

FUNDAMENTALS OF DATABASE

PROJECT PART-5

HETERO PHARMACEUTICALS DATABASE

Group – 3

Team Members:

- Sreekar Thanda
- Shravan Kumar Nuka
- Manideep Renikindi
- Varsha Umannagari
- Rama Venkat Sumith Thota

Hetero Pharmaceuticals is a service-oriented company for medicinal drugs. Hetero provides pharma services across metro cities in INDIA. The Headquarters is located at Hyderabad. Each pharmacy has details of patients, prescriptions, doctors, day-to-day transactions, Inventory of Medicines, Insurance details. Additionally, its database holds information about the nearby clinics, which serve as entities in the whole database and have their respective attributes.

There are several entities used in this project and those are listed below,

Entity:

Hetero has enormous database as its spread across several locations in India. These entities comprise of strong and weak entities which are as listed below,

Constraints:

- We are using various constraints, they are primary key, foreign key, check and unique in this database.
- Primary keys are Employee id, Medicine code, Drug id etc.
- Foreign keys are Patient id, doctor name etc.
- Check constraint is used to check the inventory stock.

1. HETERO

Hetero is an Indian pharmacy around which our project revolves and here are the important aspects that are to be considered with Hetero. Each pharmacy has a set of attributes.

Attributes: **ph_code, Address, Contact, Zip Code.**

Primary key: **Pharmacy code (ph_code).** Here, each pharmacy store has a unique code to identify them among the listed pharmacies of that company.

Relationship: Hetero Pharmacy Store is in One-Many relationships with prescription as every pharmacy can receive multiple prescriptions from patients, One-Many relationship with employee since there are multiple employees in a single pharmacy and Many-Many relationship with inventory because there are several Medicines/Supplies in an inventory and store needs, several of such supplies.

2. Employees data:

Employees are individuals working across several stores and they include cashiers, store managers etc... Each employee has a set of attributes.

Attributes: **Employee Name, Designation, Employee Contact, Salary, Gender, Pharmacy Code.**

Primary key: Employee Identity Number (Employee_ID). Each employee would have unique Employee_ID when compared to other employees in the store.

Relationship: This Employees entity shares Many to One relationship with Pharmacy store as there are several employees working in a single store.
shares Many-One relationship with Pharmacy Locations. Employees data is considered as strong entity as this an independent to pharmacy.

3. **Inventory:** Inventory is the excess stock/surplus supplies present in warehouse or storage unit for every pharmacy company from where the supplies are sent out to every store depending upon the requirement. Inventory stores the data of the quantity of medicines and related information with respect to medicines, vaccines etc. Each inventory possesses a specific set of attributes.

Attributes: Item identity Number, Item name, Required_Quantity, Availability, presently_available_quantity.

Primary Key: Item identity Number (Item_Id). Each medicine is identified by a specific identity number for identification as several medicines from several companies have same or similar compositions and this id is used to distinguish them easily.

Relationship: This entity shares Many-Many relationships with Pharmacy since the inventory holds data for several medicines and the pharmacy possesses several medicines and requires similar medicines.

4. **Prescription:** Prescription is a list of medicine that a patient needs to use and that varies with person to person and disease to disease as doctor recommends to the patient.

Attributes: Prescription Identification Number (Prescription_Id), Date, Pharmacy Code, Doctor_Id, Patient_id.

Primary Key: Prescription Identification Number (Prescription_Id).

Relationship: It shares Many-One relationship with Pharmacy since there is a single is a single prescription and there can be many pharmacy stores to purchase that medicine. Prescription shares Ternary Relationship with Doctor and Patient and Prescription shares ternary relationship with Billing and patient.

5. **Patients:** Patients are those that suffer from any ailments and they the crucial for any pharmacy because patients give complete analysis to most of the entities in this E-R representation.

Attributes: Patient Identification Number (Patient_Id), Patient Name, Date of Birth, Disease, Gender, Contact, AGE

With the Date Of Birth we could derive the age, Hence age could be a derived attribute.

Note: AGE is a derived Attribute here.

Primary key: Patient Identification Number (Patient_Id).

Relationship: Patient has ternary relationship with Prescription and doctor and patient possess ternary relationship with billing and prescription.

