Project for Database Design
Phase III. Implementation

0. Pre-Illumination

To clearly describe the implementation of our database, I separate this report into four sections. Section 1 is the project description and Section 2 is the updated relational schema that meets the third normal form to avoid anomaly in data update, insertion & deletion. Section 3 includes the dependency diagrams that I draw for each relation table. Then in Section 4, I used SQL statements to build the relational database in Oracle, created views and queries to answer business questions. Finally, a short summary is given at the end of this report.

1. Project Description

Dallas Care is a hospital and medical care center. Dallas Care would like one relational database to be able to smoothly carry out their work in an organized way. The hospital has following modules: Person, Employee, Patient, Visitors, Pharmacy, Treatment, Rooms, Records and Medical Bill Payment.

A Person can be an Employee or a Class 1 Patient. Details of a person such as Person ID, Name (First, Middle, Last), Address, Gender, Date of Birth, and Phone number (one person can have more than one phone number) are recorded. A person ID should be in the format, 'PXXX', where XXX can be a value between 100 and 999. A Class 1 patient is a person who visits the hospital just for a doctor consultation. A person can be both an employee and a Class 1 patient.

Employee is further classified as Doctors, Nurses or Receptionists. The start date of the employee is recorded. The specialization of the doctor is stored and doctors are further classified into Trainee, Permanent or Visiting. Every Class 1 patient consults a doctor. A Class 1 patient can consult at most one doctor but one doctor can be consulted by more than one Class 1 patient.

A Class 2 patient is a someone who is admitted into the hospital. A Class 2 patient can be an Employee or a Class 1 Patient or both. A doctor attends Class 2 patients. One doctor can attend many Class 2 patients but a Class 2 patient can be attended to by at most 2 doctors. The date of patient being admitted into the hospital is recorded.

A Visitor log is maintained for the Class2 Patients, which stores information such as patient ID, visitor ID, visitor name, visitor’s address, and visitor’s contact information.

Pharmacy details such as Medicine code, Name, Price, Quantity and Date of expiration is recorded. The database also stores the information of the various kinds of treatments that are offered in the hospital. The treatment details such as ID, name, duration and associated medicines are recorded. When a treatment is assigned to a Class 2 patient, the treatment details, medicine details and patient details are recorded so that the doctor can easily access this information.

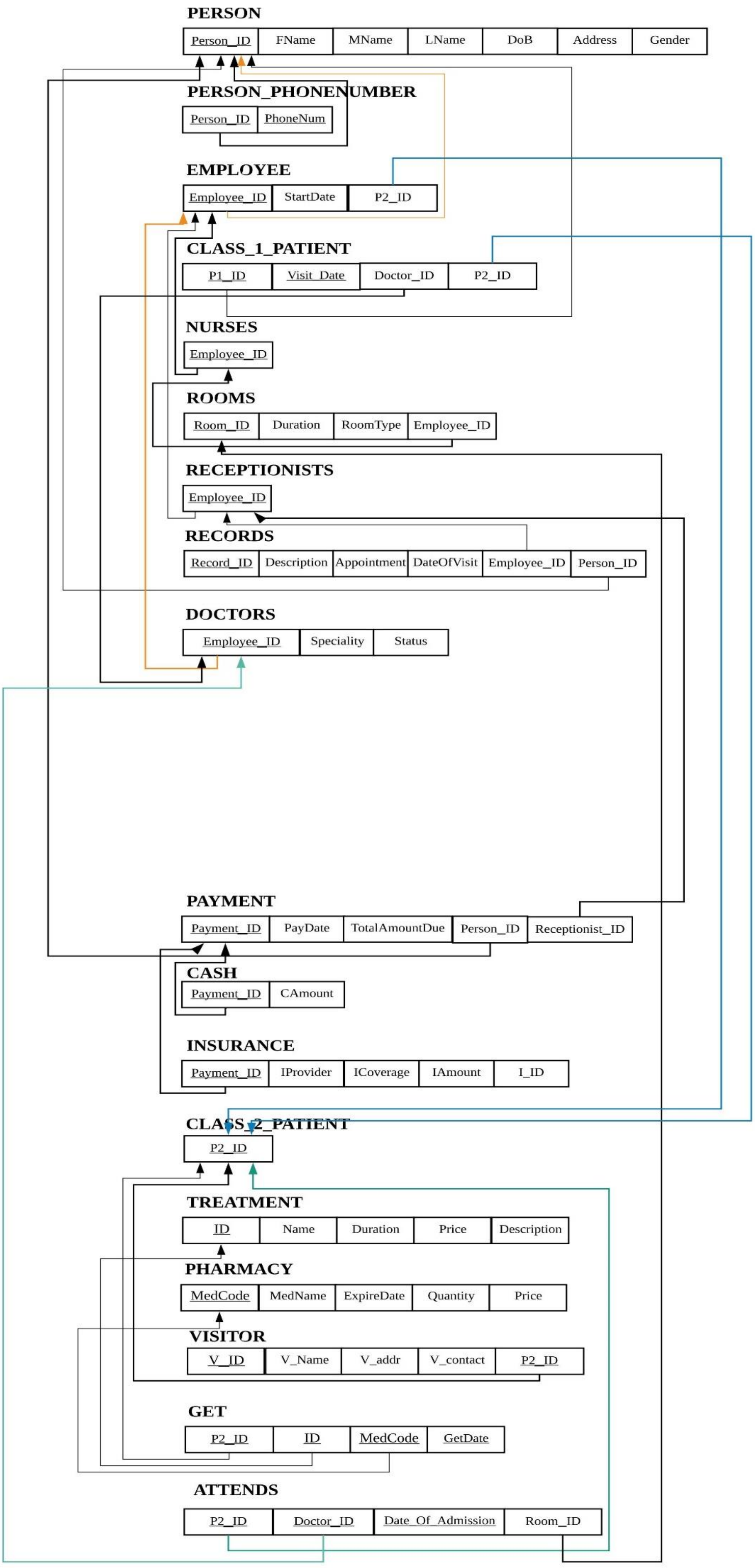
Nurses governs rooms. Each nurse can govern more than one room, but each room has only one nurse assigned to it. The room details such as room ID, room type and duration is recorded. Each Class 2 patient is assigned a room on being admitted to the hospital.

A records database is maintained by the receptionist who keeps record of information such as record ID, patient ID, date of visit, appointment and description. The receptionist also records the payment information with the patient’s ID, date of payment and the total amount due. Payment is further classified into Cash or Insurance. A person can pay by cash, or by insurance or pay via a combination of both. The cash amount is recorded if a person pays by cash. For Insurance, the insurance details such as Insurance ID, Insurance Provider, Insurance coverage and the amount is recorded.

2. Relational Schema

The updated relational schema is shown in the following figure:

Final result



PERSON

Person_ID | FName | MName | LName | DoB | Address | Gender

PERSON_PHONENUMBER

Person_ID | PhoneNum

EMPLOYEE

Employee_ID | StartDate | P2_ID

CLASS_1_PATIENT

P1_ID | Visit_Date | Doctor_ID | P2_ID

NURSES

Employee_ID

ROOMS

Room_ID | Duration | RoomType | Employee_ID

RECEPTIONISTS

Employee_ID

RECORDS

Record_ID | Description | Appointment | DateOfVisit | Employee_ID | Person_ID

DOCTORS

Employee_ID | Speciality | Status

PAYMENT

Payment_ID | PayDate | TotalAmountDue | Person_ID | Receptionist_ID

CASH

Payment_ID | CAmount

INSURANCE

Payment_ID | IProvider | ICoverage | IAmount | I_ID

CLASS_2_PATIENT

P2_ID

TREATMENT

ID | Name | Duration | Price | Description

PHARMACY

MedCode | MedName | ExpireDate | Quantity | Price

VISITOR

V_ID | V_Name | V_addr | V_contact | P2_ID

GET

P2_ID | ID | MedCode | GetDate

ATTENDS

P2_ID | Doctor_ID | Date_Of_Admission | Room_ID

3. Dependency Diagram

We now draw a dependency diagram for each table from Figure 1 as follows:

3.1 PERSON

There is only one attribute in the left-hand side of the functional dependencies, which is the key of relational schema PERSON, Person_ID. Therefore, every other attribute of this relational schema is functionally dependent on Person_ID. The dependency diagram is shown as Figure 3.1.

PERSON

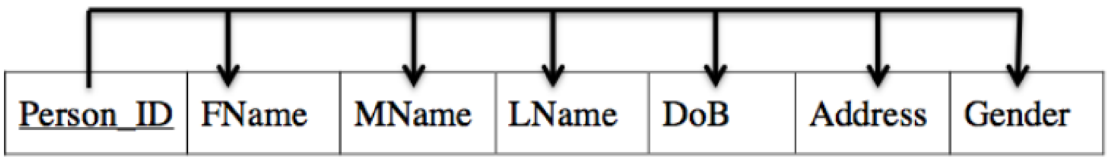


Figure 3.1. Dependency Diagram of PERSON

3.2 EMPLOYEE

There is only one attribute in the left-hand side of the functional dependencies, which is the key of relational schema EMPLOYEE, EMPLOYEE_ID. Therefore, every other attribute of this relational schema is functionally dependent on EMPLOYEE_ID. The dependency diagram is shown as Figure 3.2.

