DENTAL CLINIC DATA MANAGEMENT

SQL PROJECT

Madhav Choudhary

- LinkedIn - Github

PROJECT FILES

Problem-Statement:

A Dentists' Polyclinic has various dental departments such as General Dentistry, Periodontist and Endodontist. The patients can go to all these Departments.

The polyclinic has many types of doctors depending on the dental departments.

a) General dentistry: Cavities, Missing Teeth, Mobile Teeth
b) Endodontist: Root canals
c) Periodontist: Gums

The Doctor's work from Monday to Saturday, from 4:00pm-8:30pm. The appointments are given according to these timings. Salary of the doctors along with the number of years they have been working in the clinic is stored to keep a track of their performance and the number of years they have been serving , based on which the salary raise and promotions can be done.

When a customer arrives in the dental clinic, we identify them based on their patient id which is unique for each patient . We ask for all the details. i.e. the name of the patient, disease, sex, phone number and we even feed their arrival time and arrival date into our system based upon which we decide whether the doctor is available or not, if the patient registers within the specified time slot (Monday-Saturday,4:00pm-8:30pm) then appointment can be made otherwise notification will be displayed asking the patient to come within the time slot.

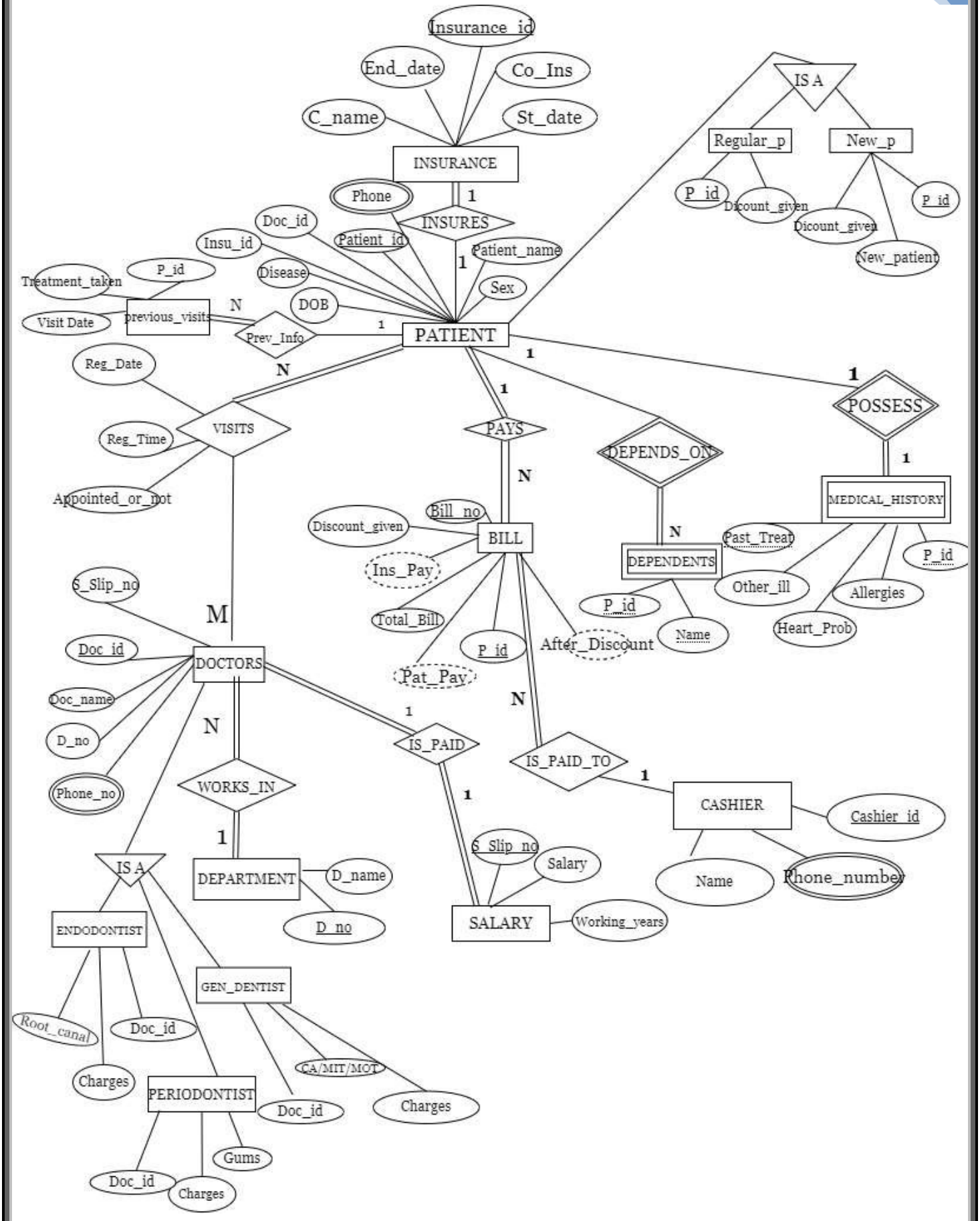
If a customer has health insurance, which is nowadays very important as the medical costs are very high in the country and without insurance, a large portion of population can't afford the healthcare, we store the insurance ID (unique for each customer), company name, start date, end date and Co-Insurance. Co-Insurance is a percentage amount that insurance company pays for a medicinal purchase or treatment. In case the insurance of any patient expires then he needs to pay all the amount of the treatment by himself. The previous record of the patients (On what dates they have visited the clinic in the past based upon which the system automatically calculates how many times they have visited to the clinic, what all treatments did they take) are also stored in the database of the clinic .

