SQL> select * from billed_patient;

P_ID       ACC_NO     ACC_HOLDER     BANK_NAME          BILLING_ADDR              EXP_DATE
---------- ---------- -------------- ------------------ ------------------------- ---------
100        1234567890 Maya           Bank of America    2214 Hillcrest Dallas TX  04-MAR-14

SQL>
```

- INSERTION OF TABLE Events

---

1. insert into Events
   Values (‘Cancer Aid’,’12-May-13’,’Awareness against cancer.’);

```
SQL> select * from events;

NAME       HELD_TIME DESCRIPTION
---------- --------- ----------------------------------------
Cancer Aid 12-MAY-13 awareness against cancer

SQL>
```

- INSERTION OF TABLE Event_hold_emp

---

1. insert into Event_hold_emp
   Values (‘Cancer Aid’, '123674334');

```
SQL> insert into event_hold_emp values('Cancer Aid','123674334');

1 row created.

SQL> select * from event_hold_emp;

NAME       E_SSN
---------- ---------
Cancer Aid 123674334

SQL>
```

- INSERTION OF TABLE Event_hold_vol

---

1. insert into Event_hold_vol
   Values ('Cancer Aid','200');

```
SQL> insert into event_hold_vol values('Cancer Aid','200');

1 row created.

SQL> select * from event_hold_vol;

NAME       V_ID
---------- -----------
Cancer Aid 200

SQL>
```

- INSERTION OF TABLE volunteer

---

1. insert into volunteer
   Values ('200','45','30');
2. insert into volunteer
   Values ('201','24','40');

```
SQL> insert into volunteer values('200','45','30');

1 row created.

SQL> insert into volunteer values('201','24','40');

1 row created.

SQL> select * from volunteer;

ID          AGE AVA
----------- --- ---
200         45  30
201         24  40

SQL>
```

- INSERTION OF TABLE Attendees

---

1. insert into attendees
   Values ('i1','100','Romil','75','Cancer Aid');
2. insert into attendees
   Values ('i2','101','Vasim','98', 'Cancer Aid');

```
SQL> select * from attendees;

INVITE_SR_ P_ID        RELATIVE_NAME        E_S EVENT_NAME
---------- ----------- -------------------- --- --------------------
i1         100         Romil                75  Cancer Aid
i2         101         Vasim                98  Cancer Aid

SQL>
```

- INSERTION OF TABLE Appointment

---

1. insert into appointment
   Values ('12-04-2013','13:00','100');
2. insert into appointment
   Values('09-23-13','11:00','101');

```
SQL> insert into appointment values('04-dec-13','13:00','100');

1 row created.

SQL> insert into appointment values('23-sep-13','11:00','101');

1 row created.

SQL> select * from appointment;

DT        TIME       P_ID
--------- ---------- -----------
04-DEC-13 13:00      100
23-SEP-13 11:00      101

SQL>
```

- INSERTION OF TABLE Works_At

---

1. insert into works_at
   Values ('376AP189X8','AMB020');

```
SQL> insert into works_at values('376AP189X8','AMB020');

1 row created.

SQL> select * from works_at;

D_LIC#     A_LIC#
---------- ----------
376AP189X8 AMB020

SQL>
```

. insert into ambulance
   Values ('AMB020','01-Apr-11','Drive Hill Parking');

```
SQL> insert into ambulance values('AMB020','01-APR-11','DriveHill Parking');

1 row created.

SQL> select * from ambulance;

LIC_NO     STORED_DT LOC
---------- --------- --------------------
AMB020     01-APR-11 DriveHill Parking

SQL>
```

- INSERTION OF TABLE Medicines

---

1. insert into medicines
   Values ('M01','Stopache','5','12-Feb-13','Pain Killer','150','03-Sep-13');
2. insert into medicines
   Values ('M02','Metacin','10','19-Apr-13','Coriza','200','14-Jul-13');

```
SQL> select * from medicines;

ID         NAME            PRIC STORED_IN TYPE       QTY  DISPENSE_
---------- --------------- ---- --------- ---------- ---- ---------
M01        stopache        5    12-FEB-13 Painkiller 150  03-SEP-13
M02        metacin         10   19-APR-13 Coriza     200  14-JUL-13
```

- INSERTION OF TABLE Insurance

---

1. insert into insurance
   Values ('Bajaj','P9123','100','9722430010','16-Mar-16');