EMPLOYEE

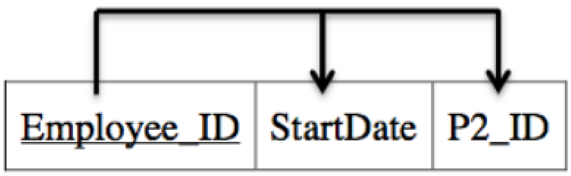


Figure 3.2 Dependency Diagram of EMPLOYEE.

3.3 CLASS_1_PATIENT

The attributes P1_ID and Visit_Date are part of the primary key. Every other attribute of this relational schema is functionally dependent on primary key. The dependency diagram is shown as Figure 3.3.

CLASS_1_PATIENT

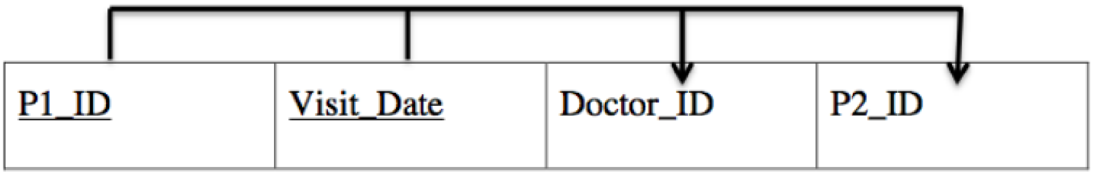


Figure 3.3 Dependency Diagram of CLASS_1_PATIENT.

3.4 ROOMS

There is only one attribute in the left-hand side of the functional dependencies, which is the key of relational schema ROOMS, Room _ID. Therefore, every other attribute of this relational schema is functionally dependent on Room _ID. The dependency diagram is shown as Figure 3.4.

ROOMS

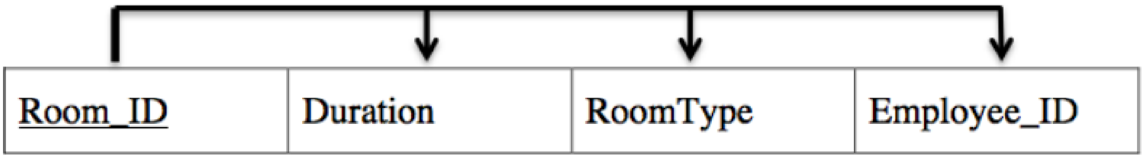
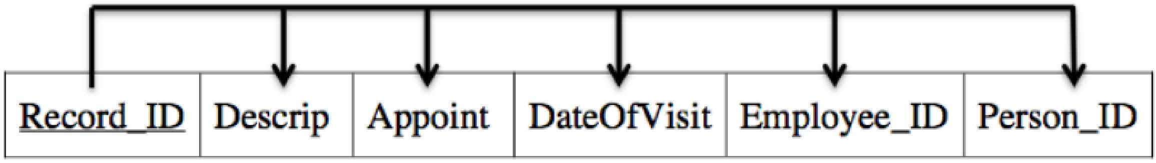


Figure 3.4 Dependency Diagram of ROOMS.

3.5 RECORDS

There is only one attribute in the left-hand side of the functional dependencies, which is the key of relational schema RECORDS, Record _ID. Therefore, every other attribute of this relational schema is functionally dependent on Record _ID. The dependency diagram is shown as Figure 3.5.

RECORDS



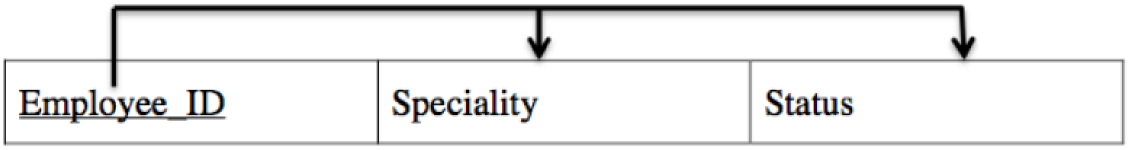
Record_ID	Descrip	Appoint	DateOfVisit	Employee_ID	Person_ID

Figure 3.5 Dependency Diagram of RECORDS.

3.6 DOCTORS

There is only one attribute in the left-hand side of the functional dependencies, which is the key of relational schema DOCTORS, Employee _ID. Therefore, every other attribute of this relational schema is functionally dependent on Employee _ID. The dependency diagram is shown as Figure 3.6.

DOCTORS



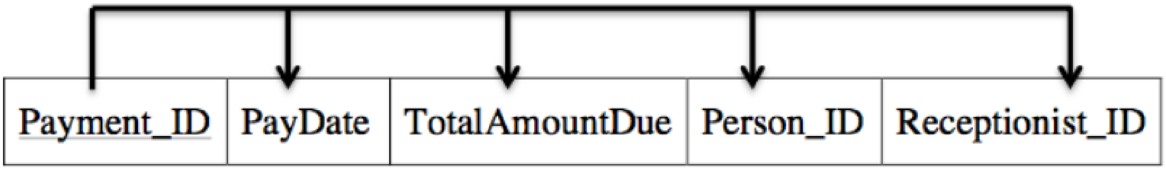
Employee_ID	Speciality	Status

Figure 3.6 Dependency Diagram of DOCTORS.

3.7 PAYMENT

There is only one attribute in the left-hand side of the functional dependencies, which is the key of relational schema PAYMENT, Payment_ID. Therefore, every other attribute of this relational schema is functionally dependent on Payment_ID. The dependency diagram is shown as Figure 3.7.

PAYMENT



Payment_ID	PayDate	TotalAmountDue	Person_ID	Receptionist_ID

Figure 3.7 Dependency Diagram of PAYMENT.

3.8 CASH

There is only one attribute in the left-hand side of the functional dependencies, which is the key of relational schema CASH, Payment_ID. Therefore, every other attribute of this relational schema is functionally dependent on Payment_ID. The dependency diagram is shown as Figure 3.8.

CASH

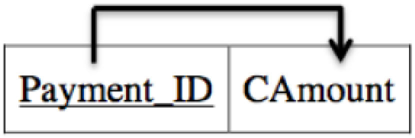


Figure 3.8 Dependency Diagram of CASH.

3.9 INSURANCE

There is only one attribute in the left-hand side of the functional dependencies, which is the key of relational schema INSURANCE, Payment_ID. Therefore, every other attribute of this relational schema is functionally dependent on Payment_ID. The dependency diagram is shown as Figure 3.9.

INSURANCE

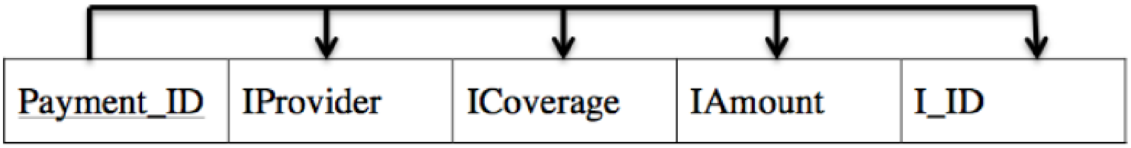


Figure 3.9 Dependency Diagram of INSURANCE.

3.10 TREATMENT

There is only one attribute in the left-hand side of the functional dependencies, which is the key of relational schema TREATMENT, ID. Therefore, every other attribute of this relational schema is functionally dependent on ID. The dependency diagram is shown as Figure 3.10.
TREATMENT

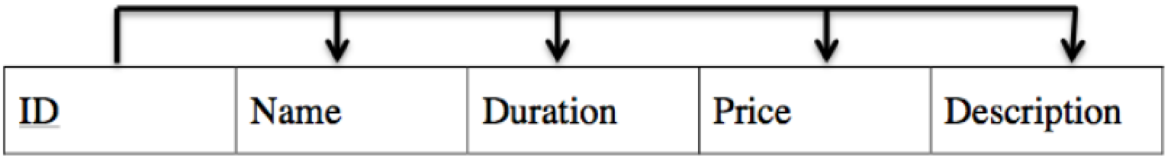


Figure 3.10 Dependency Diagram of TREATMENT.

3.11 PHARMACY

There is only one attribute in the left-hand side of the functional dependencies, which is the key of relational schema PHARMACY, MedCode. Therefore, every other attribute of this relational schema is functionally dependent on MedCode. The dependency diagram is shown as Figure 3.11.

PHARMACY

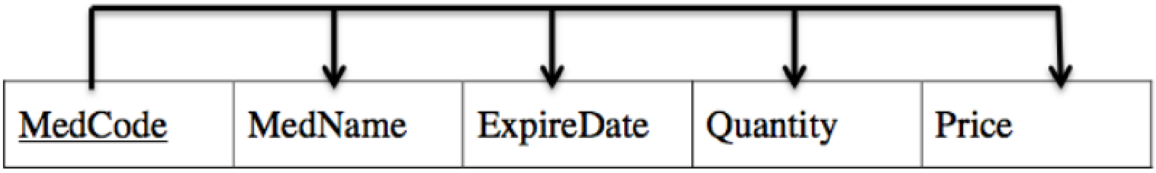


Figure 3.11 Dependency Diagram of PHARMACY.