6. **Doctor:** Doctor may or might not be present in the pharmacy, but he is the only authorized person to prescribe medication to patients. The following are the attributes.

Attributes: Doctor Registration Identity Number, Specialization, Contact.

Primary Key: Doctor Registration Identity Number (Doctor Reg_No).

Relationship: Doctor has only one Ternary Relationship with prescription and patient.

7. **Insurance:** Every patient has medical insurance that covers partial or full payment for the patient or their dependents. Each patient can have one or many insurances depending upon their necessity and preference. The attributes are as follows.

Attributes: Insurance Company, Amount, Insurance Number.

Primary Key: Insurance Number.

Relationship: Insurance has many to one relation with billing and Insurance possess Many-Many relationships with Patient.

8. **Billing:** This Entity describes about the transactions made by patient at pharmacy during purchase. Each medicine has a price depending on the drug and dosage and so this entity stores the cost price of the medicines and stores the information of day-to-day payments related to buying and selling the medicines.

Attributes: Date of Billing, Billing Amount, Transaction Identity Number, Transaction Number.

Primary Key: Transaction Identity Number (Transaction_Id).

Relationship: Billing entity has many to one relationship with insurance and billing possess ternary relationship with prescription and patient.

9. **Payroll table:**

Every Employee in a pharmacy is paid with salaries based on their working hours and other factors.

Attributes: working hours, id, empid, start date, end date, payroll_salary

Primary Key: id

Foreign Key: empid from Employee_data table

Relationship: Every Employee is getting salaries; this data is stored in the form of payroll.

RELATIONSHIP TABLES:

TERNARY RELATIONS.

1. **TREATMENT:** This is a ternary relationship table for patient, doctor, and prescription.
2. **PURCHASE:** This is a ternary relationship for patient, billing & Prescription.

MANY TO MANY RELATIONS.

1. **INSURANCE CLAIM**
2. **STOCK:**

RELATIONSHIPS:

TERNARY RELATIONSHIPS:

1. Doctor-Patient-Prescription
2. Patient-Prescription-Billing

ONE-MANY or MANY-MANY RELATIONSHIPS:

1. Hetero Pharmacy to Employee_data
2. Prescription to Hetero Pharmacy
3. Bills to Insurance

MANY-MANY RELATIONSHIPS:

1. Inventory to Hetero Pharmacy
2. Patient to Insurance

ONE-ONE RELATIONSHIP:

1. Employee_data to Payroll

CONSTRAINTS:

The set of rules, ensures that when an authorized user modifies the database, they do not disturb the data consistency, A constraint is a rule that is used for optimization purposes.

Primary key: A primary key constraint is a column or combination of columns that has the same properties as a unique constraint. You can use primary key and foreign key constraints to define relationships between tables.