```
SQL> select * from insurance;

C_NAME     POLICY_NO  P_ID       PHONE      EXP_DT
---------- ---------- ---------- ---------- ---------
Bajaj      P9123      100        9722430010 16-MAR-16

SQL>
```

- INSERTION OF TABLE Call

---

1 . insert into call
   Values ('04-Dec-13','13:00','100','Surgery','Rohan','Vinit, EmpID: 03');

```
SQL> insert into call values('04-dec-13','13:00','100','Surgery','Rohan','Vinit,Empid:03');

1 row created.

SQL> select * from call;

DT        TIME       P_ID       TREAT_TYPE           DOC_NAME             REC_INFO
--------- ---------- ---------- -------------------- -------------------- --------------------
04-DEC-13 13:00      100        Surgery              Rohan                Vinit,Empid:03

SQL>
```

- INSERTION OF TABLE Online

---

1. insert into onl
   Values ('23-Sept-13','11:00','101','Test','Rohan');

```
SQL> insert into onl values('23-sep-13','11:00','101','Test','Rohan');
1 row created.
SQL> select * from onl;
DT        TIME       P_ID       TREAT_TYPE DOC_NAME
--------- ---------- ---------- ---------- --------------------
23-SEP-13 11:00      101        Test       Rohan
SQL>
```

- INSERTION OF TABLE Relatives

---

1. insert into relatives
   Values ('100','Romil');

```
SQL> insert into relatives values('100','Romil');
1 row created.
SQL> select * from relatives;
P_ID       REL_NAME
---------- --------------------
100        Romil
SQL>
```

- INSERTION OF TABLE Bill

---

1. insert into Bill
   Values ('B600','100','Paid','40','food & Amenities','100','75',215);

```
SQL> select * from bill;
B_NO       P_ID       STATUS     BED        PRESCRIPTION         AMBULANCE  PHARM      TOTAL
---------- ---------- ---------- ---------- -------------------- ---------- ----- ----------
B232       mxp000001  UNPAID     200        Treatment and Opertn 100        10000      10100
B600       100        PAID       40         FOOD AND AMENITIES   100        75           215
SQL>
```

- INSERTION OF TABLE Event_City

---

1. insert into event_city
   Values ('Cancer Aid','Dallas');

```
SQL> insert into event_city values('Cancer Aid','Dallas');
1 row created.
SQL> select * from event_city;
EVENT_NAME           EVENT_CITY
-------------------- --------------------
Cancer Aid           Dallas
SQL>
```

- INSERTION OF TABLE E_DeptName

---

1. insert into E_DeptName
   Values ('123674356','Cardiology');

```
SQL> insert into e_deptname values('123674356','Cardiology');
1 row created.
SQL> select * from e_deptname;
E_SSN      DEPT_NAME
---------- --------------------
123674356  Cardiology
SQL>
```

- INSERTION OF TABLE PATIENT_PH_NO

---

1. insert into PATIENT_PH_NO
   Values ('100','9723101775');

```
SQL> INSERT INTO PATIENT_PH_NO VALUES('100','9723101775');
1 row created.
SQL> SELECT * FROM PATIENT_PH_NO;
P_ID       PH_NO
---------- ----------
100        9723101775
SQL>
```

5.2 Creation of Views (Answer for Question d/Phase III)

1. View1: This view returns all patients related to any treatments in the system.

CREATE VIEW T_PATIENT
AS SELECT P.P_ID,P.FNAME, P.LNAME
FROM PATIENT P, TREATMENT T
WHERE P.P_ID=T.P_ID;

```
SQL> select * from t_patient;

P_ID        FNAME      LNAME
----------- ---------- ----------
100         Arpit      Patel
mxp000001   Madhav     Patel

SQL>
```

2. View2: This view returns all patients associated with any bills with a total amount more than $10,000.

CREATE VIEW BILL_OF_PAT
AS SELECT P.P_ID, P.FNAME, P.LNAME
FROM PATIENT P, BILL B
WHERE P.P_ID=B.P_ID
AND B.TOTAL>10000;

```
SQL> create view bill_of_pat
  2  as select p.p_id,p.fname,p.lname
  3  from patient p,bill b
  4  where p.p_id=b.p_id
  5  and b.total>10000;

View created.