3.12 VISITOR

There is only one attribute in the left-hand side of the functional dependencies, which is the key of relational schema VISITOR, V_ID. Therefore, every other attribute of this relational schema is functionally dependent on V_ID. The dependency diagram is shown as Figure 3.12.

VISITOR

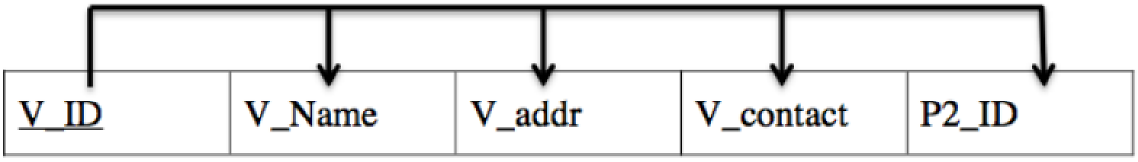


Figure 3.12 Dependency Diagram of VISITOR.

3.13 ATTENDS

The attributes P2_ID, Doctor_ID and Date_Of_Admission are part of the key. The only nonkey attribute Room_ID is functionally dependent on key. The dependency diagram is shown as Figure 3.13.
ATTENDS

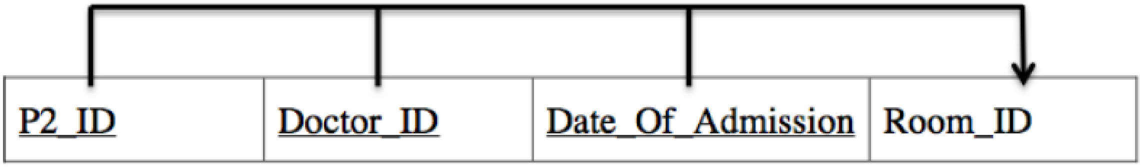
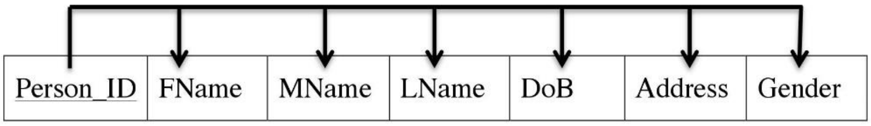


Figure 3.13 Dependency Diagram of ATTENDS.

3.14 Final Results

After drawing the dependency diagrams one after another, Figure 3.14 shows the final results for the whole database including the ones who do not have any functional dependencies.

PERSON

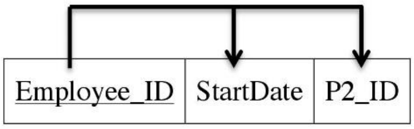


Person_ID	FName	MName	LName	DoB	Address	Gender

PERSON_PHONENUMBER

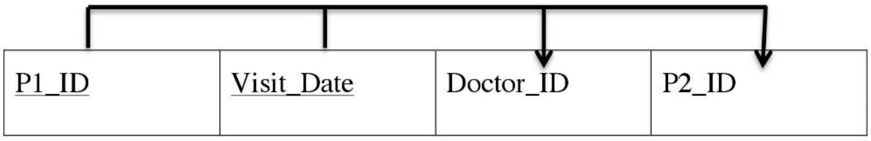
Person_ID	PhoneNum

EMPLOYEE



Employee_ID	StartDate	P2_ID

CLASS_1_PATIENT

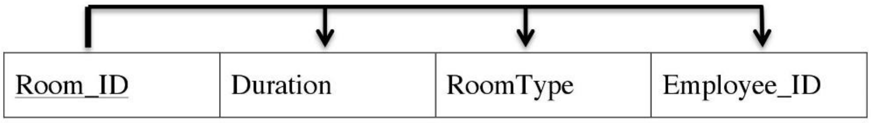


P1_ID	Visit_Date	Doctor_ID	P2_ID

NURSES

Employee_ID

ROOMS

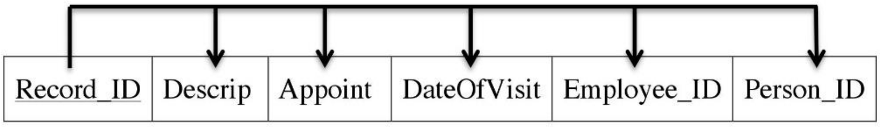


Room_ID	Duration	RoomType	Employee_ID

RECEPTIONISTS

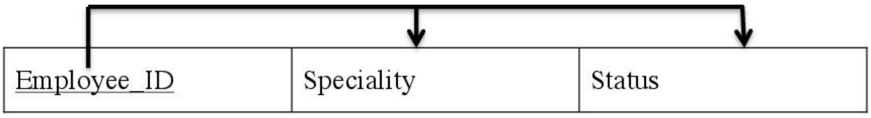
Employee_ID

RECORDS



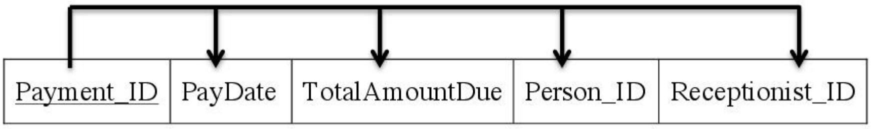
Record_ID	Descrip	Appoint	DateOfVisit	Employee_ID	Person_ID

DOCTORS



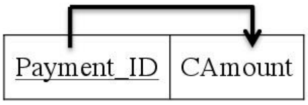
Employee_ID	Speciality	Status

PAYMENT



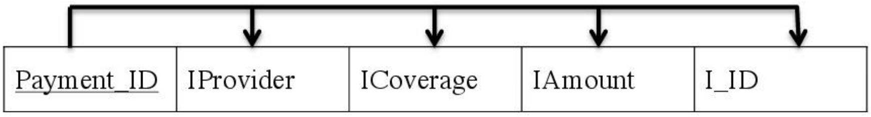
Payment_ID	PayDate	TotalAmountDue	Person_ID	Receptionist_ID

CASH



Payment_ID	CAmount

INSURANCE

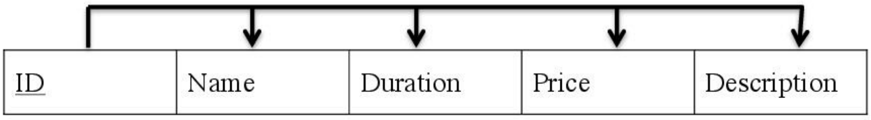


Payment_ID	IProvider	ICoverage	IAmount	I_ID

CLASS_2_PATIENT

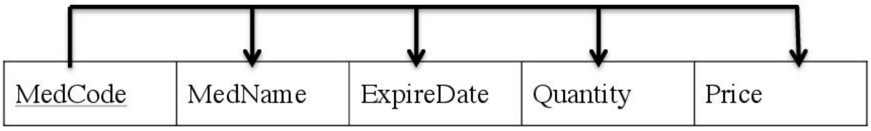
P2_ID

TREATMENT



ID	Name	Duration	Price	Description

PHARMACY



MedCode	MedName	ExpireDate	Quantity	Price

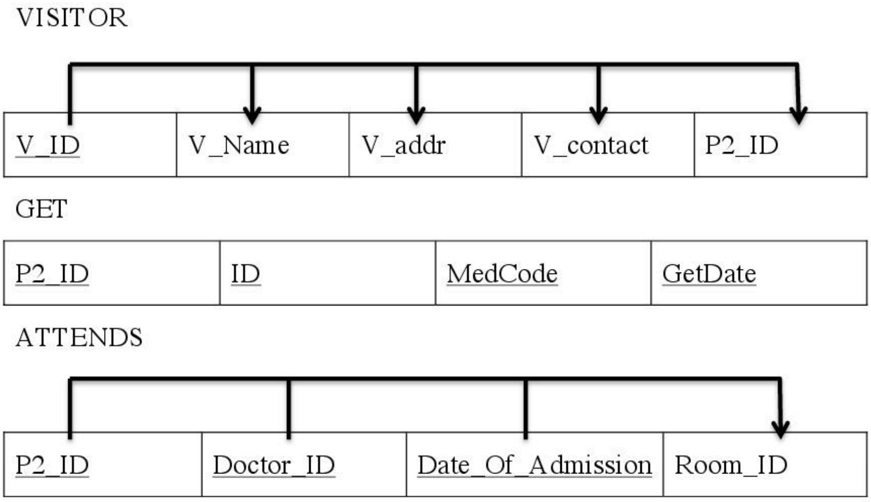


Figure 3.14 Dependency Diagram of all tables.

4. Implementation of Database

4.1 Creation of Database with SQL Statements

After normalizing every relational schema into third normal form and modifying some details, it is the time to implement our database using SQL languages into Oracle.