The patients can be of two types, a regular patient and a new patient. Patients who came to the clinic 2 or more than 2 times should be considered as regular patients and they should be given a 10% discount (regular patient). A regular patient's history is accessible . If a new patient arrives then he gets the first check-up free. The patients' medical history such as heart problems, allergies, other illnesses are stored within the database so that the doctor can give a treatment accordingly. A patient can come to the polyclinic with a dependent. The information about the dependent of the patient is stored in the database as well.

Once the treatment has been completed, a bill is generated by the system. This bill is handed over to the customer by the cashier and it contains patient id, insurance information, total charges along with the treatment taken, Price after giving discount (in case of regular and new patients), Price to paid by the patient and the insurance company. The breakdown should be automatically calculated by the system and should be stored in the table (TOTAL_BILL).

Lastly the details of the cashier should also be stored in the system .

E-R DIAGRAM


Insurance_id
End_date
Co_Ins
C_name
St_date
INSURANCE
Phone
1
INSURES
Doc_id
Insu_id
Patient_id
Patient_name
1
Sex
Disease
DOB
P_id
Treatment_taken
Visit Date
previous_visits
N
Prev_Info
1
PATIENT
Reg_Date
N
1
1
1
POSSESS
VISITS
Reg_Time
Appointed_or_not
PAYS
N
DEPENDS_ON
1
MEDICAL_HISTORY
Discount_given
Bill_no
BILL
N
DEPENDENTS
Past_Treat
P_id
Ins_Pay
Other_ill
Allergies
S_Slip_no
M
P_id
Name
Heart_Prob
Total_Bill
P_id
After_Discount
Doc_id
DOCTORS
Pat_Pay
Doc_name
N
1
N
D_no
IS_PAID
IS_PAID_TO
1
Phone_no
WORKS_IN
1
CASHIER
Cashier_id
1
S_Slip_no
IS A
DEPARTMENT
D_name
Salary
Name
Phone_number
ENDODONTIST
D_no
SALARY
Working_years
GEN_DENTIST
Root_canal
Doc_id
CA/MIT/MOT
Charges
PERIODONTIST
Doc_id
Charges
Gums
Doc_id
Charges
IS A
Regular_p
New_p
P_id
Dicount_given
Dicount_given
P_id
New_patient

SCHEMA

INSURANCE

Insurance_id	Company_name	Start_date	End_date	Co_Insurnace

PRIMARY KEY: Insurance_id

PATIENT1

Patient_id	Polyclinic_name	Patient_name	dob	Insurance_id	Sex	Disease	Doc_id
Registration_time	Registration_date						

PRIMARY KEY: Patient_id

UNIQUE: Patient_name

FOREIGN KEY: Patient1(doc_id) references doctor_info(doc_id)

Patient1(insurance_id) references insurance(insurance_id)

PATIENT_PHONE

Patient_id	Phone_number

FOREIGN KEY : Patient_phone(patient_id) references patient1(patient_id)

VISITS

Patient_id	Patient_name	registration_time	registration_date	Appointed_or_not

PRIMARY KEY: PATIENT_ID

FOREIGN KEY: Visits (patient_name) references patient1(patient_name)

Visits (patient_id) references patient1 (patient_id)

PREVIOUS_VISITS

Patient_id	visits	prev_treatment_taken_from_this_clinic

FOREIGN KEY: Previous_visits (patient_id) references patient1(patient_id)

NEW PATIENT

Patient_id	Patient_name	Insurance_id	New_patient	Discount_given

PRIMARY KEY: Patient_id

FOREIGN KEY: new_patients (patient_id) references patient1(patient_id);

New_patients (patient_name) references patient1(patient_name)

New_patients (insurance_id) references patient1(insurance_id)

REGULAR PATIENT

Patient_id	Patient_name	Insurance_id	Discount_given

PRIMARY KEY: Patient_id

FOREIGN KEY: regular_patients (patient_id) references patient1(patient_id);

regular_patients (patient_name) references patient1(patient_name)

regular_patients (insurance_id) references patient1(insurance_id)

DOCTOR_INFO

Doc_id	Salary_slipno	Doc_name	Dep_no	Dep_name

PRIMARY KEY: Doc_id

UNIQUE: Salary_slipno

FOREIGN KEY: Doctor_info (Dep_no) references department(Dep_no)

Doctor_info references department(dep_name)

Doctor_info (salary_slipno) references doc_salary (salary_slipno)

DOCTOR_PHONE

Doc_id	Phone_number

FOREIGN KEY: DOCTOR_PHONE (doc_id) references doctor_info(doc_id)

DOCTOR_SALARY

Salary_slipno	Salary	Number_of_years_working

PRIMARY KEY: Salary_slipno

DEPARTMENT

dep_no	Dep_name

PRIMARY KEY: dep_no

UNIQUE: Dep_name

ENDODONTIST

Doc_id	Root_canal	Charges

FOREIGN KEY: Endodontist (doc_id) references doctor_info(doc_id)

PERIODONTIST

Doc_id	Gums	Price

FOREIGN KEY: periodontist (doc_id) references doctor_info(doc_id)

GEN_DENTIST

Doc_id	cavities_OR_missing_teeth_OR_mobile_teeth	Price

FOREIGN KEY: gen_dentist (doc_id) references doctor_info(doc_id)

TOTAL_BILL

Bill_no	Patient_id	Total_charge	After_discount	Money_insurance	Patient_pay	Cashier_id

PRIMARY KEY: bill_no,patient_id

FOREIGN KEY: TOTAL_BILL (cashier_id) references cashier(cashier_id)

TOTAL_BILL (Insurance_id) references patient1(insurance_id)

TOTAL_BILL (patient_id) references patient1(patient_id)