SQL> select * from bill_of_pat;

P_ID        FNAME      LNAME
----------- ---------- ----------
mxp000001   Madhav     Patel

SQL>
```

3. View3: This view returns all events in Dallas.

CREATE VIEW DAL_EVENT
AS SELECT E.NAME
FROM EVENTS E, EVENT_CITY EC
WHERE E.NAME=EC.EVENT_NAME AND EC.EVENT_CITY='DALLAS';

```
SQL> create view dal_event
  2  as select e.name
  3  from events e,event_city ec
  4  where e.name=ec.event_name AND ec.event_city='Dallas';

View created.

SQL> select * from dal_event;

NAME
--------------------
Cancer Aid

SQL>
```

4. View4: This view returns all senior doctors (Lev.4-5) worked on any surgery since 1/1/2013.

CREATE VIEW SENIOR_DOC
AS SELECT E.NAME,E.ID
FROM EMPLOYEE E, DOCTOR D,DOCTOR_INFO DI
WHERE D.SSN=E.SSN AND (D.D_L=4 OR D.D_L=5) AND DI.D_DDN=D.SSN AND DI.SPECIALITY='SURGEON' AND E.JOIN_IN_DATE>to_date('01-jan-13','DD-MON-YY');

```
SQL> create view senior_doc
  2  as select e.name,e.id
  3  from employee e,doctor d,doctor_info di
  4  where d.ssn=e.ssn AND (d.d_l=4 OR d.d_l=5) AND di.d_ssn=d.ssn AND di.speciality='surgeon' AND e.join_in_date>to_date('01-jan-13','DD-MON-YY');

View created.

SQL> select * from senior_doc;

no rows selected
```

5.3 Creation of SQL Queries (Answer for Question e/Phase III)

Now we give out the SQL Queries for each of 14 questions listed in Question e as follows:

1. Retrieve the IDs and Names of all nurses who live in Dallas.

```
SQL> select e.id,e.name
  2  from employee e,nurse n
  3  where n.ssn=e.ssn and e.city='Dallas';

ID          NAME
----------- ---------------
02          Balika

SQL>
```

2. Retrieve the Names and Addresses of all in-patients that lived in ward No.20 between 01/01/2013 and 03/31/2013.

SELECT P.LNAME, P.FNAME, P.LINE, P.LINE2, P.CITY
FROM PATIENT P, IN_PATIENT IP
WHERE P.P_ID=IP.P_ID
AND IP.RM#='20'
AND IP.SDATE BETWEEN to_date('01-jan-13','DD-MON-YY') AND to_date('31-mar-13','DD-MON-YY');

```
SQL> select p.lname,p.fname,p.line,p.line2,p.city
  2  from patient p, in_patient ip
  3  where p.p_id=ip.p_id
  4  AND ip.rm#=20 and ip.sdate between to_date('01-jan-13','DD-MON-YY') AND to_date('31-mar-13','DD-MON-YY');

LNAME      FNAME      LINE                 LINE2                CITY
---------- ---------- -------------------- -------------------- ---------------
Patel      Arpit      2214 HillCrest Drive Northern Bound       DALLAS
Patel      Madhav     7777 McCallum Blvd., APT 212              Dallas

SQL>
```

3. Retrieve the IDs and Names of distinct employees whose salaries are higher than the average salary of all the employees in the same department.

SELECT DISTINCT E.ID, E.NAME
FROM EMPLOYEE E, DEPT D
WHERE E.SALARY> (SELECT AVG(SALARY) FROM EMPLOYEE GROUP BY DEPT_ID) AND E.DEPT_ID=D.ID;

```
SQL> SELECT DISTINCT E.ID,E.NAME
  2  FROM EMPLOYEE E, DEPT D
  3  WHERE E.SALARY>(SELECT AVG(SALARY) FROM EMPLOYEE GROUP BY DEPT_ID) AND E.DEPT_ID=D.ID;