4.1.1 Table Creation

Using SQL statement, we created all 18 tables as follows:

PERSON:

```
CREATE TABLE PERSON (
        PERSON_ID   CHAR(4),      CHECK ( PERSON_ID like 'P%' ),
        FNAME       CHAR(30)      NOT NULL,
        MNAME       CHAR(30),
        LNAME       CHAR(30)      NOT NULL,
        DOB         DATE          NOT NULL,
        ADDRESS     CHAR(200)     NOT NULL,
        GENDER      CHAR(1)       NOT NULL,
        PRIMARY KEY (PERSON_ID)
);
```

CLASS_2_PATIENT:

```
CREATE TABLE CLASS_2_PATIENT(
        P2_ID           CHAR(4),
        PRIMARY KEY(P2_ID)
);
```

EMPLOYEE:

```
CREATE TABLE EMPLOYEE(
        EMPLOYEE_ID         CHAR(4),
        STARTDATE           DATE            NOT NULL,
        P2_ID               CHAR(4),
        PRIMARY KEY(EMPLOYEE_ID),
        FOREIGN KEY(EMPLOYEE_ID) REFERENCES PERSON(PERSON_ID),
        FOREIGN KEY(P2_ID) REFERENCES CLASS_2_PATIENT(P2_ID)
);
```

DOCTORS:

```
CREATE TABLE DOCTORS(
        EMPLOYEE_ID         CHAR(4),
        SPECIALITY          CHAR(30)        NOT NULL,
        STATUS              CHAR(1)         NOT NULL, CHECK( STATUS IN ('T','V','P') ),
        PRIMARY KEY ( EMPLOYEE_ID ),
        FOREIGN KEY ( EMPLOYEE_ID ) REFERENCES EMPLOYEE ( EMPLOYEE_ID )
);
```

NURSES:

```
CREATE TABLE NURSES (
        EMPLOYEE_ID         CHAR(4),
        PRIMARY KEY (EMPLOYEE_ID),
        FOREIGN KEY (EMPLOYEE_ID) REFERENCES EMPLOYEE ( EMPLOYEE_ID )
);
```

ROOMS:

```
CREATE TABLE ROOMS(
        ROOM_ID             INTEGER,
        Duration            INTEGER,
        ROOMTYPE            CHAR(1)         NOT NULL,
        EMPLOYEE_ID         CHAR(4),
        PRIMARY KEY(ROOM_ID),
        FOREIGN KEY(EMPLOYEE_ID) REFERENCES NURSES(EMPLOYEE_ID)
);
```

RECEPTIONISTS:

```
CREATE TABLE RECEPTIONISTS(
        EMPLOYEE_ID         CHAR(4),
        PRIMARY KEY(EMPLOYEE_ID),
        FOREIGN KEY(EMPLOYEE_ID) REFERENCES EMPLOYEE(EMPLOYEE_ID)
);
```

PAYMENT:

```
CREATE TABLE PAYMENT(
        PAYMENT_ID                      INTEGER,
        PAY_DATE                DATE,
        TOTAL_AMOUNT_DUE                NUMBER(12,2) NOT NULL,
        PERSON_ID               CHAR(4) NOT NULL,
        RECEPTIONIST_ID         CHAR(4) NOT NULL,
        PRIMARY KEY(PAYMENT_ID),
        FOREIGN KEY(PERSON_ID) REFERENCES PERSON(PERSON_ID),
FOREIGN KEY(RECEPTIONIST_ID) REFERENCES RECEPTIONISTS(EMPLOYEE_ID)
);
```

CASH:

```
CREATE TABLE CASH(
        PAYMENT_ID                      INTEGER,
        CAMOUNT                 NUMBER(12,2)            NOT NULL,
        PRIMARY KEY(PAYMENT_ID),
        FOREIGN KEY(PAYMENT_ID) REFERENCES PAYMENT(PAYMENT_ID)
);
```

INSURANCE:

```
CREATE TABLE INSURANCE(
        PAYMENT_ID                      INTEGER,
        IPROVIDER               CHAR(30)        NOT NULL,
        ICOVERAGE               CHAR(30)        NOT NULL,
        IAMOUNT                 NUMBER(12,2)            NOT NULL,
        I_ID                    INTEGER         NOT NULL,
        PRIMARY KEY(PAYMENT_ID),
        FOREIGN KEY(PAYMENT_ID) REFERENCES PAYMENT(PAYMENT_ID)
);
```

PHARMACY:

```
CREATE TABLE PHARMACY(
        MEDCODE                 NUMBER(10),
        MEDNAME                 CHAR(30)        NOT NULL,
        EXPIREDATE                      DATE            NOT NULL,
        QUANTITY                NUMBER(10) NOT NULL,
        PRICE                   NUMBER(12,2) NOT NULL,
        PRIMARY KEY(MEDCODE)
);
```

CLASS_1_PATIENT:

```
CREATE TABLE CLASS_1_PATIENT(
        P1_ID                   CHAR(4),
        VISIT_DATE              DATE            NOT NULL,
        DOCTOR_ID               CHAR(4)         NOT NULL,
        P2_ID                   CHAR(4),
        PRIMARY KEY(P1_ID, VISIT_DATE),
        FOREIGN KEY(P1_ID) REFERENCES PERSON(PERSON_ID),
        FOREIGN KEY(P2_ID) REFERENCES CLASS_2_PATIENT(P2_ID),
        FOREIGN KEY(DOCTOR_ID) REFERENCES DOCTORS(EMPLOYEE_ID)
);
```

RECORDS:

```
CREATE TABLE RECORDS(
        RECORD_ID               INTEGER,
        DESCRIPTION             CHAR(30)        NOT NULL,
        APPOINTMENT             CHAR(30),
        DATEOFVISIT             DATE            NOT NULL,
        EMPLOYEE_ID             CHAR(4)         NOT NULL,
        PERSON_ID               CHAR(4)         NOT NULL,
        PRIMARY KEY(RECORD_ID),
        FOREIGN KEY(EMPLOYEE_ID) REFERENCES ECEPTIONISTS(EMPLOYEE_ID),
        FOREIGN KEY(PERSON_ID) REFERENCES PERSON(PERSON_ID)
);
```

VISITOR:

```
CREATE TABLE VISITOR(
        V_ID                    INTEGER,
        V_Name                  CHAR(30)        NOT NULL,
        V_Addr                  CHAR(200)       NOT NULL,
        V_Contact               CHAR(30)        NOT NULL,
        P2_ID                   CHAR(4)         NOT NULL,
        PRIMARY KEY(V_ID),
        FOREIGN KEY(P2_ID) REFERENCES CLASS_2_PATIENT(P2_ID)
);
```

ATTENDS:

```
CREATE TABLE ATTENDS(
        P2_ID                   CHAR(4),
        DOCTOR_ID               CHAR(4),
        DATE_OF_ADMISSION               DATE,
        ROOM_ID                 INTEGER NOT NULL,
        PRIMARY KEY(P2_ID, DOCTOR_ID, DATE_OF_ADMISSION),
        FOREIGN KEY(ROOM_ID) REFERENCES ROOMS(ROOM_ID),
        FOREIGN KEY(P2_ID) REFERENCES CLASS_2_PATIENT(P2_ID)
);
```

TREATMENT:

```
CREATE TABLE TREATMENT(
        ID              INTEGER,
        NAME            CHAR(30)        NOT NULL,
        DURATION        INTEGER         NOT NULL,
        PRICE           NUMBER(12,2) NOT NULL,
        DESCRIPTION CHAR(30) NOT NULL,
        PRIMARY KEY(ID)
);
```

PERSON_PHONENUMBER:

```
CREATE TABLE PERSON_PHONENUMBER(
        PERSON_ID           CHAR(4),
        PHONENUM            INTEGER,
        PRIMARY KEY(PERSON_ID, PHONENUM),
        FOREIGN KEY(PERSON_ID) REFERENCES PERSON(PERSON_ID)
);
```

GET:

```
CREATE TABLE GET(
        P2_ID               CHAR(4),
        ID                  INTEGER,
        MEDCODE             NUMBER(10),
        GETDATE             DATE,
        PRIMARY KEY(P2_ID, ID, MEDCODE, GETDATE),
        FOREIGN KEY(P2_ID) REFERENCES CLASS_2_PATIENT(P2_ID),
        FOREIGN KEY(ID) REFERENCES TREATMENT(ID),
        FOREIGN KEY(MEDCODE) REFERENCES PHARMACY(MEDCODE)
);
```

4.1.2 Data Dictionary

Update data dictionary from previous delivery (4.1.1) to add data type for each attribute in addition to specifying if it is primary key, foreign key, NULL is permitted, or its value is UNIQUE.