TOTAL_BILL (patient_name) references patient1(patient_name)

DEPENDENTS

Dependent_name	Patient_id	Phone_number

PRIMARY KEY: Dependent_name,patient_id

FOREIGN KEY: Dependents(patient_id) references patient1(patient_id)

MEDIC_HISTORY

Patient_id	Past_treatment	Allergies	Pain_tooth	Heart_probs	Other_illness

PRIMARY KEY: Patient_id,past_treatment

FOREIGN KEY: Medic_history (patient_id) references patient1(patient_id)

CASHIER

Cashier_id	Name	Phone_number	Salary

PRIMARY KEY: Cashier_id

CASHIER_PHONE

Cashier_id	Phone_number

Foreign key: Cashier_phone(cashier_id) references cashier(cashier_id)

POPULATED TABLES AND TABLE DESCRIPTION

Insurance

Field	Type	Null	Key	Default	Extra
insurance_id	int	NO	PRI	NULL	
company_name	varchar(50)	NO	MUL	NULL	
start_date	date	NO		NULL	
end_date	date	NO		NULL	
co_insurance	decimal(5,2)	YES		NULL	

insurance_id	company_name	start_date	end_date	co_insurance
101	National Insurance Co.Ltd	2011-03-12	2020-04-10	55.00
102	Go digital General Insurance	2011-09-12	2023-04-10	40.00
103	HDFC ERGO General insurance	2010-06-01	2024-05-07	60.00
104	HDFC ERGO General insurance	2010-06-01	2024-05-07	60.00
105	National Insurance Co.Ltd	2008-03-09	2022-09-23	30.00
106	National Insurance Co.Ltd	2008-03-09	2022-09-23	30.00
NULL	NULL	NULL	NULL	NULL

Patient1

Field	Type	Null	Key	Default	Extra
patient_id	int	NO	PRI	NULL	
polyclinic_name	varchar(20)	NO		NULL	
patient_name	varchar(20)	NO	UNI	NULL	
dob	date	NO		NULL	
insurance_id	int	YES	MUL	NULL	
sex	char(4)	NO		NULL	
Problem_or_Disease	varchar(50)	NO		NULL	
Dno	int	NO		NULL	
doc_id	int	NO	MUL	NULL	
registration_time	time	NO		NULL	
registration_date	date	NO		NULL	

patient_id	polyclinic_name	patient_name	dob	insurance_id	sex	Problem_or_Disease	Dno	doc_id	registration_time	registration_date
1	Dental Polyclinic	Mr.Smith	1967-03-25	101	M	Soft tissue Inflammation	1	100	17:00:00	2022-03-19
2	Dental Polyclinic	Mr.Andrews	1978-02-04	102	M	Gum Disease	2	300	14:00:00	2022-03-20
3	Dental Polyclinic	Mrs.Rodriguez	1987-07-28	103	F	Deep Decay	1	200	17:00:00	2022-03-21
4	Dental Polyclinic	Mr.Holt	1983-08-21	104	M	Cavities	3	400	21:00:00	2022-03-19
5	Dental Polyclinic	Ms.Ruby	1998-01-16	105	F	Missing Teeth	3	400	17:00:00	2022-03-20
6	Dental Polyclinic	Ms.Franceska	2000-03-19	106	F	Mobile Teeth	3	500	18:00:00	2022-03-19
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Patient_Phone

Field	Type	Null	Key	Default	Extra
patient_id	int	NO	MUL	NULL	
Phone_number	decimal(10,0)	NO		NULL	

patient_id	Phone_number
1	9821000690
1	8999452345
2	9811223300
2	9786577724
3	9013211091
4	9210747010
5	9900887045
5	9900889085
6	9601887095

(Made a table for patient_phone as phone_ number is a multivalued attribute in patient1 table)

Visits

Field	Type	Null	Key	Default	Extra
patient_name	varchar(20)	NO	MUL	NULL	
patient_id	int	NO	PRI	NULL	
registration_time	time	NO		NULL	
registration_date	date	NO		NULL	
Final_Details	varchar(45)	NO			

patient_name	patient_id	registration_time	registration_date	Final_Details
Mr.Smith	1	17:00:00	2022-03-19	REGISTRATION CAN BE DONE
Mr.Andrews	2	14:00:00	2022-03-20	SORRY ! COME WITHIN THE SPECIFIED TIMINGS
Mrs.Rodriguez	3	17:00:00	2022-03-21	REGISTRATION CAN BE DONE
Mr.Holt	4	21:00:00	2022-03-19	SORRY ! COME WITHIN THE SPECIFIED TIMINGS
Ms.Ruby	5	17:00:00	2022-03-20	SORRY ! WE ARE OPEN ONLY FROM MONDAY-S...
Ms.Franceska	6	18:00:00	2022-03-19	REGISTRATION CAN BE DONE
NULL	NULL	NULL	NULL	NULL

This table is made with the help of table patient1(table made from a query) using 'create table as' statement. Depending upon the registration timings and this table informs us whether the patient can be given an appointment or not and with the help of case function we display the final details.