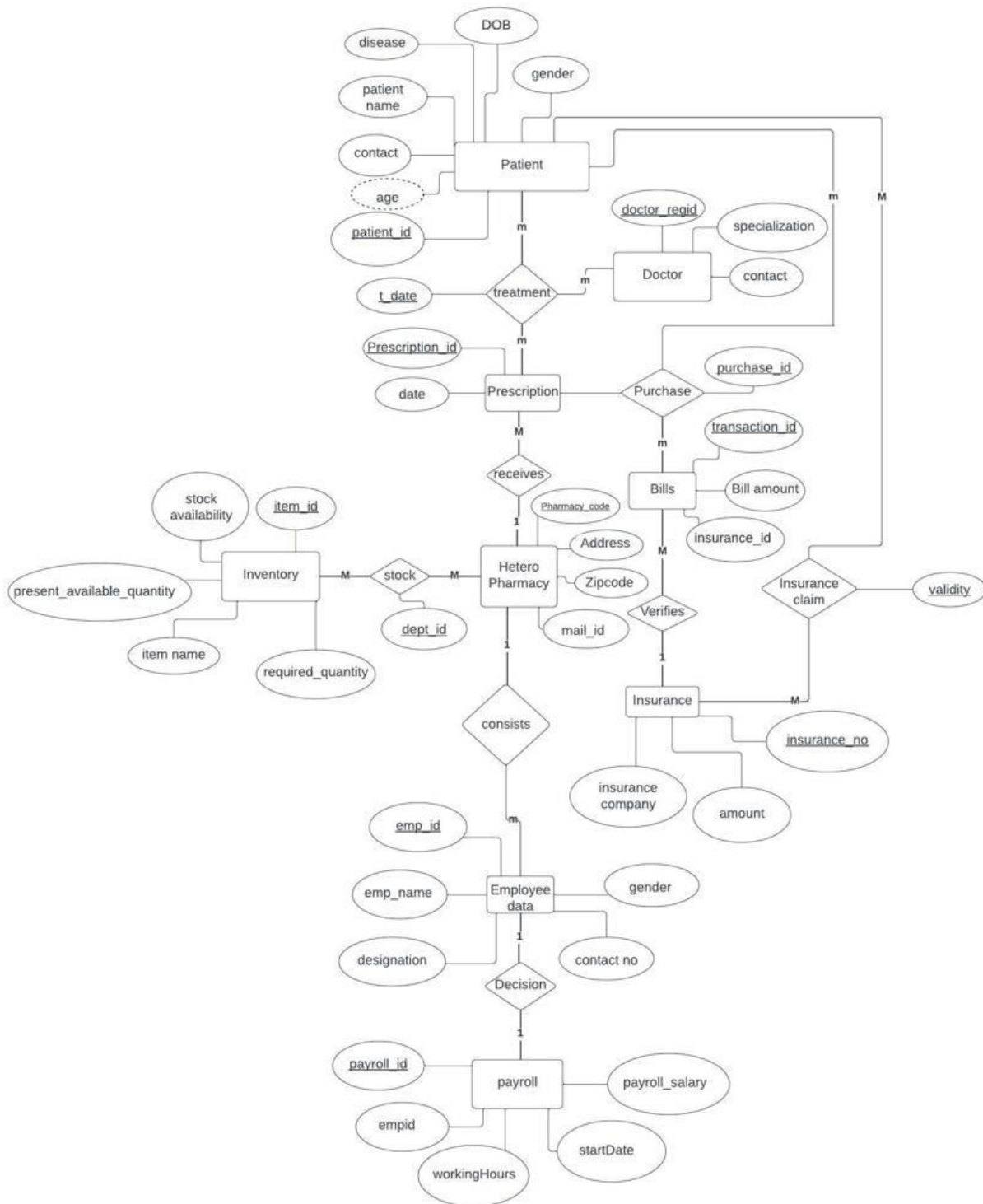
Foreign key: (referential constraint or a referential integrity constraint) is a logical rule about values in one or more columns in one or more tables. For example, a set of tables shares information about a corporation's suppliers. Occasionally, a supplier's name changes. You can define a referential constraint that states the ID of the supplier in a table must match a supplier ID in the supplier information. This constraint prevents insert, update, or delete operations that would otherwise result in missing supplier information.

Unique (unique key constraint): This is a rule that forbids duplicate values in one or more columns within a table. Unique and primary keys are the supported unique constraints. For example, a unique constraint can be defined on the supplier identifier in the supplier table to ensure that the same supplier identifier is not given to two suppliers.

Check: A check constraint (also referred to as a **table check constraint**) is a database rule that specifies the values allowed in one or more columns of every row of a table. Specifying check constraints is done through a restricted form of a search condition.

Not null: A NOT NULL constraint is a rule that prevents null values from being entered into one or more columns within a table.

The E-R Diagram is shown below:

