Project for Database Design

Phase IV. Documentation

GROUP 14 AUTHORS
Haoran Lou
Yao Li
Yongliang Yang

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.

0. Problem 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.

1. Three Questions

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

Solution:

Yes .it’s important.

Since all subclass entity inherit all attribute from super-class. in this way , all the duplicate attribute could avoid。

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

Add new attribute class_2_ID as surrogate key to class 1 patient and employ, can easily find the information of class_2_patient and inherit attribute from either class 1 patient or employ or both.

For each doctor, nurse, receptionist, Add new attribute _id ,so easily to locate which event they involved.

Assume not all people has patient id, and just class 1 patients have patient id ,to assure employee and class 1 patient could be distinguished.

Add new relation medical information, to collect information of class 2patient and their treatment and which medicine involved.

Add new relation access,to help doctor handle all class 2 patient information.

Assume every class 1 patient come hospital for consult doctors.so we can get how many times the class 1 patient consult a doctor via relation Records’ visit date

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

Database management systems are systems that manage the full data structure and exercise full control over the data stored in an organization’s database. As compared to the traditional approaches of maintaining data in an organization, the modern system has a number of advantages. Organizational data is always susceptible to losses and therefore a proper system is highly recommended when a lot of data exists. The following are benefits of using database management systems in an organization.

Data Sharing Is Improved In The Organization

Proper database management systems help in gaining better access to data as well as better management of the data. In turn, better access helps the end users share the data fast and effectively across the organization.

Improvement In Data Security

A better framework is provided for enforcement of data privacy and security policies. The risks of data security breaches are minimized and corporate data is used properly.

Effective Data Integration

When data management is improved, it promotes an integrated picture of an organization’s operations. It becomes easy to see how operations in one segment of the organization affects other segments of the organization. Thus, effective integration of data is accomplished through the use of data management solutions.

Database Management Systems Minimize Data Inconsistency

Data inconsistency occurs when different versions of data exist in different places in an organization. By using a proper management system and data quality management tools, the problem of data inconsistency is minimized.

Better Access To Data

A management system helps in getting quick solutions to database queries, and therefore, data access is faster and more accurate. End users like sales people will have enhanced access to the data, enabling a faster sales cycle and a more sound decision making process.

Increase In Productivity Of The End User

By deploying the best data quality tools and database management systems, the productivity of the end user is increased. With the data management tools, the end users are empowered to make quick and informed decisions that can decide the success and failure of a company in a long run.

Quick Decision Making

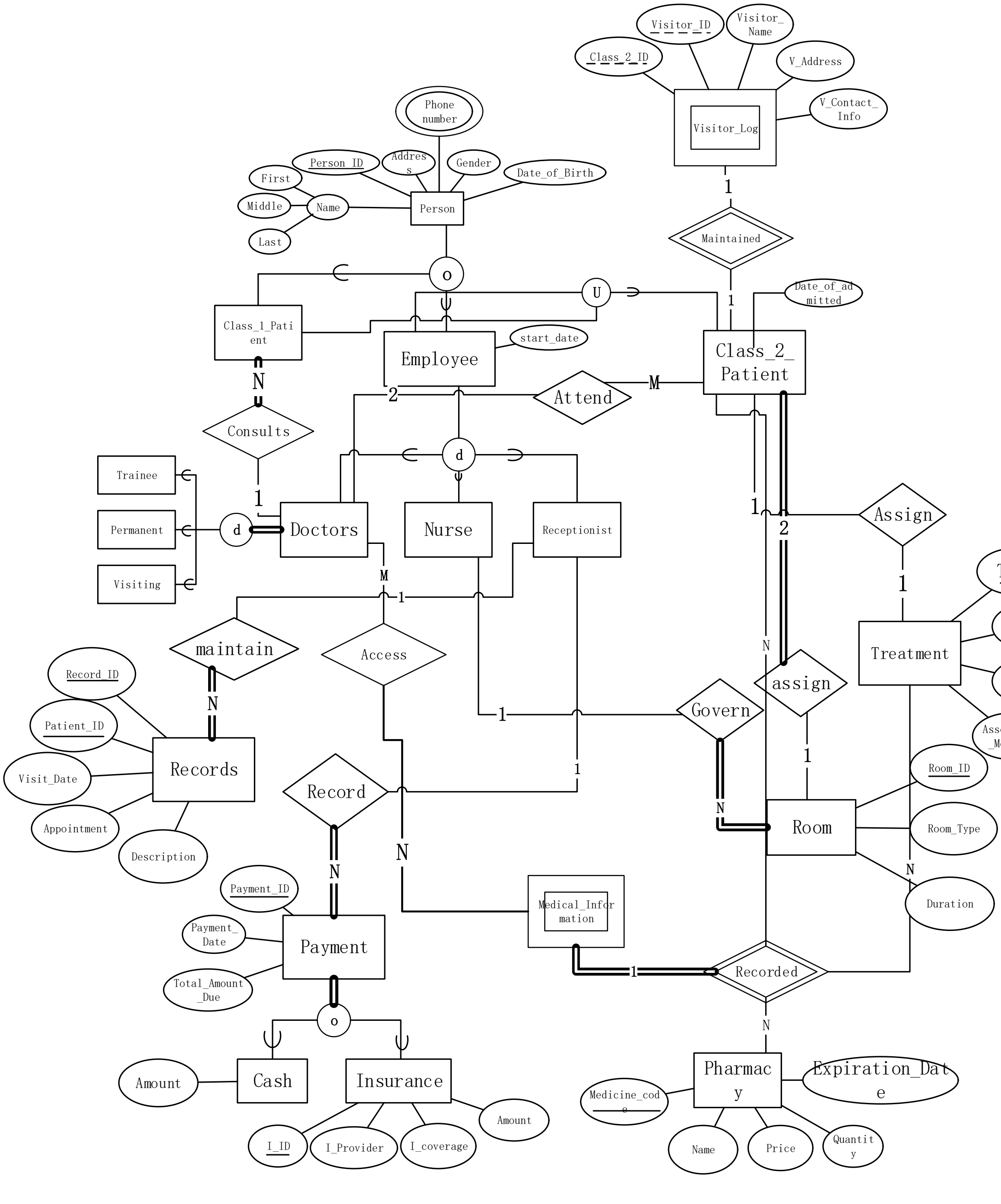
When data is better managed and access is improved, quality information is generated and the user is enabled to make faster decisions. A good database managing system helps in providing a framework to facilitate data quality initiatives and in turn, higher quality information helps in making better, faster decisions in an organization.

Looking to implement a data management system into your organization? Look no further because RingLead Data Management Solutions (DMS) has got you covered.

RingLead's cloud-based DMS platform can capture, clean, protect and enrich all of the data inside your CRM or Marketing Automation System in real time. RingLead DMS Cleanse can remove duplicates currently clogging up your database and prevent more from entering from web submissions, list imports and manual entry using DMS Duplicate Prevention. Additionally, RingLead DMS Enrichment can enrich all of your data in batch or list using crowdsourced data for the highest per field match rates in the industry.

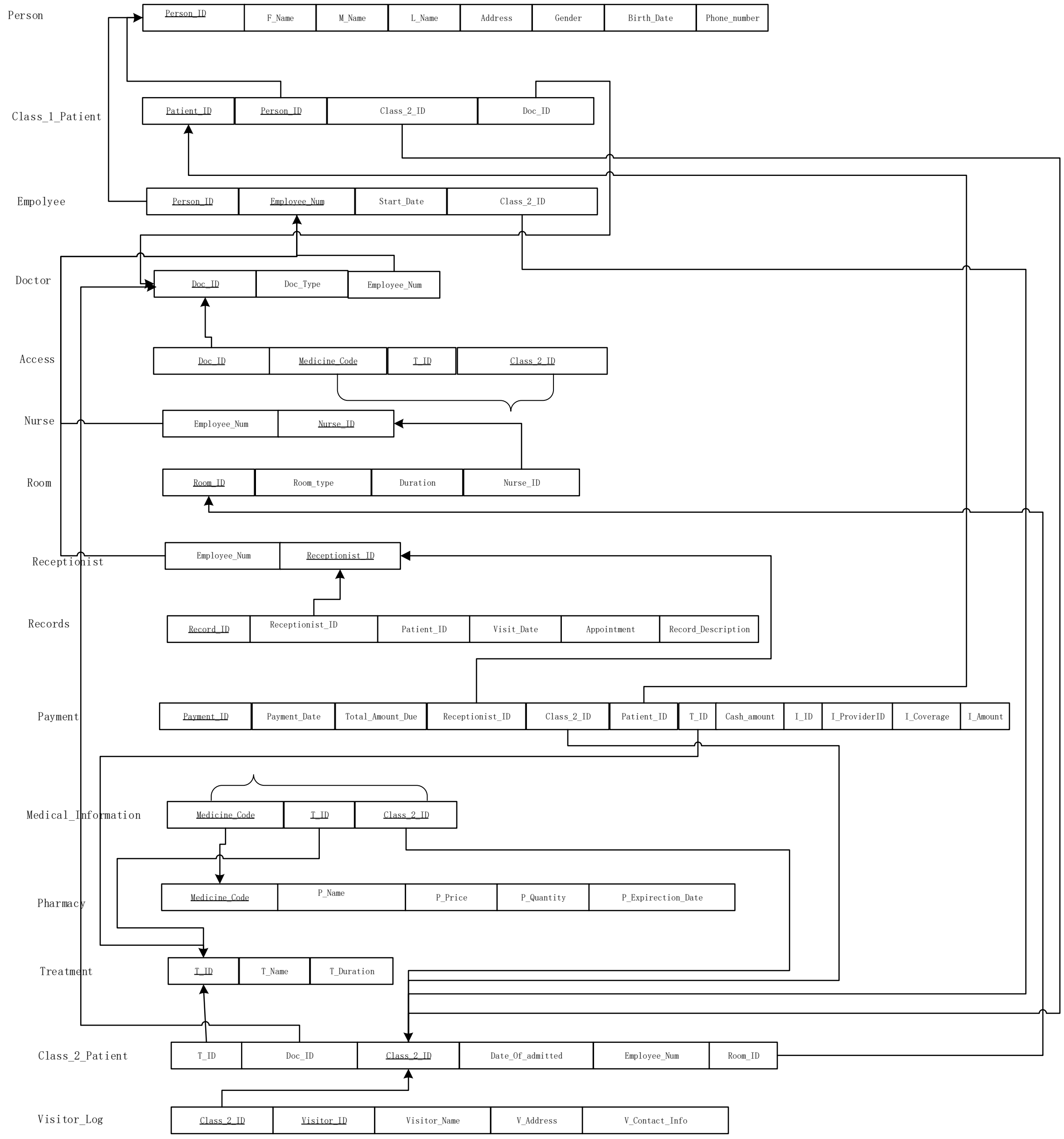
For the hospital, it meets all the requirements and the cost is not hinder the way compared with the business benefits. On the other hand, the hospital indeed requires the different access for different users and the whole system is not sample enough with just basic records.