Previous_Visits

Field	Type	Null	Key	Default	Extra
patient_id	int	NO	MUL	NULL	
visits	date	NO		NULL	
prev_treatment_taken_from_this_clinic	varchar(50)	NO		NULL	

patient_id	visits	prev_treatment_taken_from_this_clinic
1	2017-08-11	Root Canal
1	2019-02-07	Root Canal
2	2016-12-11	Gums
2	2018-12-11	Gums
3	2018-03-10	Cavities
4	2018-03-23	Missing Teeth
5	2017-10-25	Mobile Teeth

New_patient

Field	Type	Null	Key	Default	Extra
patient_id	int	NO	PRI	NULL	
patient_name	varchar(20)	NO	MUL	NULL	
insurance_id	int	YES	MUL	NULL	
New_patient	varchar(50)	NO		_utf8mb4\'WELCOME!YOUR FIRST CHECKUP IS...	DEFAULT_GENERATED
discount_given	decimal(5,2)	YES		100	DEFAULT_GENERATED

patient_id	patient_name	insurance_id	New_patient	discount_given
6	Ms.Franceska	106	WELCOME!YOUR FIRST CHECKUP IS FREE	100.00
NULL	NULL	NULL	NULL	NULL

This table is made with the help of patient1 and previous visits (i.e table made from nested queries) using 'create table as' statement where the entries in the table patient1 and previous visit table are compared and the details of the patient whose patient_id is present in patient1 but not in previous_visit is stored in the table new_patient.

Regular_patients

Field	Type	Null	Key	Default	Extra
patient_id	int	NO	PRI	NULL	
patient_name	varchar(20)	NO	MUL	NULL	
insurance_id	int	YES	MUL	NULL	
discount_given	decimal(5,2)	YES		10	DEFAULT_GENERATED

patient_id	patient_name	insurance_id	discount_given
1	Mr.Smith	101	10.00
2	Mr.Andrews	102	10.00
NULL	NULL	NULL	NULL

This table is made with the help of patient1 and previous visits (i.e table made from nested queries) using 'create table as' statement where the table patient1 and previous visit table are compared and the details of the patient whose patient_id is present in previous_visits more than 2 times is stored in the table regular_patient.

Doctors

Field	Type	Null	Key	Default	Extra
doc_id	int	NO	PRI	NULL	
salary_slipno	int	NO	UNI	NULL	
doc_name	varchar(20)	NO		NULL	
Dep_no	int	NO	MUL	NULL	
Dep_name	varchar(20)	NO	MUL	NULL	

doc_id	salary_slipno	doc_name	Dep_no	Dep_name
100	100	Dr. Ray	1	Endodontist
200	101	Dr. Bing	1	Endodontist
300	102	Dr. Stromberg	2	Periodontist
400	103	Dr. David	3	General Dentist
500	104	Dr. James	3	General Dentist
NULL	NULL	NULL	NULL	NULL

DOCTOR_PHONE

Field	Type	Null	Key	Default	Extra
doc_id	int	NO	MUL	NULL	
Phone_number	decimal(10,0)	NO		NULL	

doc_id	Phone_number
100	9821054690
100	8976452345
200	9811223344
200	9786574624
300	9143211091
400	9213447010
500	9900887755

(Made a table for doctor_phone as phone_ number is a multivalued attribute in doctor_info table)

Salary

Field	Type	Null	Key	Default	Extra
salary_slipno	int	NO	PRI	NULL	
salary	decimal(10,0)	NO		NULL	
Number_of_years_working	int	NO		NULL	

salary_slipno	salary	Number_of_years_working
100	500000	4
101	250200	1
102	512200	5
103	700000	8
104	656666	6
NULL	NULL	NULL

Department

Field	Type	Null	Key	Default	Extra
dep_no	int	NO	PRI	NULL	
dep_name	varchar(20)	NO	UNI	NULL	

dep_no	dep_name
1	Endodontist
3	General Dentist
2	Periodontist
NULL	NULL

Endodontist

Field	Type	Null	Key	Default	Extra
doc_id	int	NO	MUL	NULL	
root_canal	varchar(100)	NO		NULL	
charges	int	NO		NULL	

doc_id	root_canal	charges
100	Soft tissue inflammation	4000
200	Deep decay	7000

Periodontist

Field	Type	Null	Key	Default	Extra
doc_id	int	NO	MUL	NULL	
gums	varchar(100)	NO		NULL	
price	int	NO		NULL	

doc_id	gums	price
300	Gum Disease	6000

Gen_dentist

Field	Type	Null	Key	Default	Extra
doc_id	int	NO	MUL	NULL	
cavities_OR_missing_teeth_OR_mobile_teeth	varchar(20)	NO		NULL	
PRICE	int	NO		NULL	

doc_id	cavities_OR_missing_teeth_OR_mobile_teeth	PRICE
400	Cavities	2000
400	Missing Teeth	2500
400	Mobile Teeth	2700
500	Cavities	3000
500	Missing Teeth	3500
500	Mobile Teeth	3700

TOTAL_BILL

Field	Type	Null	Key	Default	Extra
Patient_id	int	NO	PRI	NULL	
insurance_id	int	YES	MUL	NULL	
patient_name	varchar(20)	NO	MUL	NULL	
charges	int	YES		NULL	
discount_given	int	NO		NULL	
charge_after_discount	int	NO		NULL	
Money_Insurance	int	NO		NULL	
Patient_Pay	int	NO		NULL	
bill_no	int	NO	PRI	NULL	auto_increment
cashier_id	int	NO	MUL	NULL	

Patient_id	insurance_id	patient_name	charges	discount_given	charge_after_discount	Money_Insurance	Patient_Pay	bill_no	cashier_id
1	101	Mr.Smith	4000	400	3600	1980	1620	1	302
2	102	Mr.Andrews	6000	600	5400	2160	3240	2	301
3	103	Mrs.Rodriguez	7000	0	7000	4200	2800	3	302
4	104	Mr.Holt	2000	0	2000	1200	800	4	301
5	105	Ms.Ruby	2500	0	2500	750	1750	5	302
6	106	Ms.Franceska	3700	3700	0	0	0	6	301
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Here we made a table total_bill with the help of patient1,endodontist,pedodontist and gen_dentist table(table made from query) with the help of 'create table as' statement and further with the help of case function we made a column named charges and filled that particular column on the basis of patient_id and the treatment taken, this is how we found out how much the total charge of that particular patient was and next we updated the column discount_given,money_insurance_patient_pay twith the help of 'update inner join' statement