PERSON:

Attributes	Description	Data Type	Data Format	Primary Key?	Foreign Key?	Nullable?	Value Unique?
PERSON_ID	Person ID	CHAR(4)	PXXX	y			y
FNAME	First name	CHAR(30)					
MNAME	Middle name	CHAR(30)				y	
LNAME	Last name	CHAR(30)					
DOB	Date of birth	DATE	DD-MM-YYYY				
ADDRESS	Address	CHAR(200)					
GENDER	Gender	CHAR(1)					

CLASS_2_PATIENT:

Attributes	Description	Data Type	Data Format	Primary Key?	Foreign Key?	Nullable?	Value Unique?
P2_ID	ID of class 2 patients	CHAR(4)	PXXX	y			y

EMPLOYEE:

Attributes	Description	Data Type	Data Format	Primary Key?	Foreign Key?	Nullable?	Value Unique?
EMPLOYEE_ID	Employee ID	CHAR(4)	PXXX	y	y		y
STARTDATE	Start working date	DATE	DD-MM-YYYY				
P2_ID	ID of class 2 patients	CHAR(4)	PXXX		y	y	

DOCTORS:

Attributes	Description	Data Type	Data Format	Primary Key?	Foreign Key?	Nullable?	Value Unique?
EMPLOYEE_ID	Doctor ID	CHAR(4)	PXXX	y	y		y
SPECIALITY	Specialty of the Doctor	CHAR(30)					
STATUS	'T'-Trainee, 'V'-Visiting, 'P'-Permanent	CHAR(1)					

NURSES:

Attributes	Description	Data Type	Data Format	Primary Key?	Foreign Key?	Nullable?	Value Unique?
EMPLOYEE_ID	Nurse ID	CHAR(4)	PXXX	y	y		y

ROOMS:

Attributes	Description	Data Type	Data Format	Primary Key?	Foreign Key?	Nullable?	Value Unique?
ROOM_ID	Room ID	INTEGER	XXX or XX	y			y
Duration	Duration of stay	INTEGER				y	

ROOMTYPE	Room type	CHAR(1)					
EMPLOYEE_ID	Nurse ID	CHAR(4)	PXXX		y		

RECEPTIONISTS:

Attributes	Description	Data Type	Data Format	Primary Key?	Foreign Key?	Nullable?	Value Unique?
EMPLOYEE_ID	Receptionist ID	CHAR(4)	PXXX	y	y		y

PAYMENT:

Attributes	Description	Data Type	Data Format	Primary Key?	Foreign Key?	Nullable?	Value Unique?
PAYMENT_ID	Payment ID	INTEGER		y			y
PAY_DATE	Payment date	DATE	DD-MM-YYYY			y	
TOTAL_AMOUNT_DUE	Total amount due	NUMBER(12,2)					
PERSON_ID	Patient ID	CHAR(4)	PXXX		y		
RECEPTIONIST_ID	Receptionist ID	CHAR(4)	PXXX		y		

CASH:

Attributes	Description	Data Type	Data Format	Primary Key?	Foreign Key?	Nullable?	Value Unique?
PAYMENT_ID	Payment ID	INTEGER		y	y		y
CAMOUNT	Amount paid in cash	DATE	DD-MM-YYYY				

INSURANCE:

Attributes	Description	Data Type	Data Format	Primary Key?	Foreign Key?	Nullable?	Value Unique?
PAYMENT_ID	Payment ID	INTEGER		y	y		y

IPROVIDER	Insurance provider	CHAR(30)					
ICOVERAGE	Insurance coverage	CHAR(30)					
IAMOUNT	Insurance Amount	NUMBER(12,2)					
I_ID	Insurance ID	INTEGER					

PHARMACY:

Attributes	Description	Data Type	Data Format	Primary Key?	Foreign Key?	Nullable?	Value Unique?
MEDCODE	Medicine code	NUMBER(10)		y			y
MEDNAME	Medicine name	CHAR(30)					
EXPIREDATE	Expiration date	DATE	DD-MM-YYYY				
QUANTITY	Quantity in inventory	NUMBER(10)					
PRICE	Price	NUMBER(12,2)					

CLASS_1_PATIENT:

Attributes	Description	Data Type	Data Format	Primary Key?	Foreign Key?	Nullable?	Value Unique?
P1_ID	ID of class 1 patient	CHAR(4)	PXXX	y			y
VISIT_DATE	Date of visiting the hospital	DATE	DD-MM-YYYY	y			y
DOCTOR_ID	Doctor ID	CHAR(4)	PXXX		y		
P2_ID	ID of class 2 patient	CHAR(4)	PXXX		y	y	

RECORDS:

Attributes	Description	Data Type	Data Format	Primary Key?	Foreign Key?	Nullable?	Value Unique?
RECORD_ID	RECORD ID	INTEGER		y			y

DESCRIPTION	Description of illness	CHAR(30)					
APPOINTMENT	Appointment	CHAR(30)				y	
DATEOFVISIT	Date of visiting the hospital	DATE	DD-MM-YYYY				
EMPLOYEE_ID	Receptionist ID	CHAR(4)	PXXX		y		
PERSON_ID	Patient ID	CHAR(4)	PXXX		y		

VISITORS:

Attributes	Description	Data Type	Data Format	Primary Key?	Foreign Key?	Nullable?	Value Unique?
V_ID	VISITOR ID	INTEGER		y			y
V_Name	Visitor Name	CHAR(30)					
V_Addr	Visitor address	CHAR(200)					
V_Contact	Visitor contact info	CHAR(30)					
P2_ID	Patient ID	CHAR(4)	PXXX		y		

ATTENDS:

Attributes	Description	Data Type	Data Format	Primary Key?	Foreign Key?	Nullable?	Value Unique?
P2_ID	Patient ID	CHAR(4)	PXXX	y	y		y
DOCTOR_ID	Doctor ID	CHAR(4)	PXXX	y			y
DATE_OF_ADMISSION	Date of admission	DATE	DD-MM-YYYY	y			y
ROOM_ID	ROOM ID	INTEGER			y		

TREATMENT:

Attributes	Description	Data Type	Data Format	Primary Key?	Foreign Key?	Nullable?	Value Unique?
ID	Treatment ID	INTEGER		y			y

NAME	Treatment Name	CHAR(30)					
DURATION	Duration of treatment	INTEGER					
PRICE	Price	NUMBER(12,2)					
DESCRIPTION	Description	CHAR(30)					

PERSON_PHONENUMBER:

Attributes	Description	Data Type	Data Format	Primary Key?	Foreign Key?	Nullable?	Value Unique?
PERSON_ID	Person ID	CHAR(4)	PXXX	y	y		y
PHONENUM	phone	INTEGER		y			y

GET:

Attributes	Description	Data Type	Data Format	Primary Key?	Foreign Key?	Nullable?	Value Unique?
P2_ID	Patient ID	CHAR(4)	PXXX	y	y		y
ID	Treatment ID	INTEGER		y	y		y
MEDCODE	Medicine Code	NUMBER(10)		y	y		y
GETDATE	Date of getting the treatment	DATE	DD-MM-YYYY	y		y	y

4.1.3 A Database State

We insert some values into the database in order to test our SQL create view and query statement. Here we just give one example of insertions as follows:

PERSON:

INSERTION OF TABLE PERSON

```
---------------------------------------------------------------------------------------------------
insert into PERSON
values (‘P101’, ‘John’, ‘B’, ‘Smith’, ‘1965-01-09’, ‘731 Fondren, Houston, TX’, ‘M’  );
---------------------------------------------------------------------------------------------------------
```

Table 1 shows the states for PERSON database schemas.

PERSON_I	FNAME	MNAME	LNAME	DOB	ADDRESS	GENDER
P101	John	B	Smith	09-JAN-85	731 Fondren,	M
P102	Franklin	T	Wong	08-DEC-55	638 Voss, Houston,	M
P103	Alicia	J	Zelaya	19-JAN-68	3321 Castle,	F
P104	Jennifer	S	Wallace	20-JUN-41	291 Berry,	F
P105	Ramesh	K	Narayan	15-SEP-62	975 Fire Oak,	M
P106	Joyce	A	English	31-JUL-72	5631 Rice,	F
P107	Ahmad	V	Jabbar	29-MAR-69	980 Dallas,	M
P108	James	E	Borg	10-NOV-47	450 Stone,	M
P109	Rachel	G	Green	03-AUG-67	567 Ross Rd,	F
P110	Monica		Thomas	09-OCT-65	199 Carolyns Circle,Dalla	F
P111	Orville	M	Carter	14-JAN-70	4584 Whispering Pines Circle,Dallas,T X 75240	M
P112	Mona		Payne	20-OCT-86	1527 Deercove Drive,Dallas,	F
P113	Russell	G	Berry	15-JUN-72	4557 Wilson Avenue,Dallas,T X 75204	M
P114	Anne		Reynolds	02-JUL-72	1298 Florence Street,Dallas,	F
P115	Hugh	K	GilbertAnne	08-MAR-73	2505 Carolyns Circle,Dalla	M
P116	Jonathan	L	Allison	17-APR-80	3927 Ash Street,Dallas, TX 75240	M
P117	Alexander		Walton	19-OCT-82	1515 Moore Avenue,Dallas,T X 75240	M
P118	Cecil	T	Waters	23-DEC-84	4561 Sycamore Circle,Dalla	F
P119	XINJIE1		GU1	01-JAN-90	4444,AVE 1, DALLAS 1	M