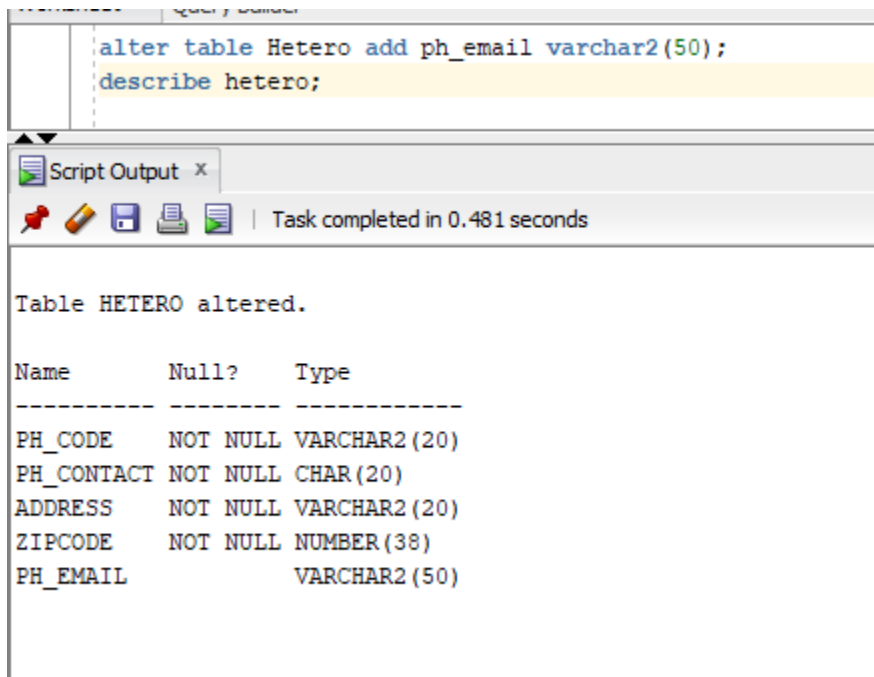


New Assumptions:

Adding the email to Hetero table,

Earlier Hetero table existed as with attributes, Ph_code, Ph_contact, address, zipcode with ph_code as primary key,

Now after altering the table (adding email to hetero table) the following attributes for it are as follows, Ph_code, Ph_contact, ph_email, address, zipcode



```
alter table Hetero add ph_email varchar2(50);
describe hetero;
```

Script Output x

Task completed in 0.481 seconds

Table HETERO altered.

Name	Null?	Type
PH_CODE	NOT NULL	VARCHAR2(20)
PH_CONTACT	NOT NULL	CHAR(20)
ADDRESS	NOT NULL	VARCHAR2(20)
ZIPCODE	NOT NULL	NUMBER(38)
PH_EMAIL		VARCHAR2(50)

project part -5

Normalization

ASSUMPTIONS:

Altering the Hetero Table by adding upon Email Address in the table.

HETERO TABLE:

The following are the attributes in the table: Ph_Code, Ph_Contact, Ph_Email, Ph_Address, Ph_Zipcode.

Now, let us consider Ph_Code = A, Ph_Contact = B, Ph_Email = C, Ph_Address = D, Ph_Zipcode = E.

The Functional Dependencies for the above table are as follows:

$A \rightarrow BCD$, $D \rightarrow E$, $A \rightarrow DE$.

i. Closures:

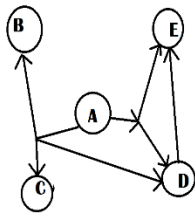
$A^+ = ABCDE$

$B^+ = B$

$C^+ = C$

$D^+ = DE$

ii. Candidate Keys: {A}



Prime Attributes: {A}

Non-Prime Attributes: {B, C, D, E}.

iii. Super Keys:

A, AB, AC, AD, AE, ABC, ABD, ABE, ACD, etc.

iv. The table is not In BCNF because the given L.H.S of the Functional Dependencies(A,D) on which D is not super key.

v. Checking if the given table in the 3rd Normal Form?

Ans: The R.H.S are prime Attributes then the table is said to be in 3rd Normal Form. Here, in the given R.H.S of the Functional Dependencies {B, C, D} are non- Prime attributes. Therefore, the table is not in the 3rd Normal Form.

vi. Checking if the given Table in 2nd Normal Form?

Ans: If the given Non-Prime Attributes are fully dependent on the Candidate key, then the table is in the 2nd Normal Form.

It could be seen that B, C, D are completely dependent on candidate key {A}. Therefore, the table is in 2nd Normal Form.

Normalizing the R (A, B, C, D, E) into BCNF.

Therefore, Decomposing the table into 2 sub tables.

Let D: {R1(A, B, C, D), R2(A, D, E)}.

Now,

R1(A, B, C, D)

Functional dependencies are = {A → BCD}

- i. Closure: A+=ABCD
- ii. Candidate key: A
- iii. Super Keys: A, AB, AC, AD, So on.
- iv. L.H.S of Functional Dependencies (A) is a super key. Therefore, R1 is in BCNF.