(For discount_given- **update** total_bill e **INNER JOIN** regular_patients r **ON** e.patient_id = r.patient_id **SET** e.discount_given = (charges * r.discount_given/100.00);

update total_bill e **INNER JOIN** new_patients n **ON** e.patient_id = n.patient_id **SET** e.discount_given = (charges * n.discount_given/100.00);)

(For money_insurance- **update** total_bill e **inner join** insurance i **on** e.insurance_id=i.insurance_id **SET** Money_insurance= ((charge_after_discount) * co_insurance/100.00);)

(FOR patient_pay- **update** total_bill e **inner join** insurance i **on** e.insurance_id=i.insurance_id

set patient_pay=(charge_after_discount)-money_insurance)

Dependents

Field	Type	Null	Key	Default	Extra
depen_name	varchar(100)	NO	PRI	NULL	
phone_no	decimal(10,0)	NO		NULL	
patient_id	int	NO	PRI	NULL	

	depen_name	phone_no	patient_id
▶	Roger	9165625400	1
	Fin	9165623880	2
	Thomas	9789879765	3
	Alfie	9914323523	4
	Arthur	9678229119	5
*	NULL	NULL	NULL

Medical History

	Field	Type	Null	Key	Default	Extra
▶	patient_id	int	NO	PRI	NULL	
	past_treatment	varchar(50)	NO	PRI	NULL	
	allergies	varchar(50)	YES		NULL	
	pain_tooth	varchar(50)	YES		NULL	
	heart_probs	varchar(50)	YES		NULL	
	other_illness	varchar(50)	YES		NULL	

	patient_id	past_treatment	allergies	pain_tooth	heart_probs	other_illness
▶	1	Root Canal	Penicillin	NULL	High BP	Diabetes
	2	Root Canal	NULL	Upper Left Tooth	NULL	Rhinitis
	3	Loose Teeth	Pollen	NULL	High BP	Arthritis
	4	Decay	Pollen	Lower Left Side	NULL	NULL
	5	Gingivitis	Lignocaine	Lower Right Side	High BP	Cardiac Problem
*	NULL	NULL	NULL	NULL	NULL	NULL

CASHIER

	Field	Type	Null	Key	Default	Extra
▶	Name	varchar(20)	NO		NULL	
	cashier_id	int	NO	PRI	NULL	
	salary	int	NO		NULL	

	Name	cashier_id	salary
▶	Amit	301	3000
	Rohit	302	400
*	NULL	NULL	NULL

CASHIER PHONE

	Field	Type	Null	Key	Default	Extra
▶	cashier_id	int	NO	MUL	NULL	
	Phone_number	decimal(10,0)	NO		NULL	

	cashier_id	Phone_number
▶	301	9999765642
	301	6542758545
	302	8645324455
	302	7689000678

QUERIES:

```
/*Question-01 */

select * from patient1 where patient_id IN (Select patient_id from medic_hist where heart_probs='High BP');
```

patient_id	polyclinic_name	patient_name	dob	insurance_id	sex	Problem_or_Disease	Dno	doc_id	registration_time	registration_date
1	Dental Polyclinic	Mr.Smith	1967-03-25	101	M	Soft tissue Inflammation	1	100	17:00:00	2022-03-19
3	Dental Polyclinic	Mrs.Rodriguez	1987-07-28	103	F	Deep Decay	1	200	17:00:00	2022-03-21
5	Dental Polyclinic	Ms.Ruby	1998-01-16	105	F	Missing Teeth	3	400	17:00:00	2022-03-20
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

```
/*Question-02 */

select cashier.Name,cashier.cashier_id,salary,phone_number from cashier,cashier_phone where
cashier.cashier_id=cashier_phone.cashier_id;
select cashier.name,cashier.cashier_id,salary,phone_number from cashier left outer join cashier_phone on
cashier.cashier_id=cashier_phone.cashier_id;
```

	name	cashier_id	salary	phone_number
▶	Amit	301	3000	6542758545
	Amit	301	3000	9999765642
	Rohit	302	400	7689000678
	Rohit	302	400	8645324455

```
/*Question-03 */

select * from patient1 where patient_id NOT IN (select patient_id from dependents);
```

	patient_id	polyclinic_name	patient_name	dob	insurance_id	sex	Problem_or_Disease	Dno	doc_id	registration_time	registration_date
▶	6	Dental Polyclinic	Ms.Franceska	2000-03-19	106	F	Mobile Teeth	3	500	18:00:00	2022-03-19
*	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

```
/*Question-04 */

select doctor_info.doc_id,doc_name,dep_no,dep_name,phone_number,doctor_info.salary_slipno,salary,
Number_of_years_working
from doctor_info,doctor_phone,doc_salary where doctor_info.doc_id IN
(select doc_id from doctor_phone group by doc_id having count(doc_id) >= 2 )
AND doctor_info.doc_id=doctor_phone.doc_id and
doc_salary.salary_slipno=doctor_info.salary_slipno ;
```

doc_id	doc_name	dep_no	dep_name	phone_number	salary_slipno	salary	Number_of_years_working
100	Dr. Ray	1	Endodontist	9821054690	100	500000	4
100	Dr. Ray	1	Endodontist	8976452345	100	500000	4
200	Dr. Bing	1	Endodontist	9811223344	101	250200	1
200	Dr. Bing	1	Endodontist	9786574624	101	250200	1

```
/*Question-05 */

select * from gen_dentist where doc_id IN (select doc_id from doctor_info where doc_name='Dr. David');
```