2. EER diagram with all assumptions



3. Relational Schema in Third Normal Form

3.1 Relational Schema



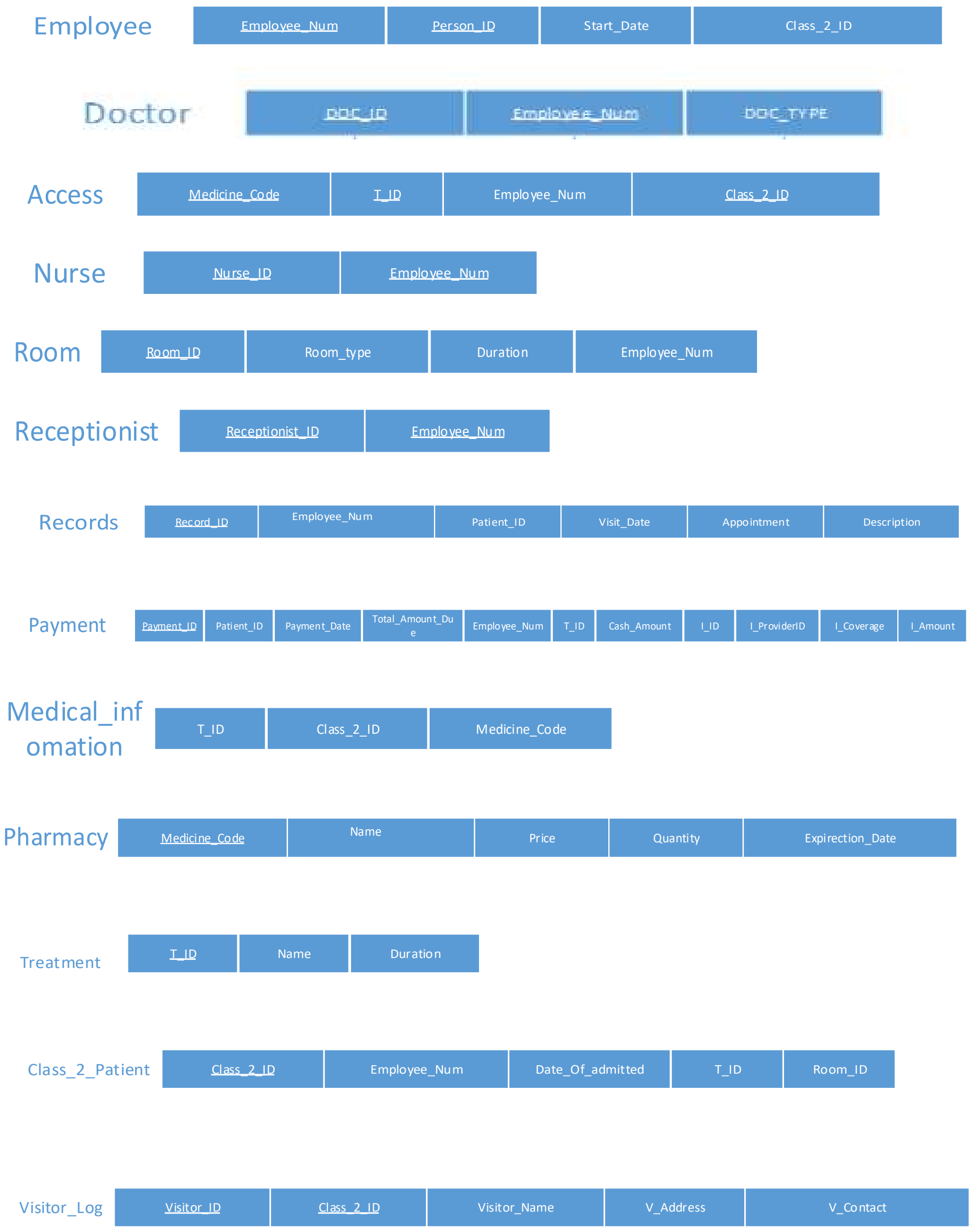
3.2 Format for Every Relation

Person

Person_ID	F_Name	M_Name	L_Name	Address	Gender	Birth_Date	Phone

Class_1_Patient

Patient_ID	Person_ID	Class_2_ID	Employee_Num



Employee: Employee_Num, Person_ID, Start_Date, Class_2_ID

Doctor: DOC_ID, Employee_Num, DOC_TYPE

Access: Medicine_Code, T_ID, Employee_Num, Class_2_ID

Nurse: Nurse_ID, Employee_Num

Room: Room_ID, Room_type, Duration, Employee_Num

Receptionist: Receptionist_ID, Employee_Num

Records: Record_ID, Employee_Num, Patient_ID, Visit_Date, Appointment, Description

Payment: Payment_ID, Patient_ID, Payment_Date, Total_Amount_Due, Employee_Num, T_ID, Cash_Amount, I_ID, I_ProviderID, I_Coverage, I_Amount

Medical_information: T_ID, Class_2_ID, Medicine_Code

Pharmacy: Medicine_Code, Name, Price, Quantity, Expirection_Date

Treatment: T_ID, Name, Duration

Class_2_Patient: Class_2_ID, Employee_Num, Date_Of_admitted, T_ID, Room_ID

Visitor_Log: Visitor_ID, Class_2_ID, Visitor_Name, V_Address, V_Contact

4. All Requested SQL Statements

4.1Creation of Database with SQL Statements

4.1.1 Table Creation

Using SQL statement, we created 15 tables as follows:

- ACCSEE1

```sql
CREATE TABLE ACCESS1
(
  DOC_ID VARCHAR(200) NOT NULL,
  MEDICINE_CODE VARCHAR(200) NOT NULL,
  T_ID VARCHAR(200) NOT NULL,
  CLASS_2_ID VARCHAR(200) NOT NULL,
  primary key (DOC_ID, MEDICINE_CODE, T_ID, CLASS_2_PATIENT_ID),
  foreign key (MEDICINE_CODE, T_ID, CLASS_2_ID) references MEDICAL_INFORMATION(MEDICINE_CODE, T_ID, CLASS_2_ID),
  foreign key (DOC_ID) references DOCTOR(DOC_ID)
);
```

- CLASS_1_PATIENT

```sql
CREATE TABLE CLASS_1_PATIENT
(
  PATIENT_ID VARCHAR(255) NOT NULL,
  PERSON_ID VARCHAR(255) NOT NULL,
  CLASS_2_ID VARCHAR(255),
  EMPLOYEE_NUM VARCHAR(255),
  primary key (PATIENT_ID),
  foreign key (PERSON_ID) references PERSON(PERSON_ID)
);
alter table CLASS_1_PATIENT
add DOC_ID varchar(255) null;
alter table CLASS_1_PATIENT
add foreign key (DOC_ID) references DOCTOR(DOC_ID);
```

- CLASS_2_PATIENT

```sql
CREATE TABLE CLASS_2_PATIENT
(
  T_ID VARCHAR(200),
  EMPLOYEE_NUM VARCHAR(200),
  CLASS_2_ID VARCHAR(200) NOT NULL,
  DATE_OF_ADMITTED DATE,
  ROOM_ID VARCHAR(255),
  primary key(CLASS_2_ID),
  foreign key(T_ID) references TREATMENT(T_ID),
  foreign key(ROOM_ID) references ROOM(ROOM_ID)
);
alter table CLASS_2_PATIENT
add DOC_ID varchar(255) null;
alter table CLASS_2_PATIENT
add foreign key (DOC_ID) references DOCTOR(DOC_ID);
```

- DOCTOR

```sql
CREATE TABLE DOCTOR
(
  EMPLOYEE_NUM VARCHAR(255) NOT NULL,
  DOC_ID VARCHAR(255),
  DOC_TYPE VARCHAR(255),
  primary key(DOC_ID),
  foreign key (EMPLOYEE_NUM) references EMPLOYEE(EMPLOYEE_NUM)
);
```

- EMPLOYEE

```
CREATE TABLE EMPLOYEE
(
  PERSON_ID VARCHAR(255) NOT NULL,
  EMPLOYEE_NUM VARCHAR(255) NOT NULL,
  START_DATE DATE,
  CLASS_2_ID VARCHAR(255),
  primary key (EMPLOYEE_NUM),
  foreign key (PERSON_ID) references PERSON(PERSON_ID)
);
```