R2(A, D, E)

Functional Dependencies are = {D → E, A → DE}

- i. Closure:
A+=AED
D+=DEA
- ii. Candidate Key=A, D
- iii. Super Keys = A, D, AD, AE, DE, ADE.
- iv. The R2 (A, D, E) is in BCNF since the L.H.S of Functional Dependencies (A,D) are Super Keys.

Proof that the above decomposition is preserved.

For R1(A, B, C, D), Functional Dependencies are as follows: F1={A → BCD}

For R2 (A, D, E) Functional Dependencies are as follows: F2= {A → DE, D → E}

$R1 \cap R2 = A$

$R1 - R2 = BC$

$R2 - R1 = DE.$

According to rule, $(R1 \cap R2)$ should determine either of $(R1 - R2)$ or $(R2 - R1)$.

This means either $A \rightarrow BC$ or $A \rightarrow DE$ should exist in the given Functional Dependencies then the decomposition is said to be dependency preserving and Loss-Less Decomposition.

Since, $A \rightarrow DE$ exists in the Functional Dependency, this decomposition is known to be preserved and it is a Loss- Less Decomposition.

Therefore, the Normalized Table of Hetero Pharmacy table is

Hetero(ph_code, ph_contact, ph_email,address)

Location(ph_code,address,zipcode)

EMPLOYEE_DATA:

The following are the attributes in the table.

Emp_id, Emp_name, Designation, Emp_contact, Salary, Gender, Ph_code.

Let us Consider Emp_id = A, Emp_name = B, Designation = C, Emp_contact = D, Salary = E, Gender = F, Ph_code = G.

Consider the given Functional Dependencies for the above table as follows:

$A \rightarrow B$, $A \rightarrow C$, $A \rightarrow D$, $A \rightarrow E$, $A \rightarrow F$, $A \rightarrow G$, $AB \rightarrow C$, $AG \rightarrow BCDF$, $ABC \rightarrow EF$, $ABC \rightarrow E$, $F \rightarrow B$;

- i. The Closures are as follows:

$A^+ = ABCGEFG$

$B^+ = B$

$C^+ = C$

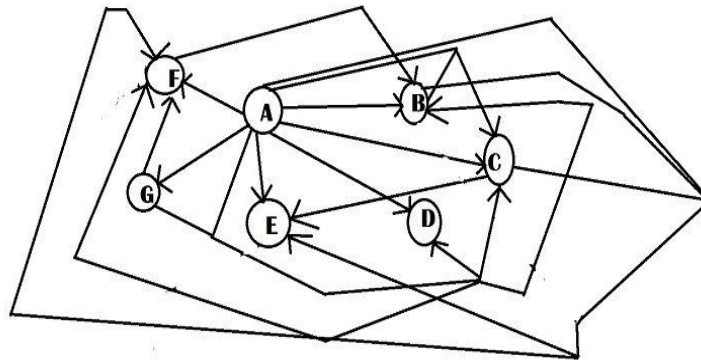
$D^+ = D$

$E^+ = E$

$F^+ = FB$

$G^+ = G$.

- ii. Candidate Keys: {A}



$V_{ni} = \{A\}$

$V_{nout} = \{B, C, D, E, F, G\}$.

- iii. Super Keys: { A, AB, AC, AD, AE, AF, AG, ABC, ACD, so on. }
- iv. As L.H.S of Functional Dependencies (A, AB, AG, ABC) are the given super keys the table Employee_data is in BCNF.

PAY_ROLL TABLE

The following are the attributes in the table.

id, Emp_id, Working Hours, Start_date, end_date, payroll_salary

Let us Consider id=A, Emp_id = B, Working Hours = C, Start_date =D, End_date = E, Payroll_salary = F,

Consider the given Functional Dependencies as follows:

$F = \{A \rightarrow B, A \rightarrow C, AB \rightarrow D, ACB \rightarrow F, AD \rightarrow E, ADE \rightarrow C, AFB \rightarrow C, AF \rightarrow C\}$

i. The Closures are as follows:

$A^+ = ABCDEF$

$B^+ = B$

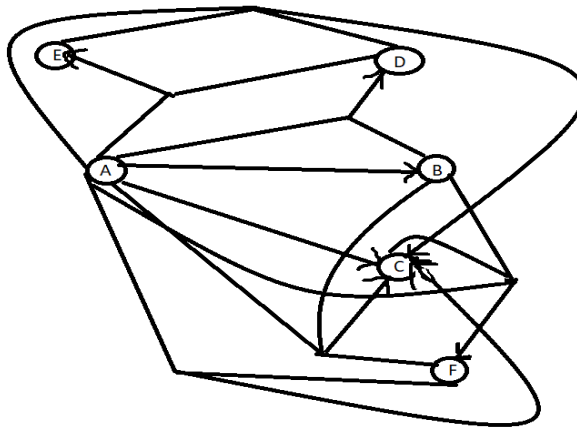
$C^+ = C$

$D^+ = D$

$E^+ = E$

$F^+ = F$

ii. Candidate Keys are as follows: A.



$V_{in} = \{A\}$

$V_{out} = \{B, C, D, E, F\}$

iii. Super Keys: A, AB, AC, AD, AE, AF, ABC, ABE, ABF, ACD, ACE, ACF, ADE, ADF, ABCD, ABCE, ABCF, so on.

iv. The payroll table is in BCNF. Since, the L.H.S of Functional Dependencies (A, AB, ABC, AD, ADE, AFB) are the super keys.

STOCK TABLE

The following are the attributes in the table.

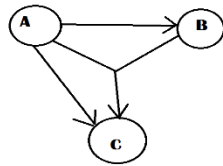
(dept_id, ph_code, item_id)

Let us Consider dept_id=A, ph_code = B, item_id = C

Consider the given Functional Dependencies as follows:

$A \rightarrow B, A \rightarrow C, AB \rightarrow C$

- i. The Closures are as follows:
 $A^+ = ABC$
 $B^+ = B$
 $C^+ = C$
- ii. Candidate Keys are as follows: A.



- $V_{in} = \{A\}$
 $V_{out} = \{B, C\}$
- iii. Super Keys: A, AB, AC, ABC
- iv. The Stock Table is in BCNF. Since, the L.H.S of Functional Dependencies (A,AB) are the given super keys.

INVENTORY TABLE:

The following are the attributes in the table:

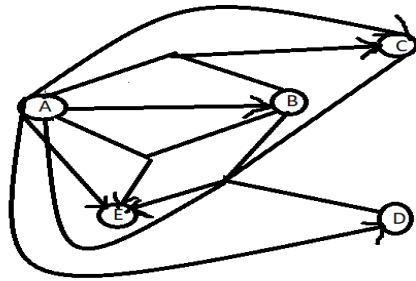
(Item_id, Item_name, Availability, Required_Quantity, Presently_Available_Quantity)

Now, Let us consider Item_id = A, Item_name = B, Availability = C, Required_Quantity = D, Presently_Available_Quantity = E.

Functional Dependencies for the above table are as follows:

$A \rightarrow B$, $A \rightarrow C$, $A \rightarrow D$, $A \rightarrow E$, $AB \rightarrow C$, $ABC \rightarrow DE$, $AB \rightarrow E$.

- i. Closures:
 $A^+ = ABCDE$.
 $B^+ = B$
 $C^+ = C$
 $D^+ = D$
 $E^+ = E$
- ii. Candidate Keys: A



iii. Super Keys: A, AB, AC, AD, AE, ABC, ABD, ABE..... so on.

iv. The Inventory table is in BCNF because the L.H.S of Functional Dependencies (A, AB, ABC) are the super keys.

PRESCRIPTION TABLE:

The following are the attributes in the table:

Prescription_Id, Prescription_Date , Ph_Code, Doctor_Id, Patient_Id.

Now, let us consider Prescription_Id = A, Prescription_Date = B, Ph_Code = C, Doctor_Id = D, Patient_Id = E.

Functional Dependencies for the above table are as follows:

$A \rightarrow B$, $A \rightarrow C$, $A \rightarrow D$, $A \rightarrow E$, $AE \rightarrow B$, $ABD \rightarrow E$, $ABC \rightarrow D$.

i. Closures

$A^+ = \text{ABCDE}$

$B^+ = B$

$C^+ = C$

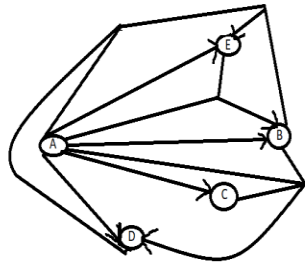
$D^+ = D$

$E^+ = E$

ii. Candidate Keys: A

$V_{ni} = \{A\}$

$V_{n_{out}} = \{B, C, D, E\}$



iii. Super Keys:

A, AB, AC, AD, AE, ABC, ABD, ABE, ACD, ACE, ADE, ABCD, ABCE, ABCDE.

iv. The L.H.S of Functional Dependencies (A, AE, ABD, ABC) are the super keys. Hence the table is in BCNF.