doc_id	cavities_OR_missing_teeth_OR_mobile_teeth	PRICE
400	Cavities	2000
400	Missing Teeth	2500
400	Mobile Teeth	2700

```
/*Question-06 */

select endodontist.doc_id,doc_name,root_canal,charges,dep_name from endodontist,doctor_info where
endodontist.doc_id=doctor_info.doc_id ;
```

doc_id	doc_name	root_canal	charges	dep_name
100	Dr. Ray	Soft tissue inflammation	4000	Endodontist
200	Dr. Bing	Deep decay	7000	Endodontist

```
/*Question-07 */

select * from doctor_info where salary_slipno IN (select salary_slipno from doc_salary where salary>
(select AVG(salary) from doc_salary));
```

doc_id	salary_slipno	doc_name	Dep_no	Dep_name
400	103	Dr. David	3	General Dentist
500	104	Dr. James	3	General Dentist
NULL	NULL	NULL	NULL	NULL

NORMALISATION OF TABLES:

1.) Insurance

Insurance_id -> company_name,start_date,end_date,co_insurance

Finding Closure:

$(Insurance\_ID)^+$={ company_name,start_date,end_date,co_insurance, Insurance_id}

Candidate keys={Insurance_id}

Normalisation:

Since all the attributes of the relation are atomic values and all the tuples have single values for the domain of their attributes, the table is in 1NF.

Since the table is in 1NF and there are no partial dependencies, it is also in 2NF.

Since the table is in 2NF and there are no transitive dependencies, it is also in 3NF.

Since the table is in 3NF and the determining attributes are Super keys, the table is also in BCNF.

2.) Patient1

Patient_id->patient_id, polyclinic_name, patient_name,dob,insurance_id, sex, problem_or_disease, Dno,Doc_id, registration_time, registration_date

Patient_name->patient_id

Insurance_id->patient_id

Finding Closure:

$(Patient\_ID)^+$={ Patient_ID , polyclinic_name Patient name, sex, problem_or_disease, Dno,Doc_id,registration_time, registration_date}

$(Patient\_name)^+$={ Patient_ID , polyclinic_name Patient name, sex, problem_or_disease, Dno,Doc_id,registration_time, registration_date}

$(Insurance\_id)^+$={ Patient_ID , polyclinic_name Patient name, sex, problem_or_disease, Dno,Doc_id,registration_time, registration_date}

Candidate keys={ Patient_ID, Patient_name, Insurance_id}

Normalisation:

Since all the attributes of the relation are atomic values and all the tuples have single values for the domain of their attributes, the table is in 1NF.

Since the table is in 1NF and there are no partial dependencies, it is also in 2NF.

Since the table is in 2NF and there are no transitive dependencies, it is also in 3NF.

Since the table is in 3NF and the determining attributes are Super keys, the table is also in BCNF.

3.) Patient phone

Phone_number→patient_id

This table was already divided as I observed that the table Patient1 had multiple phone numbers associated with a single patient therefore the attribute phone_number which is multivalue attribute was divided and kept into different table.

Finding closure:

$(Phone\_number)^+$={ Phone_number,patient_id}

Candidate key={ Phone_number}

Normalisation:

Since all the attributes of the relation are atomic values and all the tuples have single values for the domain of their attributes, the table is in 1NF.

Since the table is in 1NF and there are no partial dependencies, it is also in 2NF.

Since the table is in 2NF and there are no transitive dependencies, it is also in 3NF.

Since the table is in 3NF and the determining attributes are Super keys, the table is also in BCNF.

4.) Visits:

Patient_id -> Patient name,registration_time,registration_date,final_details

Registration_time, ,registration_date -> Final_details

Patient_name -> patient_id

Finding Closure:

$(Patient\_ID)^+$={ Patient_ID ,Patient name,registration_time, registration_date, final_details}

$(Registration\_time, ,registration\_date)^+$= { registration_time, registration_date, final_details }

$(Patient\_name)^+$={ Patient_name,patient_id, registration_time, registration_date, final_details}

Candidate key={ Patient_ID ,Patient name }

Normalisation:

Since all the attributes of the relation are atomic values and all the tuples have single values for the domain of their attributes, the table is in 1NF.

Since the table is in 1NF and there are no partial dependencies, it is also in 2NF.

Registration_time, ,registration_date being a non-prime attribute determines Final_details which is another non-prime attribute. Hence the table is not in 3NF. To convert it into 3NF, we make an entirely new table for

Registration_time, ,registration_date -> Final_details

Therefore we split table into two tables:

R_1 (Patient name,registration_time,registration_date)

R_2 (Registration_time, ,registration_date, final_details)

Next the table R_1 and R_2 are further in BCNF as the determining attributes are Super keys in R_1 and R_2

5.) Previous Visits:

Patient_id,visits -> prev_treatment taken from clinic

Finding Closure:

Patient_id,visits={prev_treatment taken from clinic,patient_id , visits}

Candidate key={ Patient_id,visits }

Normalisation:

Since all the attributes of the relation are atomic values and all the tuples have single values for the domain of their attributes, the table is in 1NF.

Since the table is in 1NF and there are no partial dependencies, it is also in 2NF.

Since the table is in 2NF and there are no transitive dependencies, it is also in 3NF.

Since the table is in 3NF and the determining attributes are Super keys, the table is also in BCNF.