- MEDICAL_INFORMATION

```
CREATE TABLE MEDICAL_INFORMATION
(
  MEDICINE_CODE VARCHAR(200) NOT NULL,
  T_ID VARCHAR(200) NOT NULL,
  CLASS_2_ID VARCHAR2(200) NOT NULL,
  primary key (MEDICINE_CODE,T_ID,CLASS_2_ID),
  foreign key(T_ID) references TREATMENT(T_ID),
  foreign key(CLASS_2_ID) references CLASS_2_PATIENT(CLASS_2_ID)
);
```

- NURSE

```
CREATE TABLE NURSE
(
  EMPLOYEE_NUM VARCHAR(255) NOT NULL,
  NURSE_ID VARCHAR(255) NOT NULL,
  primary key(NURSE_ID),
  foreign key (EMPLOYEE_NUM) references EMPLOYEE(EMPLOYEE_NUM)
);
```

- PYMENT

```
CREATE TABLE PAYMENT
(
  PAYMENT_ID VARCHAR2(20) NOT NULL,
  PAYMENT_DATE DATE,
  TOTAL_AMOUNT_DUE VARCHAR(20),
  RECEPTIONIST_ID VARCHAR(20),
  PATIENT_ID VARCHAR(40),
  T_ID VARCHAR(20),

  CASH_AMOUNT VARCHAR(100),
  I_ID VARCHAR(60),
  I_PROVIDERID VARCHAR(100),
  I_COVERAGE VARCHAR(250),
  I_AMOUNT VARCHAR(100),
  primary key (PAYMENT_ID),
  foreign key (RECEPTIONIST_ID) references RECEPTIONIST(RECEPTIONIST_ID),
  foreign key (PATIENT_ID) references CLASS_1_PATIENT(PATIENT_ID)
);
alter table PAYMENT
add CLASS_2_ID varchar(255) null;
alter table PAYMENT
add foreign key (CLASS_2_ID) references CLASS_2_PATIENT(CLASS_2_ID);
alter table PAYMENT
add foreign key (T_ID) references TREATMENT(T_ID);
```

- PERSON

```
CREATE TABLE PERSON
(
  PERSON_ID VARCHAR(255) NOT NULL,
  F_NAME VARCHAR(255) NOT NULL,
  M_NAME VARCHAR(255),
  L_NAME VARCHAR(255) NOT NULL,
  ADDRESS VARCHAR(255),
  GENDER VARCHAR(255),
  BITH_DATE DATE,
  PHONE_NUMBER VARCHAR(255),
  primary key (PERSON_ID)
);
```

- PHARMACY

```
CREATE TABLE PHARMACY
(
  MEDICINE_CODE VARCHAR(200) NOT NULL,
  P_PRICE VARCHAR(200),
  P_NAME VARCHAR(200),
  P_QUANTITY VARCHAR(200),
  P_EXPIRECTION_DATE DATE,
  primary key(MEDICINE_CODE)
);
```

- RECEPTIONIST

```
CREATE TABLE RECEPTIONIST
(
  RECEPTIONIST_ID VARCHAR(255) NOT NULL,
  EMPLOYEE_NUM VARCHAR(255) NOT NULL,
  primary key (RECEPTIONIST_ID),
  foreign key (EMPLOYEE_NUM) references EMPLOYEE(EMPLOYEE_NUM)
);
```

- RECORDS

```
CREATE TABLE RECORDS
(
  RECORD_ID VARCHAR(255) NOT NULL,
```

```sql
    RECEPTIONIST_ID VARCHAR(255),
    PATIENT_ID VARCHAR(255),
    VISIT_DATE DATE,
    APPOINTMENT DATE,
    RECORD_DESCRIPTION VARCHAR(255),
    primary key(RECORD_ID),
    foreign key (RECEPTIONIST_ID) references RECEPTIONIST(RECEPTIONIST_ID),
    foreign key (PATIENT_ID) references CLASS_1_PATIENT(PATIENT_ID)
);
```

- ROOM

```sql
CREATE TABLE ROOM
(
    ROOM_ID VARCHAR(255) NOT NULL,
    ROOM_TYPE VARCHAR(255),
    ROOM_DURATION VARCHAR(255),
    NURSE_ID VARCHAR(255) NOT NULL,
    primary key(ROOM_ID),
    foreign key (NURSE_ID) references NURSE(NURSE_ID)
);
```

- TREATMENT

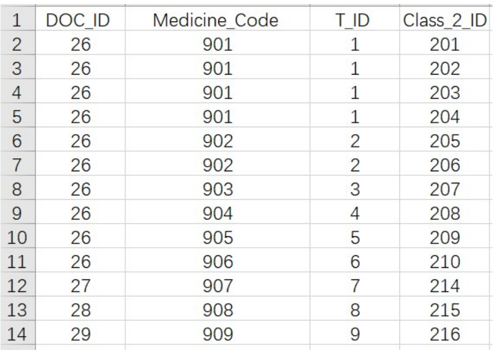
```sql
CREATE TABLE TREATMENT
(
    T_ID VARCHAR(200) NOT NULL,
    T_NAME VARCHAR(250),
    T_DURATION VARCHAR(200),
    primary key(T_ID)
);
```

- VISITOR_LOG

```sql
CREATE TABLE VISITOR_LOG
(
    CLASS_2_ID VARCHAR(200),
    VISITOR_ID VARCHAR(200) NOT NULL,
    VISITOR_NAME VARCHAR(200),
    V_ADDRESS VARCHAR(200),
    V_CONTACT_INFO VARCHAR(200),
    primary key(VISITOR_ID),
    foreign key (CLASS_2_ID) references CLASS_2_PATIENT(CLASS_2_ID)
);
```

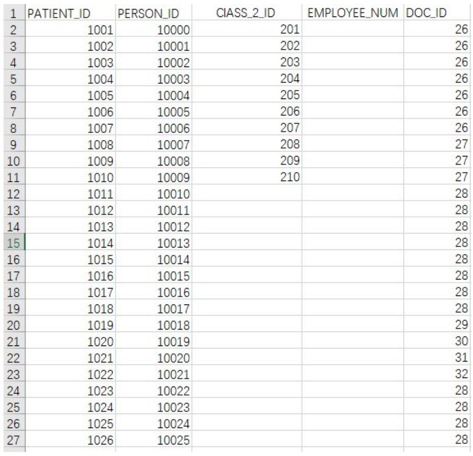
4.1.2 A Database State

- ACCSEE1



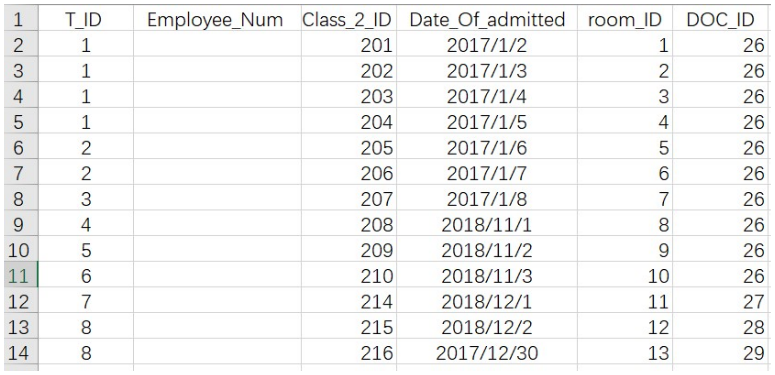
1	DOC_ID	Medicine_Code	T_ID	Class_2_ID
2	26	901	1	201
3	26	901	1	202
4	26	901	1	203
5	26	901	1	204
6	26	902	2	205
7	26	902	2	206
8	26	903	3	207
9	26	904	4	208
10	26	905	5	209
11	26	906	6	210
12	27	907	7	214
13	28	908	8	215
14	29	909	9	216

- CLASS_1_PATIENT；



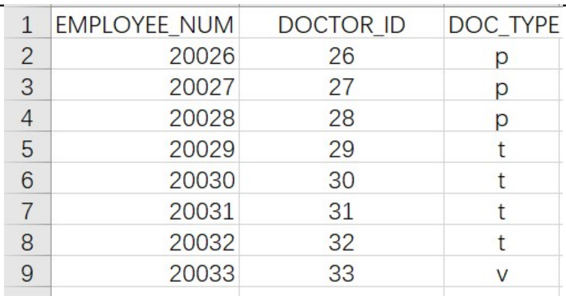
1	PATIENT_ID	PERSON_ID	ClASS_2_ID	EMPLOYEE_NUM	DOC_ID
2	1001	10000	201		26
3	1002	10001	202		26
4	1003	10002	203		26
5	1004	10003	204		26
6	1005	10004	205		26
7	1006	10005	206		26
8	1007	10006	207		26
9	1008	10007	208		27
10	1009	10008	209		27
11	1010	10009	210		27
12	1011	10010			28
13	1012	10011			28
14	1013	10012			28
15	1014	10013			28
16	1015	10014			28
17	1016	10015			28
18	1017	10016			28
19	1018	10017			28
20	1019	10018			29
21	1020	10019			30
22	1021	10020			31
23	1022	10021			32
24	1023	10022			28
25	1024	10023			28
26	1025	10024			28
27	1026	10025			28