DOCTOR TABLE:

The following are the attributes in the table:

Doctor_Id, Doctor_Name, Specialization, Contact.

Now, let us consider Doctor_Id = A, Doctor_Name = B, Specialization = C, Contact = D.

Functional Dependencies for the above table are as follows:

$A \rightarrow B$, $A \rightarrow C$, $AB \rightarrow D$, $AB \rightarrow C$, $ABC \rightarrow D$.

i. Closures:

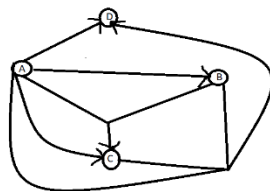
$A^+ = ABCD$

$B^+ = B$

$C^+ = CA$

$D^+ = D$

ii. Candidate Key: {A}



iii. Super Keys: {A, AB, AC, AD, ABC, ABD, ABCD}

iv. The Doctor table is in BCNF, Since the L.H.S of Functional Dependencies (A, AB, ABC) are Super Keys.

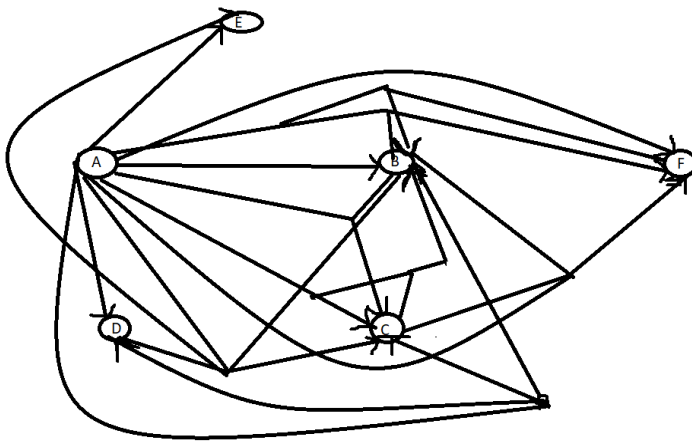
PATIENT TABLE:

The following are the attributes in the table: (Patient_Id, Patient_Name, Date_Of_Birth, Gender, Contact, Disease.)

Now, let us consider Patient_Id = A, Patient_Name = B, Date_Of_Birth = C, Gender = D, Contact = E, Disease = F.

Functional Dependencies for the above table are as follows:

$A \rightarrow B$, $A \rightarrow C$, $A \rightarrow D$, $A \rightarrow E$, $A \rightarrow F$, $A \rightarrow BCD$, $AB \rightarrow F$, $AB \rightarrow C$, $ABC \rightarrow F$, $AC \rightarrow B$, $AF \rightarrow B$, $ABD \rightarrow CE$.



- i. Closures:
 $A^+ = ABFCED$
 $B^+ = B$
 $C^+ = C$
 $D^+ = D$
 $E^+ = E$
 $F^+ = F$
- ii. Candidate Keys: {A}
 $V_{in} = \{A\}$
 $V_{out} = \{B, C, D, E, F\}$
- iii. Super Keys: A, AB, AC, AD, AE, AF, ABC, ABD, ABE, ABF.
- iv. The table is in BCNF. Since, L.H.S of Functional Dependencies (A, AB, ABC, AC, AF, ABD) are super keys.

TREATMENT TABLE:

The following are the attributes in the table: Token_Number, Prescription_Id, Patient_Id, Doctor_Id.

Now, let us consider Token_Number = A, Prescription_Id = B, Patient_Id = C, Doctor_Id = D.