ID          NAME
----------- ---------------
04          Jem
03          Vinit
02          Balika
```

4. Retrieve the Names of patients who have surgery appointments with Dr. Gregory House on 01/01/2013.

SELECT P.NAME
FROM PATIENT P, CALL C
WHERE C.TREAT_TYPE='SURGERY' AND C.DOC_NAME='DR. GREGORY HOUSE' AND c.DT=to_date('01-JAN-13','DD-MON-YY') AND C.P_ID=P.P_ID;
UNION
SELECT P.NAME
FROM PATIENT P, ONL O
WHERE 0.TREAT_TYPE='SURGERY' AND O.DOC_NAME='DR. GREGORY HOUSE' AND o.DT=to_date('01-JAN-13','DD-MON-YY') AND O.P_ID=P.P_ID;

```
QL> select p.fname
 2  from patient p, call c
 3  where c.treat_type='Surgery' AND c.doc_name='Dr. GREGORY HOUSE' AND c.DT=to_date('01-jan-13','DD-MON-YY') AND c.p_id=p.p_id
 4  UNION
 5  select p.fname
 6  from patient p, onl o
 7  where o.treat_type='Surgery' AND o.doc_name='Dr. GREGORY HOUSE' AND o.DT=to_date('01-jan-13','DD-MON-YY') AND o.p_id=p.p_id;

o rows selected

QL>
```

5. Retrieve the Names and DL# of all drivers whose ages are older than 50.

SELECT E.NAME, D.LIC#
FROM EMPLOYEE E, DRIVER D
WHERE D.S_SSN=E.SSN AND E.AGE>50;

```
SQL> select e.name,d.lic#
  2  from employee e, driver d
  3  where d.s_ssn=e.ssn AND e.age>50;

NAME            LIC#
--------------- ----------
Maddy           376AP189X8

SQL>
```

6. Retrieve the total number of ambulances dispatched during 2012.

SELECT COUNT (*)
FROM AMBULANCE
WHERE STORED_DT BETWEEN to_date('01-JAN-12','DD-MON-YY') AND to_date('31-DEC-12','DD-MON-YY');

```
LIC_NO     STORED_DT LOC
---------- --------- --------------------
AMB020     01-APR-11 DriveHill Parking
AMB021     02-MAR-12 UTD Parking

SQL> select count(*)
  2  from ambulance
  3  where stored_dt between to_date('01-jan-12','DD-MON-YY') AND to_date('31-dec-12','DD-MON-YY');

  COUNT(*)
----------
         1

SQL>
```

7. Add a bank financial record to the patient ID "mxl123456".

```
SQL> insert into billed_patient values('mxl123456','0932558964','Robert','Chase Bank','7740 McCallum,Dallas','2-JAN-14'
1 row created.

SQL>
```

8. Retrieve the SSNs, and Addresses of all in-patients whose insurances have expired today.

SELECT I.SSN, P.LINE, P.LINE2, P.CITY, P.STATE, P.ZIP
FROM IN_PATIENT I, PATIENT P, INSURANCE INS
WHERE I.P_ID=INS.P_ID AND I.P_ID=P.P_ID AND INS.EXP_DT=SYSDATE;

```
SQL> select i.ssn,p.line,p.line2,p.city,p.state,p.zip
  2  from in_patient i, patient p, insurance ins
  3  where i.p_id=ins.p_id AND i.p_id=p.p_id AND ins.EXP_dt=sysdate;

no rows selected

SQL>
```

9. Retrieve the Name of the most popular Rx-medicine prescribed by senior doctors in "Cardiology" department and dispensed in 2013.

SELECT M.NAME, COUNT (M.ID)
FROM MEDICINE M, TREATS T, EMPLOYEE E, DEPT D
WHERE T.M_ID=M.ID
AND E.DEPT_ID=D.ID AND D.ID='CARDIOLOGY' AND DISPENSE_DT>12/31/2012 AND DISPENSE_DT<1/1/2014
GROUP BY M.ID