6.) New patients

Patient_id → patient_name,insurance_id,discount_given

Patient_name → patient_id

Insurance_id → patient_id

Finding Closure:

$(Patient\_ID)^+$ ={ Patient_id, patient_name,insurance_id,discount_given}

$(Patient\_name)^+$={ Patient_id, patient_name,insurance_id,discount_given}

$(Insurance\_id)^+$= { Patient_id, patient_name,insurance_id,discount_given}

Candidate key={ Patient_ID, Patient_name, Insurance_id }

Normalization of table:

Since all the attributes of the relation are atomic values and all the tuples have single values for the domain of their attributes, the table is in 1NF.

Since the table is in 1NF and there are no partial dependencies, it is also in 2NF.

Since the table is in 2NF and there are no transitive dependencies, it is also in 3NF.

Since the table is in 3NF and the determining attributes are Super keys, the table is also in BCNF.

7.) Regular Visits

Patient_id → patient_name,insurance_id,discount_given

Patient_name → patient_id

Insurance_id → patient_id

Finding Closure:

$(Patient\_ID)^+$ ={ Patient_id, patient_name,insurance_id,discount_given}

$(Patient\_name)^+$={ Patient_id, patient_name,insurance_id,discount_given}

$(Insurance\_id)^+$= { Patient_id, patient_name,insurance_id,discount_given}

Candidate key={ Patient_ID, Patient_name, Insurance_id }

Since all the attributes of the relation are atomic values and all the tuples have single values for the domain of their attributes, the table is in 1NF.

Since the table is in 1NF and there are no partial dependencies, it is also in 2NF.

Since the table is in 2NF and there are no transitive dependencies, it is also in 3NF.

Since the table is in 3NF and the determining attributes are Super keys, the table is also in BCNF.

8.) DOCTORS

Doc_id → salary_slipno,doc_name,dep_no,dep_name

Salary_slipno→doc_id

Doc_name→doc_id

Dep_no→dep_name

Dep_name→dep_no

Finding closure:

$(Doc\_id)^+$ ={doc_id, salary_slipno,doc_name,dep_no,dep_name}

$(Salary\_slipno)^+$={ doc_id, salary_slipno,doc_name,dep_no,dep_name}

$(Doc\_name)^+$ = { doc_id, salary_slipno,doc_name,dep_no,dep_name}

$(Dep\_no)^+$ = { Dep_no,dep_name}

$(Dep\_name)^+$ = { Dep_no,dep_name}

Candidate key={ Doc_id, Salary_slipno, Doc_name}

Normalization of table:

Since all the attributes of the relation are atomic values and all the tuples have single values for the domain of their attributes, the table is in 1NF.

Since the table is in 1NF and there are no partial dependencies, it is also in 2NF.

Dep_no being a non-prime attribute determines dep_name which is another non-prime attribute. Hence the table is not in 3NF. To convert it into 3NF, we make an entirely new table for

Dep_no→dep_name

Therefore we split table into two tables:

R_1 (doc_id, salary_slipno,doc_name,dep_no)

R_2 (Dep_no,dep_name)

Next the table R_1 and R_2 are further in BCNF as the determining attributes are Super keys in R_1 and R_2

9.) DOCTOR_PHONE

Phone_number→doctor_id

This table was already divided as I observed that the table Doctors had multiple phone numbers associated with a single doctor therefore the attribute phone_number which is multivalue attribute was divided and kept into different table.

Finding closure:

$(Phone\_number)^+$={ Phone_number,doctor_id}

Candidate key={ Phone_number}

Normalisation:

Since all the attributes of the relation are atomic values and all the tuples have single values for the domain of their attributes, the table is in 1NF.

Since the table is in 1NF and there are no partial dependencies, it is also in 2NF.

Since the table is in 2NF and there are no transitive dependencies, it is also in 3NF.

Since the table is in 3NF and the determining attributes are Super keys, the table is also in BCNF.

10.) SALARY

Salary_slipno→salary,no_of_years_working

Finding closure:

$(Salary\_slipno)^+$={ Salary_slipno, salary,no_of_years_working }

Candidate key={ Salary_slipno }

Normalisation:

Since all the attributes of the relation are atomic values and all the tuples have single values for the domain of their attributes, the table is in 1NF.

Since the table is in 1NF and there are no partial dependencies, it is also in 2NF.

Since the table is in 2NF and there are no transitive dependencies, it is also in 3NF.

Since the table is in 3NF and the determining attributes are Super keys, the table is also in BCNF.

11.) DEPARTMENT:

Dep_no→ dep_name

Dep_name→dep_no

Finding closure:

$(Dep\_no)^+$={ Dep_no,dep_name}

$(Dep\_name)^+$={ Dep_no,dep_name}

Candidate key={ Dep_no,dep_name }

Normalisation:

Since all the attributes of the relation are atomic values and all the tuples have single values for the domain of their attributes, the table is in 1NF.

Since the table is in 1NF and there are no partial dependencies, it is also in 2NF.

Since the table is in 2NF and there are no transitive dependencies, it is also in 3NF.

Since the table is in 3NF and the determining attributes are Super keys, the table is also in BCNF.

12.) Endodontist

Doc_id -> rootcanal,charges

Root_canal -> doc_id

Finding closure:

$(Doc\_id)^+$ = {root_canal,charges,doc_id}

$(Root\_canal)^+$ = {root,canal,doc_id,charges }

Candidate key={ root_canal,doc_id }

Normalisation:

Since all the attributes of the relation are atomic values and all the tuples have single values for the domain of their attributes, the table is in 1NF.

Since the table is in 1NF and there are no partial dependencies, it is also in 2NF.

Since the table is in 2NF and there are no transitive dependencies, it is also in 3NF.

Since the table is in 3NF and the determining attributes are Super keys, the table is also in BCNF.