- CLASS_2_PATIENT



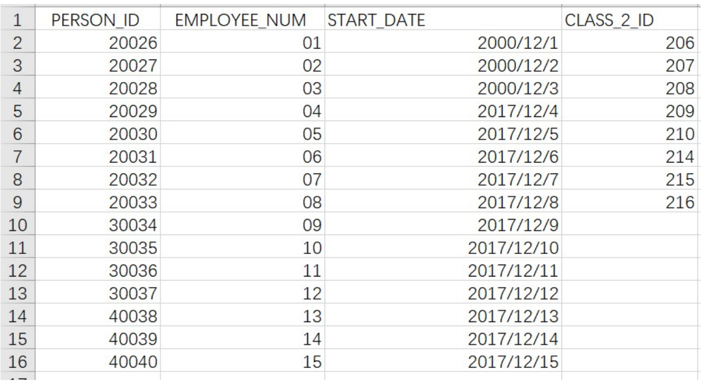
1	T_ID	Employee_Num	Class_2_ID	Date_Of_admitted	room_ID	DOC_ID
2	1		201	2017/1/2	1	26
3	1		202	2017/1/3	2	26
4	1		203	2017/1/4	3	26
5	1		204	2017/1/5	4	26
6	2		205	2017/1/6	5	26
7	2		206	2017/1/7	6	26
8	3		207	2017/1/8	7	26
9	4		208	2018/11/1	8	26
10	5		209	2018/11/2	9	26
11	6		210	2018/11/3	10	26
12	7		214	2018/12/1	11	27
13	8		215	2018/12/2	12	28
14	8		216	2017/12/30	13	29

- DOCTOR



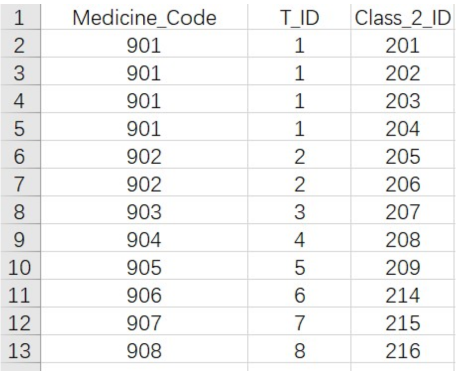
1	EMPLOYEE_NUM	DOCTOR_ID	DOC_TYPE
2	20026	26	p
3	20027	27	p
4	20028	28	p
5	20029	29	t
6	20030	30	t
7	20031	31	t
8	20032	32	t
9	20033	33	v

- EMPLOYEE



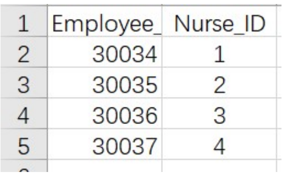
1	PERSON_ID	EMPLOYEE_NUM	START_DATE	CLASS_2_ID
2	20026	01	2000/12/1	206
3	20027	02	2000/12/2	207
4	20028	03	2000/12/3	208
5	20029	04	2017/12/4	209
6	20030	05	2017/12/5	210
7	20031	06	2017/12/6	214
8	20032	07	2017/12/7	215
9	20033	08	2017/12/8	216
10	30034	09	2017/12/9	
11	30035	10	2017/12/10	
12	30036	11	2017/12/11	
13	30037	12	2017/12/12	
14	40038	13	2017/12/13	
15	40039	14	2017/12/14	
16	40040	15	2017/12/15	

- MEDICAL_INFORMATION



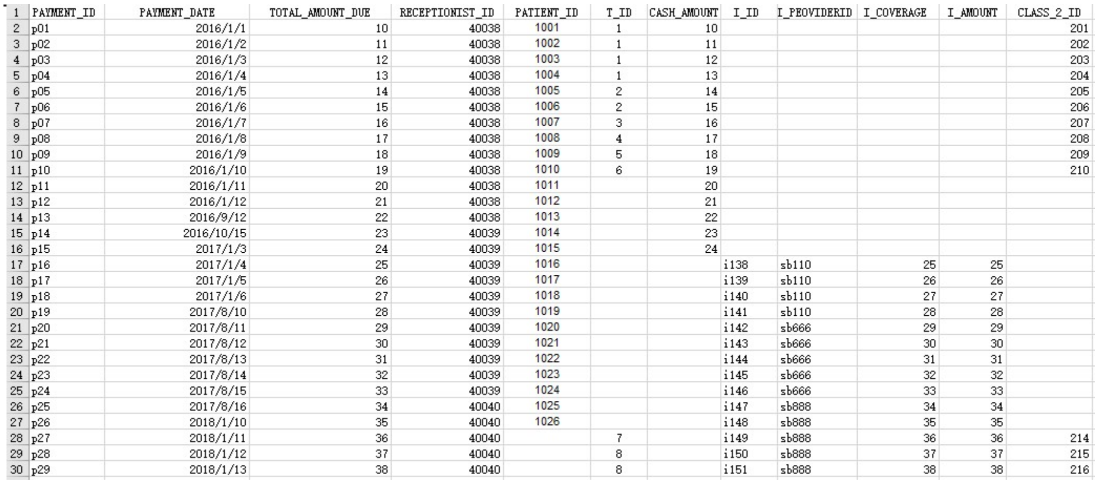
1	Medicine_Code	T_ID	Class_2_ID
2	901	1	201
3	901	1	202
4	901	1	203
5	901	1	204
6	902	2	205
7	902	2	206
8	903	3	207
9	904	4	208
10	905	5	209
11	906	6	214
12	907	7	215
13	908	8	216

- NURSE



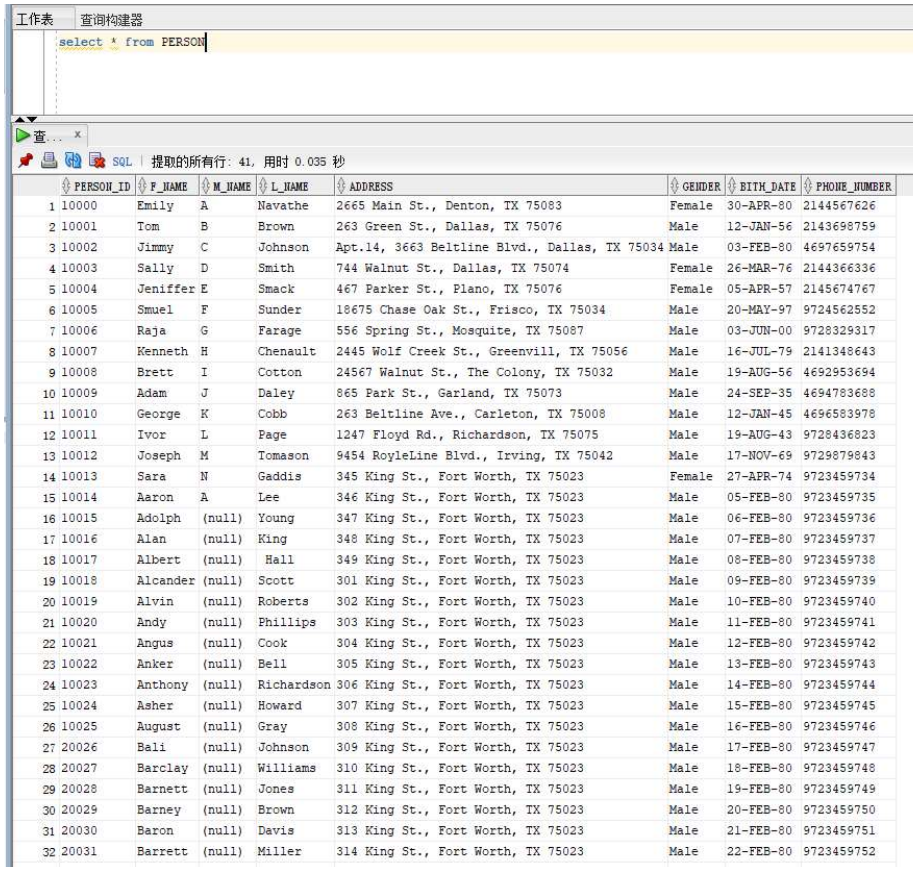
1	Employee_	Nurse_ID
2	30034	1
3	30035	2
4	30036	3
5	30037	4

- PYMENT



1	PAYMENT_ID	PAYMENT_DATE	TOTAL_AMOUNT_DUE	RECEPTIONIST_ID	PATIENT_ID	T_ID	CASH_AMOUNT	I_ID	I_PEOVIDERID	I_COVERAGE	I_AMOUNT	CLASS_2_ID
2	p01	2016/1/1	10	40038	1001	1	10					201
3	p02	2016/1/2	11	40038	1002	1	11					202
4	p03	2016/1/3	12	40038	1003	1	12					203
5	p04	2016/1/4	13	40038	1004	1	13					204
6	p05	2016/1/5	14	40038	1005	2	14					205
7	p06	2016/1/6	15	40038	1006	2	15					206
8	p07	2016/1/7	16	40038	1007	3	16					207
9	p08	2016/1/8	17	40038	1008	4	17					208
10	p09	2016/1/9	18	40038	1009	5	18					209
11	p10	2016/1/10	19	40038	1010	6	19					210
12	p11	2016/1/11	20	40038	1011		20					
13	p12	2016/1/12	21	40038	1012		21					
14	p13	2016/9/12	22	40038	1013		22					
15	p14	2016/10/15	23	40039	1014		23					
16	p15	2017/1/3	24	40039	1015		24					
17	p16	2017/1/4	25	40039	1016			i138	sb110	25	25	
18	p17	2017/1/5	26	40039	1017			i139	sb110	26	26	
19	p18	2017/1/6	27	40039	1018			i140	sb110	27	27	
20	p19	2017/8/10	28	40039	1019			i141	sb110	28	28	
21	p20	2017/8/11	29	40039	1020			i142	sb666	29	29	
22	p21	2017/8/12	30	40039	1021			i143	sb666	30	30	
23	p22	2017/8/13	31	40039	1022			i144	sb666	31	31	
24	p23	2017/8/14	32	40039	1023			i145	sb666	32	32	
25	p24	2017/8/15	33	40039	1024			i146	sb666	33	33	
26	p25	2017/8/16	34	40040	1025			i147	sb888	34	34	
27	p26	2018/1/10	35	40040	1026			i148	sb888	35	35	
28	p27	2018/1/11	36	40040		7		i149	sb888	36	36	214
29	p28	2018/1/12	37	40040		8		i150	sb888	37	37	215
30	p29	2018/1/13	38	40040		8		i151	sb888	38	38	216