P120	XINJIE2		GU2	02-JAN-90	4444,AVE 1, DALLAS 2	M
P121	XINJIE3		GU3	03-JAN-90	4444,AVE 1, DALLAS 3	M
P122	XINJIE4		GU4	04-JAN-90	4444,AVE 1, DALLAS 4	M
P123	XINJIE5		GU5	05-JAN-90	4444,AVE 1, DALLAS 5	M
P124	XINJIE6		GU6	06-JAN-90	4444,AVE 1, DALLAS 6	M
P125	XINJIE7		GU7	07-JAN-90	4444,AVE 1, DALLAS 7	M
P126	XINJIE8		GU8	08-JAN-90	4444,AVE 1, DALLAS 8	M
P127	XINJIE9		GU9	09-JAN-90	4444,AVE 1, DALLAS 9	M
P128	XINJIE10		GU10	10-JAN-90	4444,AVE 1, DALLAS 10	M
P129	XINJIE11		GU11	11-JAN-90	4444,AVE 1, DALLAS 11	M
P130	XINJIE12		GU12	12-JAN-90	4444,AVE 1, DALLAS 12	M
P131	XINJIE13		GU13	13-JAN-90	4444,AVE 1, DALLAS 13	M
P132	XINJIE14		GU14	14-JAN-90	4444,AVE 1, DALLAS 14	M
P133	XINJIE15		GU15	15-JAN-90	4444,AVE 1, DALLAS 15	M
P134	XINJIE16		GU16	16-JAN-90	4444,AVE 1, DALLAS 16	M
P135	XINJIE17		GU17	17-JAN-90	4444,AVE 1, DALLAS 17	M
P136	XINJIE18		GU18	18-JAN-90	4444,AVE 1, DALLAS 18	M
P137	XINJIE19		GU19	19-JAN-90	4444,AVE 1, DALLAS 19	M
P138	XINJIE20		GU20	20-JAN-90	4444,AVE 1, DALLAS 20	M
P139	XINJIE21		GU21	21-JAN-90	4444,AVE 1, DALLAS 21	M
P140	XINJIE22		GU22	22-JAN-90	4444,AVE 1, DALLAS 22	M

CLASS_2_PATIENT:

INSERTION OF TABLE CLASS_2_PATIENT

```
insert into CLASS_2_PATIENT
values (‘P101’ );
```

Table 2 shows the states for CLASS_2_PATIENT database schemas.

P2_ID
P101
P102
P103
P104
P105
P108
P109
P111
P112
P115
P116
P117
P118
P126
P127
P128
P129
P130
P131
P132
P133
P134
P135
P136
P137
P138
P139
P140

EMPLOYEE:

INSERTION OF TABLE EMPLOYEE

```
insert into EMPLOYEE
values (‘P101’, ‘2018-09-10’, ‘P101’ );
```

Table 3 shows the states for EMPLOYEE database schemas.

EMPLOYEE_I D	STARTDATE	P2_ID

P109	10-AUG-16	P109
P101	10-SEP-18	P101
P102	10-OCT-18	P102
P103	07-JUL-18	P103
P104	09-OCT-16	P104
P105	13-DEC-16	P105
P110	01-MAY-17	
P111	01-JUN-17	P111
P112	05-FEB-18	P112

DOCTORS:

INSERTION OF TABLE DOCTORS

```
---------------------------------------------------------------------------------------------------
insert into DOCTORS
values (‘P101’, ‘PEDIATRICS’, ‘T’ );
---------------------------------------------------------------------------------------------------
```

Table 4 shows the states for DOCTORS database schemas.

EMPLOYE E_I D	SPECIALITY	STATUS
P109	SURGERY	P
P101	PEDIATRICS	T
P102	ENT	P
P103	SURGERY	P
P110	Pediatric	V
P111	Surgery	P
P112	ENT	T

NURSES:

INSERTION OF TABLE NURSES

```
---------------------------------------------------------------------------------------------------
insert into NURSES
values (‘P102’ );
---------------------------------------------------------------------------------------------------------------------
```

Table 5 shows the states for NURSES database schemas.

EMPLOYEE_I D
P102
P103

ROOMS:

INSERTION OF TABLE ROOMS

```
----------------------------------------------------------------------------------------------------
insert into ROOMS
values (123, 3, ‘1’, ‘P102’ );
----------------------------------------------------------------------------------------------------
```

Table 6 shows the states for ROOMS database schemas.

ROOM_ID	DURATION	ROOMTYPE	EMPLOYEE_I D
123	3	1	P102
212	4	1	P102
432	5	1	P102
133	3	2	P102
436	4	2	P102
34	5	2	P103
566	10	3	P103
543	20	3	P103

RECEPTIONISTS:

INSERTION OF TABLE RECEPTIONISTS

```
----------------------------------------------------------------------------------------------------
insert into RECEPTIONISTS
values (‘P101’ );
---------------------------------------------------------------------------------------------------
```

Table 7 shows the states for RECEPTIONISTS database schemas.

EMPLOYEE_I D
P101
P102
P103

PAYMENT:

INSERTION OF TABLE PAYMENT

```
----------------------------------------------------------------------------------------------------
insert into PAYMENT
values (1, ‘2018-11-12’, 100, ‘P101’, ‘P101’);
----------------------------------------------------------------------------------------------------
```

Table 8 shows the states for PAYMENT database schemas.

PAYMENT_ID	PAY_DATE	TOTAL_AMOUNT_DUE	PERSON_ID	RECEPTIONIS T_ID
1	12-NOV-18	100	P101	P101
2	12-NOV-18	300	P102	P102
3	11-NOV-18	1000	P103	P103
4	13-NOV-18	10	P104	P101
5	10-NOV-18	2400	P105	P102
6	11-NOV-18	150	P106	P103
7	12-NOV-18	400	P107	P101
8	13-NOV-18	300	P108	P102
9	12-NOV-18	100	P119	P101
10	12-NOV-18	300	P120	P102
11	11-NOV-18	1000	P121	P103
12	13-NOV-18	10	P122	P101
13	10-NOV-18	2400	P123	P102
14	11-NOV-18	150	P124	P103
15	12-NOV-18	400	P125	P101
16	13-NOV-18	300	P126	P102
17	14-NOV-18	100	P127	P101
18	15-NOV-18	300	P128	P102
19	16-NOV-18	1000	P129	P103
20	17-NOV-18	10	P130	P101
21	18-NOV-18	2400	P131	P102
22	19-NOV-18	150	P132	P103
23	20-NOV-18	400	P133	P101
24	21-NOV-18	300	P134	P102
25	22-NOV-18	100	P135	P101
26	23-NOV-18	300	P136	P102
27	24-NOV-18	1000	P137	P103
28	25-NOV-18	100	P138	P101

29	26-NOV-18	300	P139	P102
30	27-NOV-18	1000	P140	P103

CASH:

INSERTION OF TABLE CASH

```
----------------------------------------------------------------------------------------------------
insert into CASH
values (1, 100 );
----------------------------------------------------------------------------------------------------
```

Table 9 shows the states for CASH database schemas.

PAYMENT_I	CAMOUNT
1	100
2	300
3	1000
4	10
9	100
10	300
11	1000
12	10
13	2400
14	150
15	400
16	300
17	100
18	300
19	1000
20	10

INSURANCE:

INSERTION OF TABLE INSURANCE

```
----------------------------------------------------------------------------------------------------
insert into INSURANCE
values (5, ‘ABC’, ‘CDE’, 2400, 1);
----------------------------------------------------------------------------------------------------
```

Table 10 shows the states for INSURANCE database schemas.

PAYMENT_I	IPROVIDER	ICOVERAGE	IAMOUNT	I_ID
5	ABC	CDE	2400	1

6	ABC	CDW	150	2
7	ABC	CDE	400	3
8	ABC	DD	300	4
21	AGENT1	ALL	2400	100
22	AGENT2	ALL	150	101
23	AGENT3	ALL	400	102
24	AGENT4	ALL	300	103
25	AGENT5	ALL	100	104
26	AGENT6	ALL	300	105
27	AGENT7	ALL	1000	106
28	AGENT8	ALL	100	107
29	AGENT9	ALL	300	108
30	AGENT10	ALL	1000	109

PHARMACY:

INSERTION OF TABLE PHARMACY

```
---------------------------------------------------------------------------------------------------
insert into PHARMACY
values (1111, ‘Penicillin’, ‘2018-12-10’, 500, 30);
---------------------------------------------------------------------------------------------------
```

Table 11 shows the states for PHARMACY database schemas.

MEDCODE	MEDNAME	EXPIREDATE	QUANTITY	PRICE
1111	Penicillin	10-DEC-18	500	30
222	Amoxicillin	01-DEC-18	900	50
1234	Tetracyline	10-NOV-18	1000	100
3333	Quinine	01-JAN-19	50	300

CLASS_1_PATIENT:

INSERTION OF TABLE CLASS_1_PATIENT

```
---------------------------------------------------------------------------------------------------
insert into CLASS_1_PATIENT
values (‘P104’, ‘2018-10-10’, ‘P103’, ‘P104’ );
--------------------------------------------------------------------------------------------------
```

Table 12 shows the states for CLASS_1_PATIENT database schemas.