```
SQL> SELECT M.NAME,(
  2  SELECT COUNT(ID) FROM MEDICINES
  3  GROUP BY ID)
  4  FROM MEDICINES M, TREATS T, EMPLOYEE E, DEPT D
  5  WHERE T.M_ID=M.ID AND E.DEPT_ID=D.ID AND D.NAME='Cardiology' AND M.DISPENSE_DT BETWEEN TO_DATE('01-JAN-13','DD-MON-YY') AND TO_DATE('31-DEC-13','DD-MON-YY');

no rows selected

SQL>
```

10. Retrieve the Names, SSNs and Emails of distinct patients who have any bill with a total amount bigger than $10,000.

SELECT DISTINCT P.FNAME, P.LNAME, P.EMAIL_ID,INS.SSN
FROM IN_PATIENT INS, PATIENT P, BILL B
WHERE B.P_ID=INS.P_ID AND B.P_ID=P.P_ID AND B.TOTAL>10000;

```
SQL> SELECT DISTINCT P.FNAME,P.LNAME,P.EMAIL_ID,INS.SSN
  2  FROM IN_PATIENT INS, PATIENT P, BILL B
  3  WHERE B.P_ID=INS.P_ID AND B.P_ID=P.P_ID AND B.TOTAL>10000;

FNAME      LNAME      EMAIL_ID             SSN
---------- ---------- -------------------- ----------
Madhav     Patel      ma@gmail.com         2369856321

SQL>
```

11. Retrieve the Names, and Phone Numbers of all distinct patients that have appointments tested by all nurses in the department of Diagnostic Medicine.

SELECT DISTINCT P.FNAME, P.LNAME, PP.PH_NO
FROM PATIENT P, PATIENT_PH_NO PP, TREATS T, NURSE N, EMPLOYEE E,DEPT D
WHERE T.P_ID=P.P_ID
AND T.D_SSN=N.SSN
AND N.SSN=E.SSN
AND T.P_ID=PP.P_ID
AND E.DEPT_ID=D.ID
AND D.NAME='DIAGNOSTIC MEDICINE';

```
SQL> select distinct p.fname,p.lname,pp.ph_no
  2  From patient p,patient_ph_no pp, treats t, nurse n, employee e, dept d, arrange_work a
  3  where t.p_id=p.p_id AND t.d_ssn=a.d_ssn AND a.n_ssn=e.ssn AND t.p_id=pp.p_id AND e.dept_id=d.id AND d.name='Diagnostic';

FNAME      LNAME      PH_NO
---------- ---------- ----------
Madhav     Patel      9632587412
Arpit      Patel      9723101775

SQL>
```

12. Retrieve the Gross Profit of the hospital between 01/01/2012 and 12/31/2012. (To put it simply, Gross Profit = total amount of all bills issued in 2012 – sum of all employees' salaries)

SELECT BP.EXP_DATE,
(SELECT SUM(SALARY) FROM EMPLOYEE) AS EXP, (SELECT SUM(TOTAL) FROM BILL) AS INC
FROM BILL B, BILLED_PATIENT BP
WHERE B.P_ID=BP.P_ID AND BP.EXP_DATE BETWEEN TO_DATE('01-JAN-12','DD-MON-YY') AND TO_DATE('31-DEC-12','DD-MON-YY');

```
SQL> select bp.exp_date,
  2  (select sum(salary) from employee) as exp,(select sum(total) from bill) as inc
  3  from bill b, billed_patient bp
  4  where b.p_id=bp.p_id AND bp.Exp_date between to_date('01-jan-12','DD-MON-YY') AND to_date('31-dec-12','DD-MON-YY');

no rows selected

SQL>
```

13. For each year, retrieve the Date and the largest number of visits to the hospital during that day.

SELECT COUNT (*) FROM APPOINTMENT GROUP BY DT;

```
SQL> select count(*),dt from appointment group by dt;

  COUNT(*) DT
---------- ---------
         1 23-SEP-13
         1 04-DEC-13

SQL>
```

14. Retrieve the average age of all patients who only choose online scheduling services.

SELECT O.P_ID,(SELECT AVG(TRUNC(MONTHS_BETWEEN(SYSDATE,DOB)/12)) FROM PATIENT) AS AVG_AGE
FROM PATIENT P, ONL O
WHERE O.P_ID=P.P_ID;

```
SQL> select o.p_id, (select AVG(trunc(months_between(sysdate,DOB)/12)) from patient) AS AVG_AGE
  2  from patient p,onl o
  3  where o.p_id=p.p_id;