- PERSON

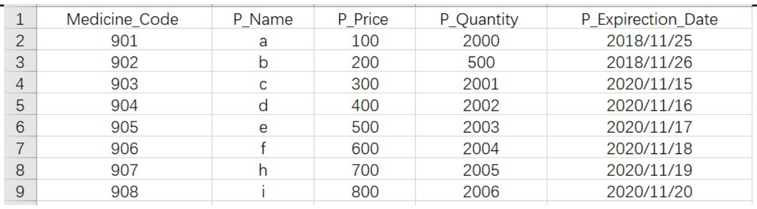


```
select * from PERSON
```

提取的所有行: 41, 用时 0.035 秒

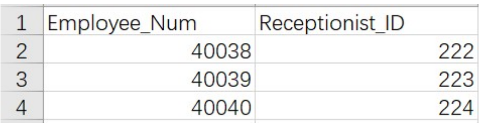
	PERSON_ID	F_NAME	M_NAME	L_NAME	ADDRESS	GENDER	BITH_DATE	PHONE_NUMBER
1	10000	Emily	A	Navathe	2665 Main St., Denton, TX 75083	Female	30-APR-80	2144567626
2	10001	Tom	B	Brown	263 Green St., Dallas, TX 75076	Male	12-JAN-56	2143698759
3	10002	Jimmy	C	Johnson	Apt.14, 3663 Beltline Blvd., Dallas, TX 75034	Male	03-FEB-80	4697659754
4	10003	Sally	D	Smith	744 Walnut St., Dallas, TX 75074	Female	26-MAR-76	2144366336
5	10004	Jeniffer	E	Smack	467 Parker St., Plano, TX 75076	Female	05-APR-57	2145674767
6	10005	Smuel	F	Sunder	18675 Chase Oak St., Frisco, TX 75034	Male	20-MAY-97	9724562552
7	10006	Raja	G	Farage	556 Spring St., Mosquite, TX 75087	Male	03-JUN-00	9728329317
8	10007	Kenneth	H	Chenault	2445 Wolf Creek St., Greenvill, TX 75056	Male	16-JUL-79	2141348643
9	10008	Brett	I	Cotton	24567 Walnut St., The Colony, TX 75032	Male	19-AUG-56	4692953694
10	10009	Adam	J	Daley	865 Park St., Garland, TX 75073	Male	24-SEP-35	4694783688
11	10010	George	K	Cobb	263 Beltline Ave., Carleton, TX 75008	Male	12-JAN-45	4696583978
12	10011	Ivor	L	Page	1247 Floyd Rd., Richardson, TX 75075	Male	19-AUG-43	9728436823
13	10012	Joseph	M	Tomason	9454 RoyleLine Blvd., Irving, TX 75042	Male	17-NOV-69	9729879843
14	10013	Sara	N	Gaddis	345 King St., Fort Worth, TX 75023	Female	27-APR-74	9723459734
15	10014	Aaron	A	Lee	346 King St., Fort Worth, TX 75023	Male	05-FEB-80	9723459735
16	10015	Adolph	(null)	Young	347 King St., Fort Worth, TX 75023	Male	06-FEB-80	9723459736
17	10016	Alan	(null)	King	348 King St., Fort Worth, TX 75023	Male	07-FEB-80	9723459737
18	10017	Albert	(null)	Hall	349 King St., Fort Worth, TX 75023	Male	08-FEB-80	9723459738
19	10018	Alcander	(null)	Scott	301 King St., Fort Worth, TX 75023	Male	09-FEB-80	9723459739
20	10019	Alvin	(null)	Roberts	302 King St., Fort Worth, TX 75023	Male	10-FEB-80	9723459740
21	10020	Andy	(null)	Phillips	303 King St., Fort Worth, TX 75023	Male	11-FEB-80	9723459741
22	10021	Angus	(null)	Cook	304 King St., Fort Worth, TX 75023	Male	12-FEB-80	9723459742
23	10022	Anker	(null)	Bell	305 King St., Fort Worth, TX 75023	Male	13-FEB-80	9723459743
24	10023	Anthony	(null)	Richardson	306 King St., Fort Worth, TX 75023	Male	14-FEB-80	9723459744
25	10024	Asher	(null)	Howard	307 King St., Fort Worth, TX 75023	Male	15-FEB-80	9723459745
26	10025	August	(null)	Gray	308 King St., Fort Worth, TX 75023	Male	16-FEB-80	9723459746
27	20026	Bali	(null)	Johnson	309 King St., Fort Worth, TX 75023	Male	17-FEB-80	9723459747
28	20027	Barclay	(null)	Williams	310 King St., Fort Worth, TX 75023	Male	18-FEB-80	9723459748
29	20028	Barnett	(null)	Jones	311 King St., Fort Worth, TX 75023	Male	19-FEB-80	9723459749
30	20029	Barney	(null)	Brown	312 King St., Fort Worth, TX 75023	Male	20-FEB-80	9723459750
31	20030	Baron	(null)	Davis	313 King St., Fort Worth, TX 75023	Male	21-FEB-80	9723459751
32	20031	Barrett	(null)	Miller	314 King St., Fort Worth, TX 75023	Male	22-FEB-80	9723459752

- PHARMACY



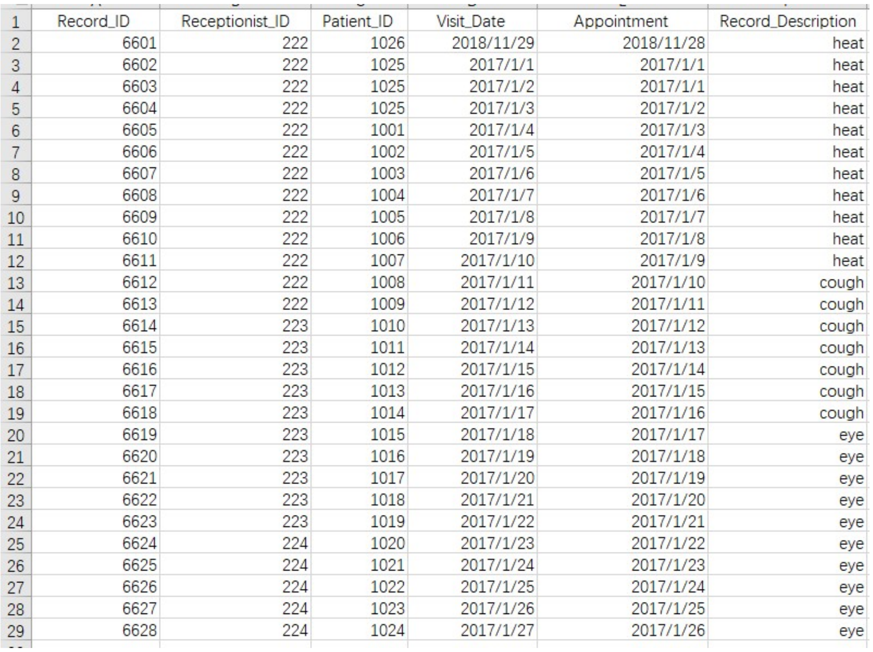
1	Medicine_Code	P_Name	P_Price	P_Quantity	P_Expirection_Date
2	901	a	100	2000	2018/11/25
3	902	b	200	500	2018/11/26
4	903	c	300	2001	2020/11/15
5	904	d	400	2002	2020/11/16
6	905	e	500	2003	2020/11/17
7	906	f	600	2004	2020/11/18
8	907	h	700	2005	2020/11/19
9	908	i	800	2006	2020/11/20

- RECEPTIONIST



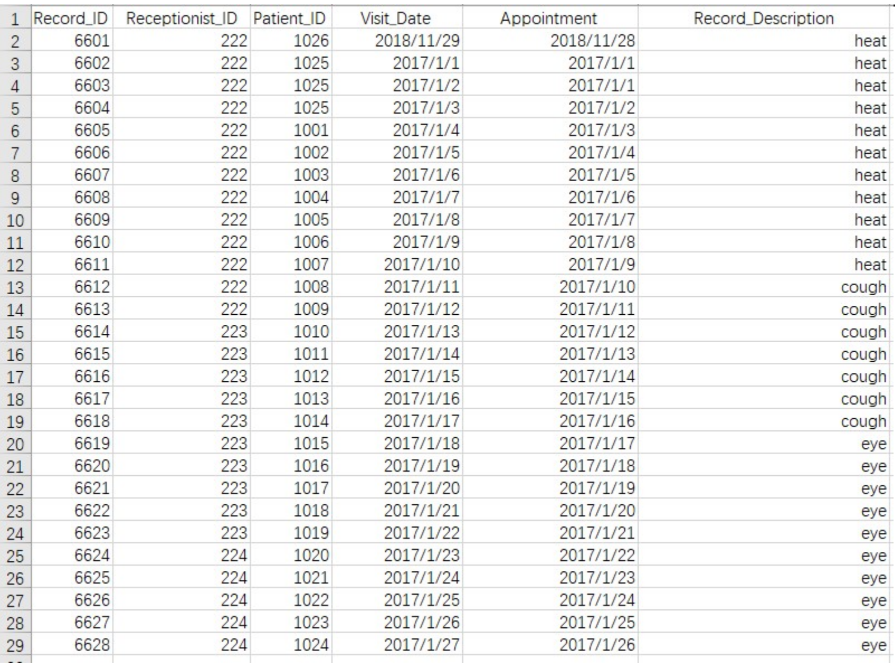
1	Employee_Num	Receptionist_ID
2	40038	222
3	40039	223
4	40040	224