P1_ID	VISIT_DATE	DOCTOR_ID	P2_ID

P104	10-OCT-18	P103	P104
P105	08-AUG-18	P102	P105
P106	06-JUN-18	P101	
P107	07-JUL-18	P102	
P108	09-SEP-18	P103	P108
P119	01-JAN-17	P110	
P120	02-JAN-17	P110	
P121	03-JAN-17	P110	
P122	04-JAN-17	P110	
P123	05-JAN-17	P110	
P124	06-JAN-17	P110	
P125	07-JAN-17	P111	
P126	08-JAN-17	P111	P126
P127	09-JAN-17	P111	P127
P128	10-JAN-17	P111	P128
P129	11-JAN-17	P112	P129
P130	12-JAN-17	P112	P130
P131	02-MAR-18	P110	P131
P132	03-MAR-18	P110	P132
P133	04-MAR-18	P110	P133
P134	05-MAR-18	P110	P134
P135	06-MAR-18	P110	P135
P136	07-MAR-18	P110	P136
P137	08-MAR-18	P110	P137
P138	09-MAR-18	P110	P138
P139	10-MAR-18	P110	P139
P140	11-MAR-18	P110	P140
P114	08-JUN-18	P109	
P115	10-JUN-18	P110	P115
P116	16-JUN-18	P111	P116
P117	26-JUL-18	P112	P117
P118	17-AUG-18	P111	P118

RECORDS:

INSERTION OF TABLE RECORDS

```
----------------------------------------------------------------------------------------------------
insert into RECORDS
values (1, ‘HEART ISSUE’, ‘YES’, ‘2018-10-10’, ‘P101’, ‘P104’ );
---------------------------------------------------------------------------------------------------
```

Table 13 shows the states for RECORDS database schemas.

RECORD_ID	DESCRIPTION	APPOINTMENT	DATEOFVISIT	EMPLOYEE_I D	PERSON_ID
11	EYES	YES	06-JUN-18	P101	P106
12	EYES	YES	10-NOV-18	P102	P101
13	STOMACH	YES	01-NOV-18	P101	P102
1	HEART ISSUE	YES	13-SEP-18	P101	P108
2	EYES	YES	08-AUG-18	P102	P105
3	HEART ISSUE	YES	20-OCT-18	P102	P104
4	STOMACH	YES	07-JUL-18	P102	P107
5	HEART ISSUE	YES	09-SEP-18	P103	P108
6	EYES	YES	07-JUN-18	P102	P109
7	EYES	YES	08-JUL-18	P103	P111
8	EYES	YES	14-JUL-18	P103	P112
9	EYES	YES	10-OCT-18	P103	P104
10	EYES	YES	09-SEP-18	P103	P105
24	HEART ISSUE	YES	01-JAN-17	P101	P119
25	HEART ISSUE	YES	02-JAN-17	P102	P120
26	HEART ISSUE	YES	03-JAN-17	P103	P121
27	STOMACH	YES	04-JAN-17	P101	P122
28	STOMACH	YES	05-JAN-17	P102	P123
29	EYES	YES	06-JAN-17	P103	P124
30	EYES	YES	07-JAN-17	P101	P125
31	EYES	YES	08-JAN-17	P102	P126
32	EYES	YES	09-JAN-17	P103	P127
33	EYES	YES	10-JAN-17	P101	P128
34	STOMACH	YES	11-JAN-17	P102	P129
35	STOMACH	YES	12-JAN-17	P103	P130

36	STOMACH	YES	02-MAR-18	P101	P131
37	STOMACH	YES	03-MAR-18	P102	P132
38	STOMACH	YES	04-MAR-18	P103	P133
39	STOMACH	YES	05-MAR-18	P101	P134
40	EYES	YES	06-MAR-18	P102	P135
41	EYES	YES	07-MAR-18	P103	P136
42	EYES	YES	08-MAR-18	P101	P137
43	EYES	YES	09-MAR-18	P102	P138
44	HEART ISSUE	YES	10-MAR-18	P103	P139
45	HEART ISSUE	YES	11-MAR-18	P101	P140
46	EYES	YES	06-AUG-18	P101	P106
47	EYES	YES	18-AUG-18	P101	P106
14	EYE	YES	05-NOV-18	P102	P103
15	STOMACH	YES	08-JUN-18	P101	P114
16	HEART ISSUE	YES	10-JUN-18	P102	P115
17	STOMACH	YES	16-JUN-18	P101	P116
18	EYE	YES	26-JUL-18	P102	P117
19	HEART ISSUE	YES	17-AUG-18	P103	P118
20	STOMACH	YES	14-JUN-18	P101	P115
21	STOMACH	YES	16-JUL-18	P102	P116
22	EYE	YES	19-AUG-18	P103	P117
23	HEART ISSUE	YES	24-OCT-18	P103	P118

VISITOR:

INSERTION OF TABLE VISITOR

----

insert into VISITOR

```
values (1, ‘Ethan’, ‘731 Fondren, Houston, TX’, ‘12345’, ‘P102’ );
------------------------------------------------------------------------------------------------------
```

Table 14 shows the states for VISITOR database schemas.

V_ID	V_NAME	V_ADDR	V_CONTACT	P2_ID
1	Ethan	731 Fondren, Houston, TX	12345	P102

ATTENDS:

INSERTION OF TABLE ATTENDS

```
-----------------------------------------------------------------------------------------------------
insert into ATTENDS
values (‘P102’, ‘P101’, ‘2018-11-01’, 432 );
----------------------------------------------------------------------------------------------------
```

Table 15 shows the states for ATTENDS database schemas.

P2_ID	DOCTOR_ID	DATE_OF_ADMISSION	ROOM_ID
P101	P102	10-NOV-18	123
P102	P101	01-NOV-18	432
P103	P102	05-NOV-18	436
P104	P102	20-OCT-18	34
P105	P103	09-SEP-18	543
P108	P101	13-SEP-18	566
P126	P111	08-JAN-17	123
P127	P111	09-JAN-17	432
P128	P111	10-JAN-17	436
P129	P112	11-JAN-17	34
P130	P112	12-JAN-17	543
P131	P110	02-MAR-18	566
P132	P110	03-MAR-18	123
P133	P110	04-MAR-18	432
P134	P110	05-MAR-18	436
P135	P110	06-MAR-18	34
P136	P110	07-MAR-18	543
P137	P110	08-MAR-18	566
P138	P110	09-MAR-18	123
P139	P110	10-MAR-18	432
P140	P110	11-MAR-18	436
P109	P111	07-JUN-18	123
P111	P103	08-JUL-18	436
P112	P111	14-JUL-18	123
P115	P110	14-JUN-18	34

P116	P112	16-JUL-18	436
P117	P109	19-AUG-18	123
P118	P102	24-OCT-18	543

TREATMENT:

INSERTION OF TABLE TREATMENT

----

insert into TREATMENT

values (1, ‘HEART BYPASS’, 10, 10000, ‘HEART ISSUE’);

Table 16 shows the states for TREATMENT database schemas.

ID	NAME	DURATION	PRICE	DESCRIPTION
1	HEART BYPASS	10	10000	HEART ISSUE
2	HEART CHANGE	10	20000	HEART ISSUE
3	APPENDECTO MY	4	1000	STOMACH
4	SURGERY 2	4	1000	STOMACH
5	LASER	1	500	EYES
6	SURGERY EYES	1	200	EYES

PERSON_PHONENUMBER:

INSERTION OF TABLE PERSON_PHONENUMBER

----

insert into PERSON_PHONENUMBER

values (‘P101’, ‘632101111’);

Table 17 shows the states for PERSON_PHONENUMBER database schemas.

PERSON_ID	PHONENUM
P101	632101111
P101	3224567804
P101	3234567804

P102	3234537804
P102	3234597804
P103	3234567804
P104	3244567804
P106	3244767804
P107	3299767804
P108	3244367804

GET:

INSERTION OF TABLE GET

----

insert into GET

values (‘P104’, 1, 1111, ‘2018-10-10’);

Table 18 shows the states for GETdatabase schemas.

P2_ID	ID	MEDCODE	GETDATE
P104	1	1111	10-OCT-18
P104	3	3333	09-AUG-18
P104	4	222	02-AUG-18
P104	4	222	19-AUG-18
P104	5	222	15-AUG-18
P104	6	1234	23-AUG-18
P104	6	1234	23-NOV-18
P105	2	222	08-AUG-18
P105	2	222	17-OCT-18
P105	3	3333	01-AUG-18
P105	3	3333	03-AUG-18
P105	5	222	17-AUG-18
P108	3	3333	03-AUG-18

Till now we finished the process of creating tables and database states.