P_ID          AVG_AGE
----------- ----------
101               32.75

SQL>
```

15. Retrieve the attendees' first names and last names who participate any event of all themes in Dallas and with the evaluation score bigger and equal to 80.

SELECT P.P_ID, R.REL_NAME,P.FNAME,P.LNAME
FROM ATTENDEES A, EVENT_CITY E, PATIENT P, RELATIVES R
WHERE A.P_ID=P.P_ID,
AND A.P_ID=R.P_ID,
AND A.EVENT_NAME=E.EVENT_NAME,
AND A.E_SCORE>=80;

```
SQL> SELECT P.P_ID, R.REL_NAME,P.FNAME,P.LNAME
  2  FROM ATTENDEES A,EVENT_CITY E, PATIENT P, RELATIVES R
  3  WHERE A.P_ID=P.P_ID AND A.P_ID=R.P_ID and
  4  A.EVENT_NAME=E.EVENT_NAME AND A.E_SCORE>=80;

P_ID        REL_NAME             FNAME      LNAME
----------- -------------------- ---------- ----------
mxp000001   Vinit                Madhav     Patel
```

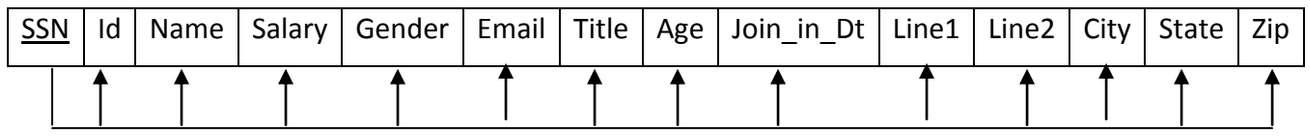
6. Dependency Diagram

We now draw a dependency diagram for each table from Figure 2 as follows:

See next page for diagram:

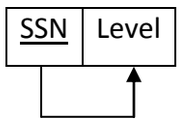
2.1 Employee

There is only one attribute on left hand side which is SSN and that is the primary key and all other attributes are deoendent on SSN.



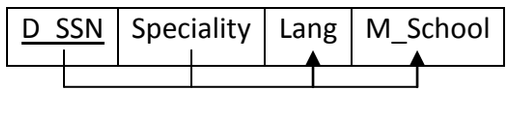
2.2 Doctor

Here level is dependent on SSN.



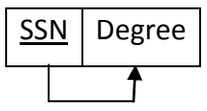
2.3 Doctor_info

Here level is dependent on SSN.



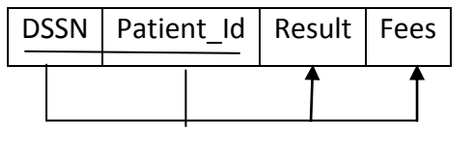
2.4 Pharmacist

Here Degree is dependent on pharmacist SSN



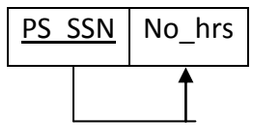
2.5 Treats

Here DSSN and Patient_id uniquely identifies which doctor treats which patient and what is the patient charged.



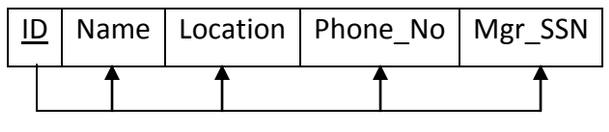
2.6 Part_Time

Here no_hrs is dependent on PS_SSN



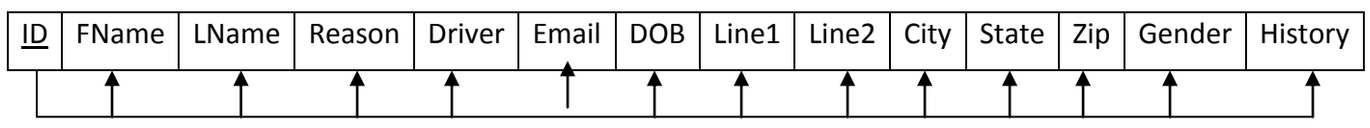
2.7 Department

Here ID uniquely identifies all the attributes



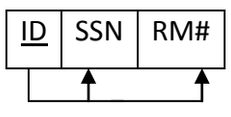
2.8 Patient

Here ID of patient is primary key which uniquely identifies all the other attributes which are dependent on ID.



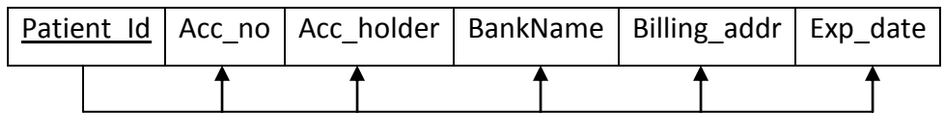
2.9 In_Patient

Here ID determines SSN of in-patient.



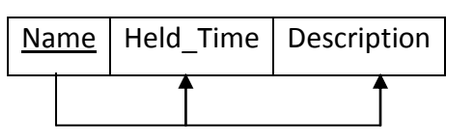
2.10 Billed_Patient

Here Patient_id uniquely identifies the all details of billed patient.



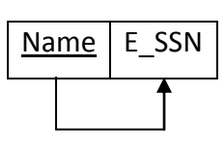
2.11 Events

Here name of event is primary key which uniquely determines Held_time and description of event.



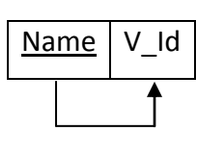
2.12 Event_Hold_Emp

Here name of Event determines which employee holds it.