- RECORDS



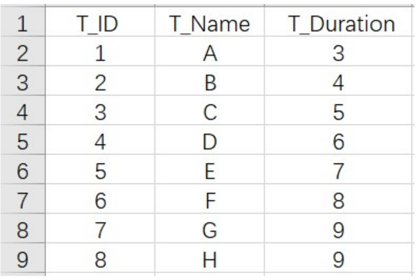
1	Record_ID	Receptionist_ID	Patient_ID	Visit_Date	Appointment	Record_Description
2	6601	222	1026	2018/11/29	2018/11/28	heat
3	6602	222	1025	2017/1/1	2017/1/1	heat
4	6603	222	1025	2017/1/2	2017/1/1	heat
5	6604	222	1025	2017/1/3	2017/1/2	heat
6	6605	222	1001	2017/1/4	2017/1/3	heat
7	6606	222	1002	2017/1/5	2017/1/4	heat
8	6607	222	1003	2017/1/6	2017/1/5	heat
9	6608	222	1004	2017/1/7	2017/1/6	heat
10	6609	222	1005	2017/1/8	2017/1/7	heat
11	6610	222	1006	2017/1/9	2017/1/8	heat
12	6611	222	1007	2017/1/10	2017/1/9	heat
13	6612	222	1008	2017/1/11	2017/1/10	cough
14	6613	222	1009	2017/1/12	2017/1/11	cough
15	6614	223	1010	2017/1/13	2017/1/12	cough
16	6615	223	1011	2017/1/14	2017/1/13	cough
17	6616	223	1012	2017/1/15	2017/1/14	cough
18	6617	223	1013	2017/1/16	2017/1/15	cough
19	6618	223	1014	2017/1/17	2017/1/16	cough
20	6619	223	1015	2017/1/18	2017/1/17	eye
21	6620	223	1016	2017/1/19	2017/1/18	eye
22	6621	223	1017	2017/1/20	2017/1/19	eye
23	6622	223	1018	2017/1/21	2017/1/20	eye
24	6623	223	1019	2017/1/22	2017/1/21	eye
25	6624	224	1020	2017/1/23	2017/1/22	eye
26	6625	224	1021	2017/1/24	2017/1/23	eye
27	6626	224	1022	2017/1/25	2017/1/24	eye
28	6627	224	1023	2017/1/26	2017/1/25	eye
29	6628	224	1024	2017/1/27	2017/1/26	eye

- ROOM



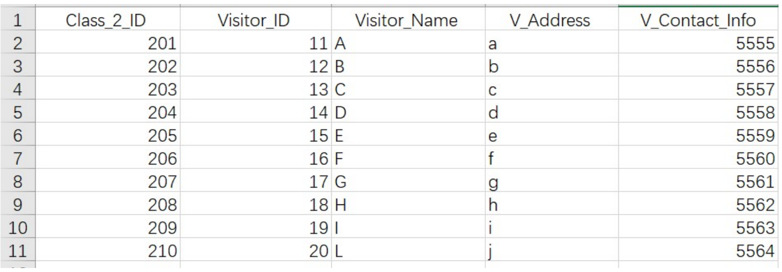
1	Record_ID	Receptionist_ID	Patient_ID	Visit_Date	Appointment	Record_Description
2	6601	222	1026	2018/11/29	2018/11/28	heat
3	6602	222	1025	2017/1/1	2017/1/1	heat
4	6603	222	1025	2017/1/2	2017/1/1	heat
5	6604	222	1025	2017/1/3	2017/1/2	heat
6	6605	222	1001	2017/1/4	2017/1/3	heat
7	6606	222	1002	2017/1/5	2017/1/4	heat
8	6607	222	1003	2017/1/6	2017/1/5	heat
9	6608	222	1004	2017/1/7	2017/1/6	heat
10	6609	222	1005	2017/1/8	2017/1/7	heat
11	6610	222	1006	2017/1/9	2017/1/8	heat
12	6611	222	1007	2017/1/10	2017/1/9	heat
13	6612	222	1008	2017/1/11	2017/1/10	cough
14	6613	222	1009	2017/1/12	2017/1/11	cough
15	6614	223	1010	2017/1/13	2017/1/12	cough
16	6615	223	1011	2017/1/14	2017/1/13	cough
17	6616	223	1012	2017/1/15	2017/1/14	cough
18	6617	223	1013	2017/1/16	2017/1/15	cough
19	6618	223	1014	2017/1/17	2017/1/16	cough
20	6619	223	1015	2017/1/18	2017/1/17	eye
21	6620	223	1016	2017/1/19	2017/1/18	eye
22	6621	223	1017	2017/1/20	2017/1/19	eye
23	6622	223	1018	2017/1/21	2017/1/20	eye
24	6623	223	1019	2017/1/22	2017/1/21	eye
25	6624	224	1020	2017/1/23	2017/1/22	eye
26	6625	224	1021	2017/1/24	2017/1/23	eye
27	6626	224	1022	2017/1/25	2017/1/24	eye
28	6627	224	1023	2017/1/26	2017/1/25	eye
29	6628	224	1024	2017/1/27	2017/1/26	eye

- TREATMENT



1	T_ID	T_Name	T_Duration
2	1	A	3
3	2	B	4
4	3	C	5
5	4	D	6
6	5	E	7
7	6	F	8
8	7	G	9
9	8	H	9

- VISITOR_LOG



1	Class_2_ID	Visitor_ID	Visitor_Name	V_Address	V_Contact_Info
2	201	11	A	a	5555
3	202	12	B	b	5556
4	203	13	C	c	5557
5	204	14	D	d	5558
6	205	15	E	e	5559
7	206	16	F	f	5560
8	207	17	G	g	5561
9	208	18	H	h	5562
10	209	19	I	i	5563
11	210	20	L	j	5564

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

Use the Create View statement to create the following views:

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

```
CREATE VIEW TopDoctor as
SELECT P.FName,P.LName,E.Start_Date ,Doc_ID
 FROM  Person as P,Employee E, Doctor D
WHERE E.Person_ID =P.Person_ID and D.Employee_Num = E.Employee_Num and E.Employee_Num in
(
(SELECT count(*)
FROM class _1_patient as c1p,
Group by c1p.Employee_Num
HAVING count(*) > 5)
and
(SELECT count(*)
  FROM Class_2_Patient as c2p
Group by c2p.Employee_Num
HAVING count(*)>10)
);
```

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 VIEW TopTreatment as
SELECT T.name, T.T_ID
FROM Treatment T,Payment P,
WHERE P.T_ID=T.T_ID and T.T_ID exist
(SELECT P.T_ID ,max(count(*))
FROM Payment P
Group by P.T_ID
)
```

3 ReorderMeds- 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 VIEW ReorderMeds as
SELECT  Medicine_Code
FROM  Pharmacy
WHERE ((to_char(sysdate,'mm') - (to_char(P_Expirection_Date,'mm')))<=1 or (P_Quantity<1000)
```

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 VIEW PotentialPatient as
SELECT F_name, L_name, Phone_Number,Patient_ID,Person_ID
```

```
FROM Record r, Person p,Class_1_Patient c1p
WHERE r.Patient_ID =c1p.Patient_ID and c1p.person_ID =p.Person_ID and c1p.class_2_ID = Null and p.Patient_ID in
(
SELECT  r.patient_ID,count(*)
Form    Record
Group by r.patient_ID
HAVING count(*) >3
)
```

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 VIEW PotentialPatient as
View5:
CREATE VIEW FrequentIssues AS

SELECT T.T_Name ,count(*)
FROM Treatment T
Group by distinct T.T_name
Order by count(*) desc
limit 1
```

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

Now we give out the SQL Queries for all questions listed in **Question e** as follows:

1.For each Doctor class, list the start date and specialization of the doctor.

```
SELECT E.start_date,D.type
FROM Doctor D,Employee E
WHERE D.Employee_Num = E.Employee_Num
```

2. Find the names of employees who have been admitted to the hospital within 3 months of joining.

```
SELECT P.F_name ,P.L_name
FROM Person p,Employee e
WHERE p.person_ID = e.person_ID and e.start_date =(sysdate -e.start_date)<90)
```

3. Find the age and class (trainee, visiting or permanent) of top 5 doctors in the hospital.

```
SELECT F_name,L_name,((to_char(sysdate,'yyyy'))to_char(E.birth_date,'yyyy'))) as age,Doc_type
FROM TopDoctor as T, Person P, Employee E, Doctor D
WHERE T.Doc_ID=D.Doc_ID
```

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

```
SELECT ph.name
FROM Medical_infomation me,Pharmacy as ph,TopTreatment as top
WHERE me.medicine_code = ph.medicine_code and top.T_ID = me.T_ID
```

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 F_name ,L_name
FROM Person p ,Employee e ,Doctor d
WHERE p.Person_ID =e.Person_ID and d.Employee_Num=e.Employee.Num and (D.doc_ID , D.Employee_Num) in
(
SELECT d.Doc_ID ,d.Employee_Num,
FROM class_2_patient c2p, class_1patient c1p ,payment p
WHERE c1p.class_2_ID=c2p.class_2_ID and c2p.doc_ID=D.doc_ID and     ((to_char(sysdate,'mm') - (to_char(Date_of_admitted,'mm'))))>5 and    ((to_char(sysdate,'mm') - (to_char(payment_date,'mm'))))>5
)
```

6. Find the total number of patients who have paid completely using insurance and the name of the insurance provider.