4.2 Creation of Views (Answer for Question d)

4.2.1 TopDoctor

This view returns the First Name, Last Name and Date of Joining of those doctors who have attended more than 5 Class 1 patients and over 10 Class 2 patients.

```
CREATE VIEW TOPDOCTOR AS
SELECT PM.FNAME, PM.LNAME, e.startdate As DateofJoining
FROM PERSON PM, EMPLOYEE E
WHERE pm.person_id=e.employee_id and pm.person_id IN(
select c1.doctor_id
from CLASS_1_PATIENT C1, PERSON P
WHERE P.Person_ID = c1.p1_id
Group by c1.doctor_id
HAVING COUNT(C1.DOCTOR_ID)>5
INTERSECT
select DOCTOR_ID
from ATTENDS A
Group by A.doctor_id
HAVING COUNT(a.p2_id)>10);
```

4.2.2 TopTreatment

This view returns the treatment name of the most common treatment in Dallas Care along with the bill payment amount when a person receives that treatment.

```
CREATE OR REPLACE VIEW TOP_TREATMENT (Treatment, Price) AS
SELECT T.NAME, T.PRICE
FROM TREATMENT T
WHERE T.ID IN ( SELECT ID
                FROM GET
                GROUP BY ID
                HAVING COUNT(ID) = (SELECT MAX(ID_COUNT)
                        FROM (
                                SELECT ID, COUNT(ID) ID_COUNT
                                FROM GET
                                GROUP BY ID)));
```

4.2.3 RecorderMeds

This view returns the medicines that need to be reordered. A medicine needs to be reordered if the expiration date is 1 month from current date or quantity is less than 1000.

```
CREATE OR REPLACE VIEW RecorderMeds AS
        SELECT MedCode, MedName, ExpireDate, Quantity, Price
        FROM PHARMACY
        WHERE ExpireDate = (SELECT add_months(SYSDATE, 1) from dual)
        OR Quantity < 1000;
```

4.2.4 PotentialPatient

This view returns the name, phone number and ID of patients who visited the hospital more than 3 times as a Class 1 patient but has not been admitted yet.

```
CREATE OR REPLACE VIEW PotentialPatient AS
   select P.fname,P.MNAME,p.lname,p.person_id,PN.PHONENUM
   FROM PERSON P, PERSON_PHONENUMBER PN
   where  PN.PERSON_ID=P.PERSON_ID
   AND P.person_id in ((
   select person.person_id
   from PERSON, RECORDS
   WHERE Person.PERSON_ID=RECORDS.PERSON_ID
   GROUP BY person.PERSON_ID
   HAVING
   COUNT(*)>=3)
   MINUS
   SELECT *
   FROM CLASS_2_PATIENT)
;
```

4.2.5 MostFrequentIssues

This view returns the maximum frequency of the reason that patients visit the hospital for and the associated treatment for the same. For example, if patients visit the hospital mostly complaining about heart issues then what are the treatment associated with heart issues.

```
CREATE OR REPLACE VIEW MostFrequentIssues AS
        SELECT Description, Name
        FROM TREATMENT
        WHERE Description =
        (SELECT * FROM
        (SELECT Description
        FROM RECORDS
        WHERE (ROWNUM = 1)
        GROUP BY Description
        ORDER BY COUNT(*) DESC
        ));
```

4.3 Creation of SQL Queries (Answer for Question f)

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

4.3.1 For each Doctor class, list the start date and specialization of the doctor:

```
Select STARTDATE, SPECIALITY
from EMPLOYEE, DOCTORS
WHERE DOCTORS.EMPLOYEE_ID=EMPLOYEE.employee_id
ORDER BY status;
```

4.3.2 Find the names of employees who have been admitted to the hospital within 3 months of joining:

```
SELECT P.Fname, P.Mname, P.Lname
FROM PERSON P, EMPLOYEE E, ATTENDS A
WHERE P.Person_ID = E.Employee_ID
AND E.Employee_ID = A.P2_ID
AND A.Date_of_Admission <= (SELECT add_months(E.StartDate, 3) from dual)
AND A.Date_of_Admission >= E.StartDate;
```

4.3.3 Find the average age and class (trainee, visiting or permanent) of top 5 doctors in the hospital:

```
WITH DOCTOR_AGE AS
(SELECT  P.PERSON_ID, TRUNC(months_between(sysdate, P.DOB) / 12) AS Age
FROM PERSON P),

   TOPDOCTORS AS
(SELECT * FROM(
SELECT DOCTOR_ID
FROM (SELECT DOCTOR_ID
FROM CLASS_1_PATIENT
UNION ALL
SELECT DOCTOR_ID
FROM ATTENDS)
GROUP BY DOCTOR_ID
ORDER BY COUNT(DOCTOR_ID) DESC)
WHERE ROWNUM <= 5)

SELECT D.STATUS AS Class, AVG(DA.Age) AS AverageAge
FROM DOCTOR_AGE DA, DOCTORS D, TOPDOCTORS T
WHERE DA.PERSON_ID = T.DOCTOR_ID AND D.EMPLOYEE_ID = T.DOCTOR_ID
GROUP BY D.STATUS;
```

4.3.4 Find the name of medicines associated with the most common treatment in the hospital.

```
Select P.MEDNAME
From TOPTREATMENT T, TREATMENT R, GET G, PHARMACY P
Where T.TREATMENT = R.Name AND R.ID = G.ID;
```

4.3.5 Find all the doctors who have not had a patient in the last 5 months. (Hint: Consider the date of payment as the day the doctor has attended a patient/been consulted by a patient:

```
select employee_id
from doctors
where employee_id not in(
Select c1.doctor_id
from payment p, class_1_patient c1
where c1.p1_id = p.person_id and pay_date> (SELECT add_months(SYSDATE, -5)
from dual)

union
select a.doctor_id
from payment p, attends a
where a.p2_id = p.person_id and pay_date> (SELECT add_months(SYSDATE, -5)
from dual)
 );
```

4.3.6 Find the total number of patients who have paid completely using insurance and the name of the insurance provider:

```
SELECT COUNT(P.Person_ID), I.IProvider
FROM PAYMENT P, INSURANCE I
WHERE P.Payment_ID = I.Payment_ID
AND P.Total_Amount_Due = I.IAmount
GROUP BY I.Iprovider;
```

4.3.7 Find the most occupied room in the hospital and the duration of the stay:

```
WITH MOST_USED_ROOM AS
(SELECT * FROM(
SELECT ROOM_ID, COUNT(ROOM_ID) Frequency
FROM ATTENDS A
GROUP BY ROOM_ID
ORDER BY COUNT(ROOM_ID) DESC)
WHERE ROWNUM = 1)
SELECT M.ROOM_ID, M.Frequency * R.DURATION AS Total_Duration
FROM MOST_USED_ROOM M, ROOMS R
WHERE M.ROOM_ID = R.ROOM_ID;
```

4.3.8 Find the year with the maximum number of patient visiting the hospital and the reason for their visit.

```
SELECT re.description, to_char(RE.dateofvisit, 'yyyy') AS THEYEAR
from records RE
where  to_char(RE.dateofvisit, 'yyyy') =

(SELECT T.visityear
FROM(
SELECT   to_char(r.dateofvisit, 'yyyy') AS VISITYEAR, COUNT(R.PERSON_ID) AS
NumberofPeople
FROM RECORDS R
GROUP BY  to_char(r.dateofvisit, 'yyyy')
ORDER BY NumberofPeople DESC)  T

WHERE ROWNUM =1);
```

4.3.9 Find the duration of the treatment that is provided the least to patients:

```
with newt as (
select id, count(*) cnt
from get
group by id
)
select duration
from newt, treatment
where cnt in (select min(cnt) from newt) and newt.id = treatment.id
;
```

4.3.10 List the total number of patients that have been admitted to the hospital after the most current employee has joined:

```
SELECT COUNT(DISTINCT P2_ID)
FROM ATTENDS
WHERE Date_of_Admission >
(SELECT MAX(StartDate) FROM EMPLOYEE);
```

4.3.11 List all the patient records of those who have been admitted to the hospital within a week of being consulted by a doctor:

```
SELECT PERSON_ID AS PATIENT_ID, RECORD_ID, DESCRIPTION, APPOINTMENT,
DATEOFVISIT, EMPLOYEE_ID AS RECEPTIONIST_ID
FROM RECORDS R
WHERE R.PERSON_ID IN (SELECT A.P2_ID
FROM CLASS_1_PATIENT C1, ATTENDS A
WHERE C1.P2_ID = A.P2_ID AND (TO_DATE(C1.VISIT_DATE) + 7 >=
TO_DATE(A.DATE_OF_ADMISSION )));
```

4.3.12 Find the total amount paid by patients for each month in the year 2017.

```
Select SUM(P.TOTAL_AMOUNT_DUE)
FROM PAYMENT P
WHERE  to_char(P.PAY_DATE, 'yyyy') = '2017';
```

4.3.13 Find the name of the doctors of patients who have visited the hospital only once for consultation and have not been admitted to the hospital.

```
select fname, lname, person_id
from person
where person_id in (select employee_id
from records
where person_id in(
select person_id
from RECORDS
where person_id in (
select c1.p1_id
FROM CLASS_1_PATIENT c1

minus
select c2.p2_id
from class_2_patient c2
)
group by person_id
having count(*)=1))
;
```

4.3.14 Find the name and age of the potential patients in the hospital:

```
SELECT Fname, MNAME, Lname, TO_CHAR(SYSDATE,'YYYY') - TO_CHAR(DoB,'YYYY')
FROM PERSON
WHERE Person_ID IN (
SELECT PERSON_ID FROM POTENTIALPATIENT);
```

5. Conclusion

In this report we modified the EER diagram and relational schemas for Dallas Care Database according to fit the third normal form. We also draw dependency diagram for each relational schema in database. Then we created tables for each relational schema and inserted the appropriate data for each table. Then we created views and used queries to answer related business questions.