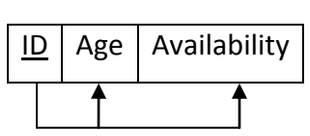
2.13 Event_Hold_Vol

Here name of volunteer determines which volunteer holds the event.



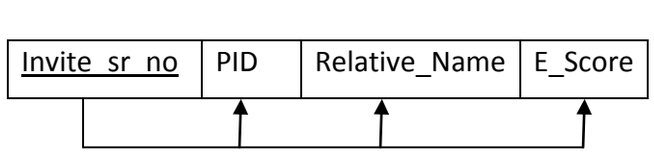
2.14 Volunteer

The ID o volunteer is primary key on which other attributes depend.



2.15 Attendees

Here Invite_sr_no uniquely determines all the information of invitees.



2.16 Appointment

Here Date and time are composite primary keys which identifies P_id to know who has made a appointment

Date	Time	PID

2.17 Medicines

Here medicines' id determines all info about medicine.

ID	Name	Price	Stored_in_dt	Type	Qty

2.18 Insurance

Here Company name and policy# are composute primary key which uniquely identifies all attributes of patient to which policy belongs.

Cpmpany_Name	Policy#	PID	Phone	Exp_dt

2.19 Ambulance

Here lic# determines the lic# of driver driving the ambulance.

Lic#	Stored_dt	Location

2.20 Call

Here Date and Time uniquely identifies the patient attributes who called to make an appointment.

Date	Time	PID	Treatment_type	Doc_name	Receptionist_info

2.21 Online

Here Date time identifies patient who made online appointment.

Date	Time	PID	Treatment_type	Doc_Name

2.22 Bill

Here B_no(Bill number) and P_id determines the bill attributes which are inturn dependent on the key.

B_No	P_ID	Status	Bed	Prescription	Ambulance	Pharmacy	Total

2.23 Nurse_lang:

Here Nurse_SSN determines N_Lang.

Nurse_SSN	N_Lang

2.24Arrange_work

D_SSN	R_SSN	Nurse_SSN

2.25Driver

S_SSN	LIC#

2.26Works_at

D_LIC#	A_LIC#

2.27Relatives

P_ID	Rel_Name

2.28Event_City

Event_Name	E_City

2.29E_DepName

ESSN	Dept_Name

2.30 Receptionist

R_SSN

2.31Nurse

SSN

2.32Out-Patient

ID

2.33Full_time

FS_SSN

2.34Support_staff

S_SSN

2.35 PATIENT_PH_NO

Here, P_id determines ph_no.

P_id	PH_NO

7. Conclusion

In this final report we summarized all the necessary descriptions and solutions for **Hospital Management System** database, including process and result of EER diagrams, relational schemas in third normal form, SQL statements to create database, create view and solve corresponding queries, as well as dependency diagram. We also implement the whole database in Oracle and using a database state to test every query. In section 2 we also explained why we use superclass/subclass relationship to build relational schema, why we choose a Relational DBMS to implement our database, and the additional five business rules shown from implementation.