```
SELECT I_Provider_name ,count(*)
FROM Class_1_Patient c1p, Class_2_Patient c2p,Payment p
WHERE c1p.Class_2_ID= c2p.Class_2_ID and p.Patient_ID = c1p.Patient_ID and p.I_ID <> null
Group by p.I_Provider_name
```

7.Find the most occupied room in the hospital and the duration of the stay.

```
SELECT r.Room_ID ,r.duration
FROM Room r
WHERE duration = max(r.duration)
```

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

```
SELECT r.Record_Description, to_char(Visit_Date,'yyyy') as year
FROM Records r
```

```
WHERE year in
(
SELECT year ,r.Patient_ID, max(count(*))
FROM Records r
Group by r.Patient_ID
)
```

9 Find the duration of the treatment that is provided the least to patients.

```
SELECT Name
FROM TREATEMENT
WHERE duration=
(
    SELECT Min(Duration)
    FROM TREATEMENT
)
```

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

```
SELECT COUNT(C.Class_2_ID) as COUNTNUMBER
FROM Class_2_Patient C
WHERE C.Date_Of_Admitted>(
    SELECT MAX(Start_date)
    FROM DOCTOR,EMPLOYEE
    WHERE DOCTOR.Employee_Num=EMPLOYEE.Employee_Num
)
```

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 DISTINCT t3.*
FROM Class_1_Patient t1 LEFT JOIN Class_2_Patient t2 on t2.Class_2_ID=t1.Class_2_ID
                        LEFT JOIN RECORDS t3 on t1.Patient_ID=t3.Patient_ID
WHERE (to_char(t2.Date_Of_admitted,'yyyymmdd') - (to_char(t3.Visit_Date,'yyyymmdd'))<7
```

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

```
SELECT sum(Total_Amount_Due)
FROM Payment
WHERE Payment_Date = to_char(Payment_Date,'2017')
Group by (Payment_ID)
```

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.

```sql
SELECT F_name,L_name
FROM Person p,Doctor d, Class_1_Patient as c1p, Employee e
WHERE p.person_id =e.person_id and e.Employee_Num = d.Employee_Num and c1p.Doc_ID =d.Doc_ID and c1p.Class_2_ID = null and (c1p.Patient_ID) exist
(
SELECT r.Patient_ID,count(Visit_Date)
FROM Records r
WHERE count(Visit_Date)=1
Group by r.patient_id
)
```

14. Find the name and age of the potential patients in the hospital.

```sql
SELECT   F_name, L_name,,((to_char(sysdate,'yyyy'))-to_char(E.birth_date,'yyyy'))) as age
FROM    PotentialPatient po,Person p
WHERE   po.Person_ID= p.Person_ID
```

5. Dependency Diagram

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

5.1 Hospital Personnel

There is only one attribute in the left-hand side of the functional dependencies, which is the key of relational schema Hospital Personnel, Person_ID. Therefore, every other attribute of this relational schema is functionally dependent on Person_ID. The dependency diagram is shown as Figure 1.

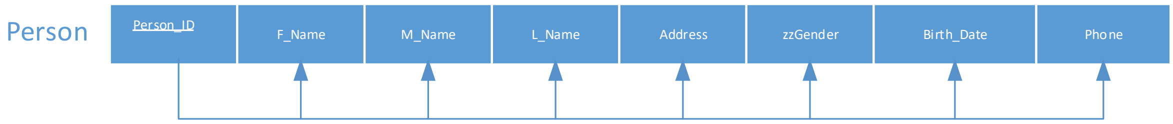


Figure 1. Dependency Diagram of Hospital Personnel

5.2 Class_1_Patient

There is only one attribute in the left-hand side of the functional dependencies, which is the key of relational schema Class_1_Patient, Patient_ID. Therefore, every other attribute of this relational schema is functionally dependent on Patient_ID. The dependency diagram is shown as Figure 2.

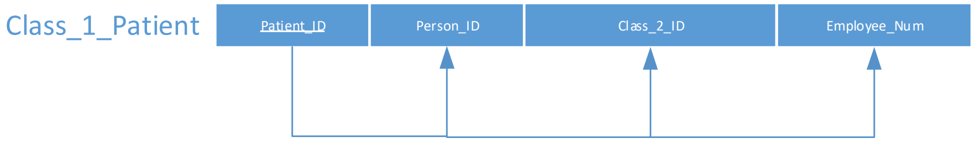


Figure 2. Dependency Diagram of Class_1_Patient

5.3 Hospital Employee

There is only one attribute in the left-hand side of the functional dependencies, which is the key of relational schema Hospital Employee,Employee_num. Therefore, every other attribute of this relational schema is functionally dependent on Employee_num. The dependency diagram is shown as Figure 3.

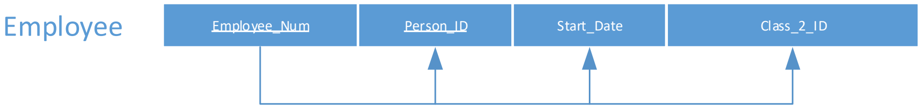


Figure 3. Dependency Diagram of Employee

5.4 Hospital Doctor

There is only one attribute in the left-hand side of the functional dependencies, which is DOC_ID. The dependency diagram is shown as Figure 4.

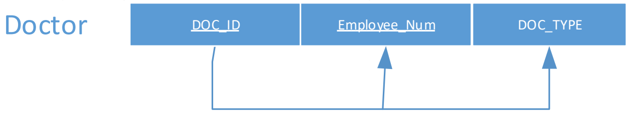


Figure 2. Dependency Diagram of Doctor

5.5 Relation Access

There are two attribute in the left-hand side of the functional dependencies, which are Medicine code and Treatment ID. The dependency diagram is shown as Figure 5.

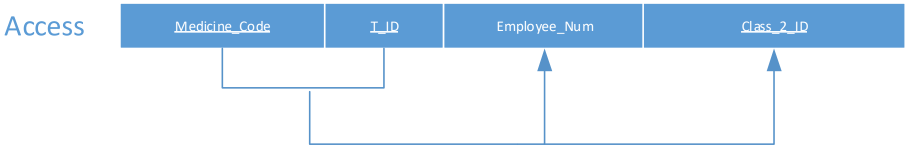


Figure 5. Dependency Diagram of Access

5.6 Hospital Nurse

The dependency diagram is shown as Figure 6.

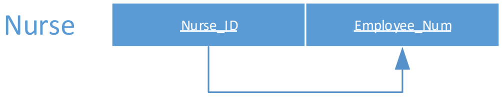


Figure 6. Dependency Diagram of Nurse

5.7 Hospital Room

The dependency diagram is shown as Figure 7.

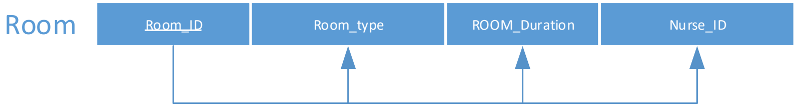


Figure 7. Dependency Diagram of Room

5.8 Hospital Receptionist

The dependency diagram is shown as Figure 8.

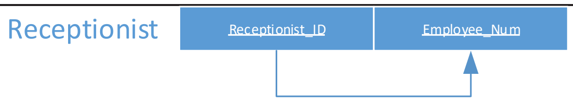


Figure 8. Dependency Diagram Receptionist

5.9 Patient Records

The dependency diagram is shown as Figure 9.

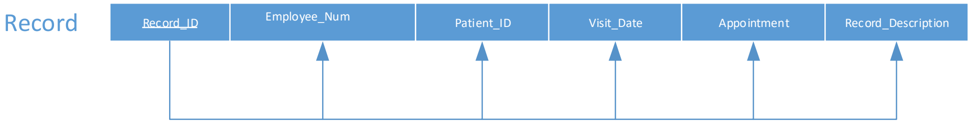


Figure 9. Dependency Diagram of Records

5.10 Payment Information

The dependency diagram is shown as Figure 10.

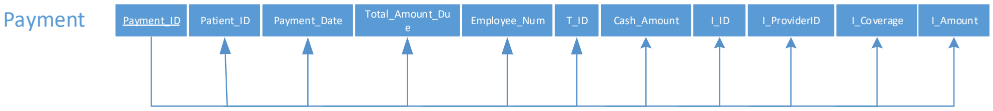


Figure 10. Dependency Diagram of Payment

5.11 Medical Information

The medicine code is depend on both T_ID and Class_2_ID, thus the primary of this relation is T_ID together with Class_2_ID. The dependency diagram is shown as Figure 11.

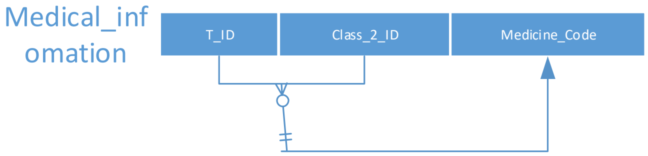


Figure 11. Dependency Diagram

5.12 Hospital Pharmacy

The dependency diagram is shown as Figure 12.