13.) Periodontist

Doc_id -> gums , price

Gums -> doc_id

Finding closure:

(Doc_id) $^{+}$ = {gums , price,Doc_id }

(Gums) $^{+}$ = {doc_id,gums,price}

Candidate key={ doc_id,gums}

Normalisation:

Since all the attributes of the relation are atomic values and all the tuples have single values for the domain of their attributes, the table is in 1NF.

Since the table is in 1NF and there are no partial dependencies, it is also in 2NF.

Since the table is in 2NF and there are no transitive dependencies, it is also in 3NF.

Since the table is in 3NF and the determining attributes are Super keys, the table is also in BCNF.

14.) Gen_dentist

Doc_id ,cavities_or_missing teeth_or_mobile_teeth -> price

Finding closure:

(Doc_id ,cavities_or_missing teeth_or_mobile_teeth)$^{+}$ = { Doc_id ,cavities_or_missing teeth_or_mobile_teeth, price }

Candidate key={ Doc_id ,cavities_or_missing teeth_or_mobile_teeth }

Normalisation:

Since all the attributes of the relation are atomic values and all the tuples have single values for the domain of their attributes, the table is in 1NF.

Since the table is in 1NF and there are no partial dependencies, it is also in 2NF.

Since the table is in 2NF and there are no transitive dependencies, it is also in 3NF.

Since the table is in 3NF and the determining attributes are Super keys, the table is also in BCNF.

15.) Total Bill

Patient_id ,bill_no -> insurance_id,patient_name,charges,discount_given,charges after discount,money_insurance , patient_pay,bill_no,cashier_id

Insurance_id -> Patient_id ,bill_no

Patient name -> Patient_id ,bill_no

Finding closure:

(Patient_id,bill _no)$^+$ =Patient_id,bill_no, insurance_id,patient_name, charges, discount_given, charges after discount,money_insurance , patient_pay,bill_no,cashier_id.

(Insurance_id) $^+$ = Patient_id, Bill_no, insurance_id, patient_name, charges, discount_given,charges after discount,money_insurance , patient_pay,bill_no,cashier_id.

(Patient_name) $^+$ = Patient_id, Bill_no, insurance_id, patient_name, charges,discount_given,charges after discount,money_insurance , patient_pay,bill_no,cashier_id.

Candidate key={ Patient_id ,bill_no, Insurance_id, Patient name }

Normalisation:

Since all the attributes of the relation are atomic values and all the tuples have single values for the domain of their attributes, the table is in 1NF.

Since the table is in 1NF and there are no partial dependencies, it is also in 2NF.

Since the table is in 2NF and there are no transitive dependencies, it is also in 3NF.

Since the table is in 3NF and the determining attributes are Super keys, the table is also in BCNF.

16.) Dependents

Depen name , patient_id -> phone number

Phone number -> patient_id,

dependent name,phone number -> patient_id

Finding closure:

(Depen_name ,patient_id)$^+$ = {Depen_name , patient_id, phone number}

(Phone number)$^+$ = {phone number ,patient_id}

(Dependent_name , phone number)$^+$ = {Dependent_name , phone number,patient_id}

Candidate key={ Dependent_name , phone number, patient_id}

Normalisation:

Since all the attributes of the relation are atomic values and all the tuples have single values for the domain of their attributes, the table is in 1NF.

Since the table is in 1NF and there are no partial dependencies, it is also in 2NF.

Since the table is in 2NF and there are no transitive dependencies, it is also in 3NF.

Since the table is in 3NF and the determining attributes are Super keys, the table is also in BCNF.

17.) Medical History

Patient_id, past_treatment->allergies,pain_tooth,heart_probs,other_illness

Finding closure:

(Patient_id, past_treatment)$^{+}$={ allergies,pain_tooth,heart_probs,other_illness, Patient_id, past_treatment }

Candidate keys={ Patient_id, past_treatment }

Normalisation:

Since all the attributes of the relation are atomic values and all the tuples have single values for the domain of their attributes, the table is in 1NF.

Since the table is in 1NF and there are no partial dependencies, it is also in 2NF.

Since the table is in 2NF and there are no transitive dependencies, it is also in 3NF.

Since the table is in 3NF and the determining attributes are Super keys, the table is also in BCNF.

18.) CASHIER

Cashier_id->Name,Salary

Name->cashier_id

Finding closure:

(Cashier_id)$^{+}$={ Name,Salary, Cashier_id}

(Name)$^{+}$={ Name,Salary, Cashier_id}

Candidate keys={ Name,Salary, Cashier_id }

Normalisation:

Since all the attributes of the relation are atomic values and all the tuples have single values for the domain of their attributes, the table is in 1NF.

Since the table is in 1NF and there are no partial dependencies, it is also in 2NF.

Since the table is in 2NF and there are no transitive dependencies, it is also in 3NF.

Since the table is in 3NF and the determining attributes are Super keys, the table is also in BCNF.

19.) CASHIER_PHONE

Phone_number→cashier_id

This table was already divided as I observed that the table Doctors had multiple phone numbers associated with a single doctor therefore the attribute phone_number which is multivalue attribute was divided and kept into different table.

Finding closure:

$(Phone\_number)^+$={ Phone_number,cashier_id}

Candidate key={ Phone_number}

Normalisation:

Since all the attributes of the relation are atomic values and all the tuples have single values for the domain of their attributes, the table is in 1NF.

Since the table is in 1NF and there are no partial dependencies, it is also in 2NF.

Since the table is in 2NF and there are no transitive dependencies, it is also in 3NF.

Since the table is in 3NF and the determining attributes are Super keys, the table is also in BCNF.