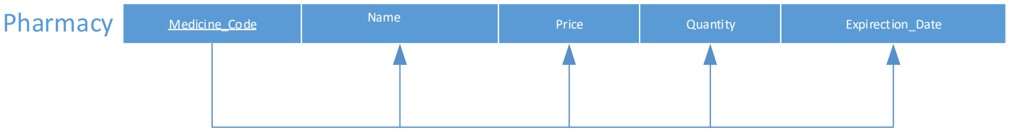


Figure 12. Dependency Diagram

5.13 Hospital Treatment

The dependency diagram is shown as Figure 13.

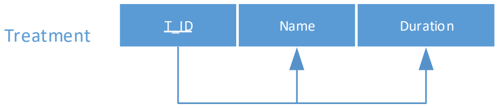


Figure 13. Dependency Diagram

5.14 Class 2 Patient

The dependency diagram is shown as Figure 14.

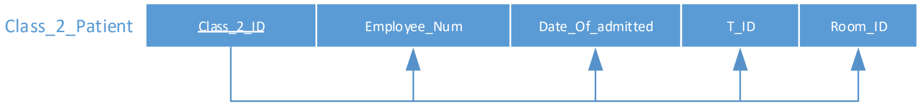


Figure 14. Dependency Diagram

5.15 Visitor Log

The dependency diagram is shown as Figure 15.

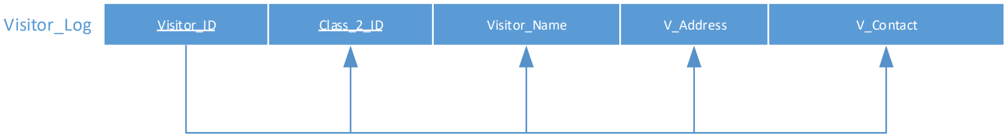
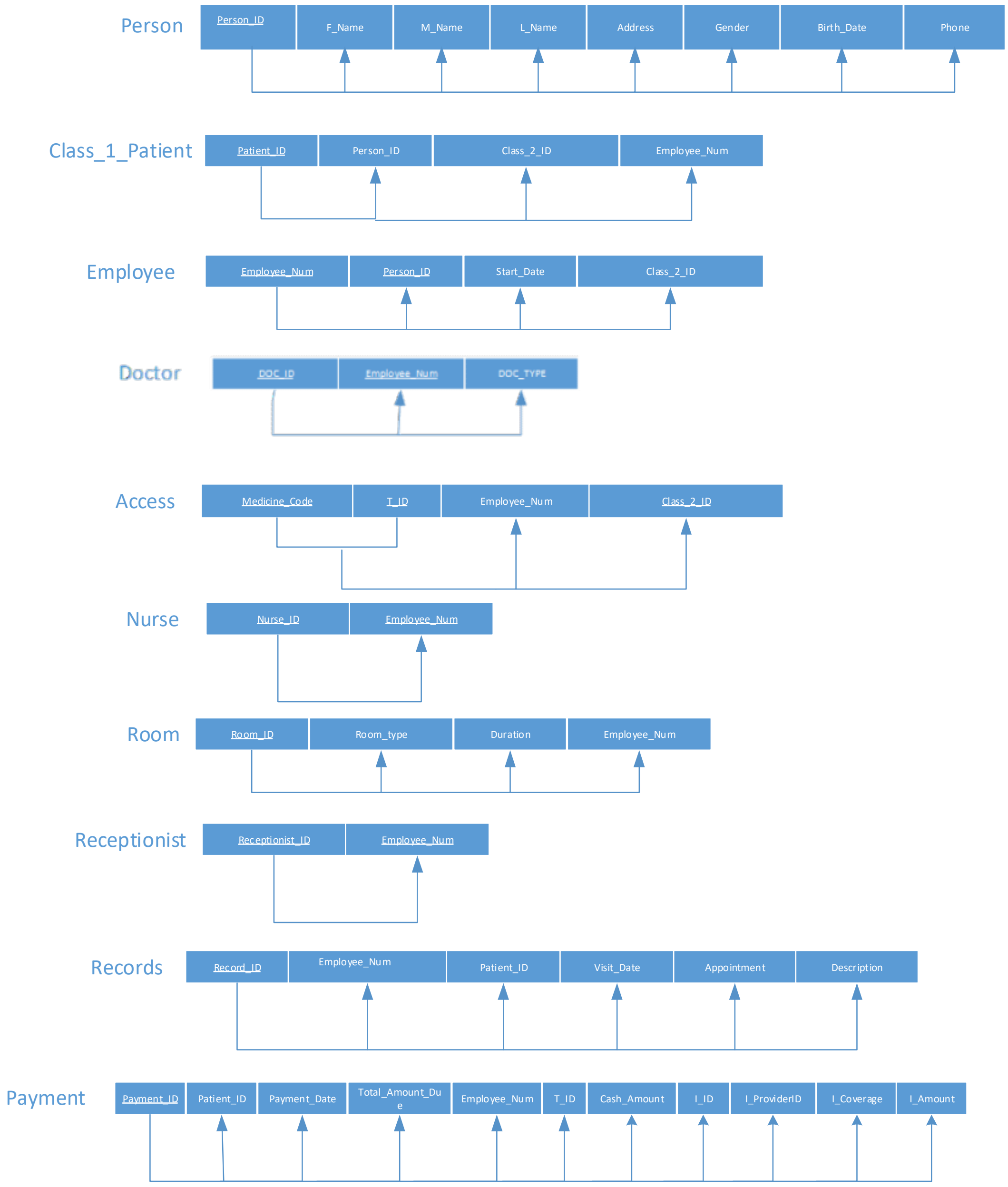


Figure 15. Dependency Diagram of Hospital Personnel

5.16 Final Results

After drawing the dependency diagrams one after another, Figure 16 shows the final results for the whole database including the ones who do not have any functional dependencies.


Person
Person_ID
F_Name
M_Name
L_Name
Address
Gender
Birth_Date
Phone
Class_1_Patient
Patient_ID
Person_ID
Class_2_ID
Employee_Num
Employee
Employee_Num
Person_ID
Start_Date
Class_2_ID
Doctor
DOC_ID
Employee_Num
DOC_TYPE
Access
Medicine_Code
T_ID
Employee_Num
Class_2_ID
Nurse
Nurse_ID
Employee_Num
Room
Room_ID
Room_type
Duration
Employee_Num
Receptionist
Receptionist_ID
Employee_Num
Records
Record_ID
Employee_Num
Patient_ID
Visit_Date
Appointment
Description
Payment
Payment_ID
Patient_ID
Payment_Date
Total_Amount_Due
Employee_Num
T_ID
Cash_Amount
I_ID
I_ProviderID
I_Coverage
I_Amount

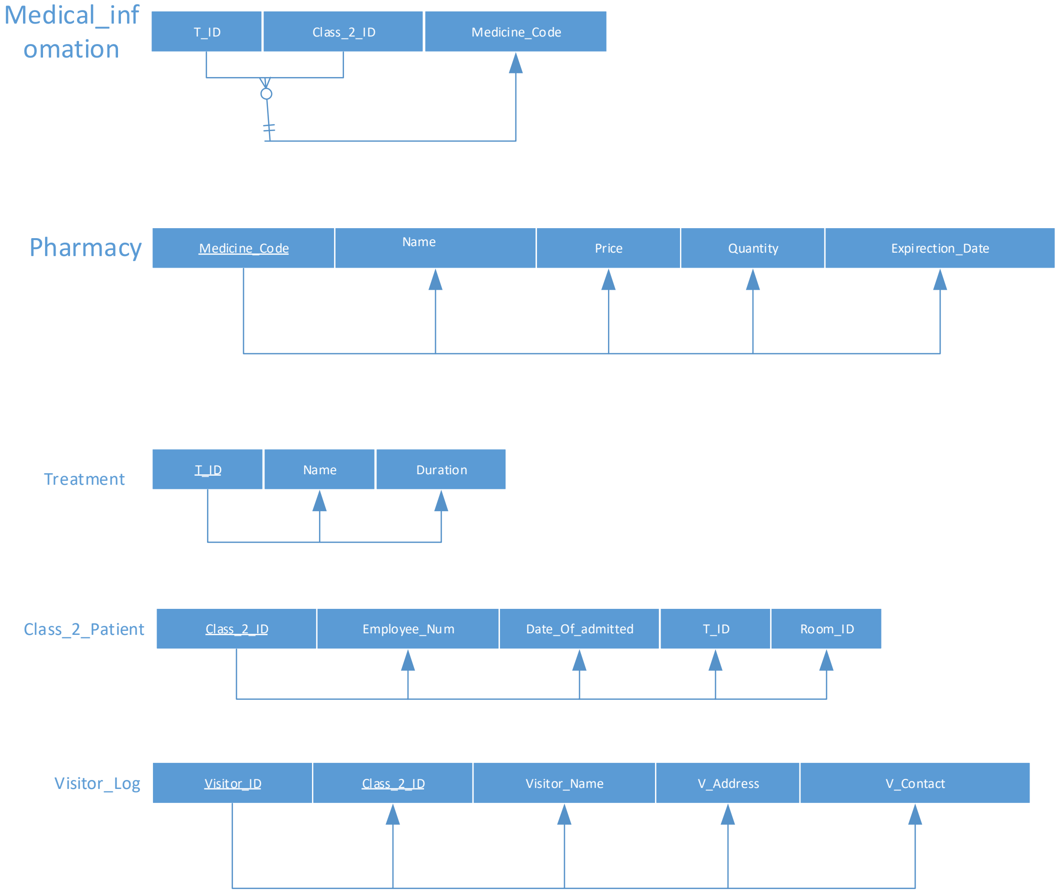


Figure 16. Whole Dependency Diagram for Dallas Care Database

6. Conclusion

In this final report we summarized all the necessary descriptions and solutions for Dallas Care 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.