Functional Dependencies for the above table are as follows:

$A \rightarrow B$, $AB \rightarrow D$, $AD \rightarrow C$, $AC \rightarrow BD$.

i. Closures:

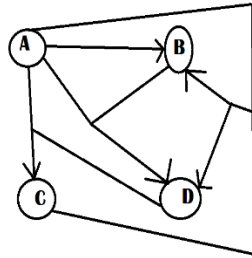
$A^+ = \mathbf{ABDC}$

$B^+ = \mathbf{B}$

$C^+ = \mathbf{C}$

$D^+ = \mathbf{D}$

ii. Candidate Keys: $\{A\}$



$V_{in} = \{A\}$ $V_{out} = \{B, C, D\}$

iii. Super Keys: $A, AB, AC, AD, ABC, ABD, ACD, ABCD$.

iv. The Treatment Table is in BCNF. Since, The L.H.S of Functional Dependencies (A, AB, AD, AC) are super keys.

BILLS TABLE:

The following are the attributes in the table: Trans_Id, Bill_Amount, Insurance_Id.

Now, let us consider Trans_Id = A, Bill_Amount = B, Insurance_Id = C.

Functional Dependencies for the above table are as follows:

$A \rightarrow B$

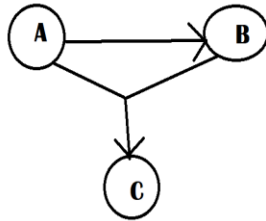
$AB \rightarrow C$

i. Closures:

$A^+ = \mathbf{ABC}$

$B^+ = \mathbf{B}$

- $C^+=C$**
 ii. Candidate Keys: {A}



- iii. $V_{ni}=\{A\}$
 $V_{nout}=\{B, C\}$
 iv. Super Keys: A, AB, AC, ABC
 v. The Bills table is in BCNF since, L.H.S of Functional Dependencies (A, AB) are super keys.

INSURANCE TABLE:

The following are the attributes in the table: Insurance_ Id, Insurance _ Company, Amount.

Now, let us consider Insurance_ Id = A, Insurance _ Company = B, Amount = C.

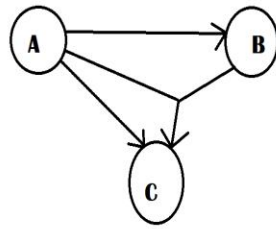
Functional Dependencies for the above table are as follows:

$A \rightarrow B$

$AB \rightarrow C$

$A \rightarrow C$

- i. Closures:
 $A^+=ABC$
 $B^+=B$
 $C^+=C$
 ii. Candidate Keys: {A}



$V_{in} = \{A\}$

$V_{out} = \{B, C\}$

iii. Super Keys: A, AB, AC, ABC.

iv. The Insurance table is in BCNF since the L.H.S of Functional Dependencies (A, AB) are Super Keys.

INSURANCE_CLAIM TABLE

The following are the attributes in the table.

(Patient_id, insurance_id, validity)

Let us Consider patient_id=A, insurance_id = B, validity = C

Consider the given Functional Dependencies as follows:

$A \rightarrow B, AB \rightarrow C$

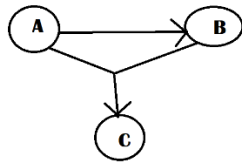
i. The Closures are as follows:

$A^+ = ABC$

$B^+ = B$

$C^+ = C$

ii. Candidate Keys are as follows: A.



$V_{ni} = \{A\}$

$V_{nout} = \{B, C\}$

iii. Super Keys: A, AB, AC, ABC

iv. The Insurance_Claim Table is in BCNF. Since, the L.H.S of Functional Dependencies (A,AB) are the super keys.

PURCHASE TABLE

The following are the attributes in the table: (Purchase_id, trans_id, prescription, patient_id)

Let us Consider purchase_id=A, trans_id=B, prescription_id=C, patient_id=D

Consider the given Functional Dependencies as follows:

$A \rightarrow B$, $B \rightarrow CD$, $D \rightarrow A$, $C \rightarrow D$

i. The Closures are as follows:

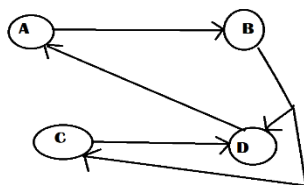
$A^+ = ABCD$

$B^+ = BACD$

$C^+ = CDAB$

$D^+ = DACB$

ii. Candidate Keys are as follows: A, B, C, D



$V_{ni} = \{A\}$

$V_{nout} = \{B, C, D\}$

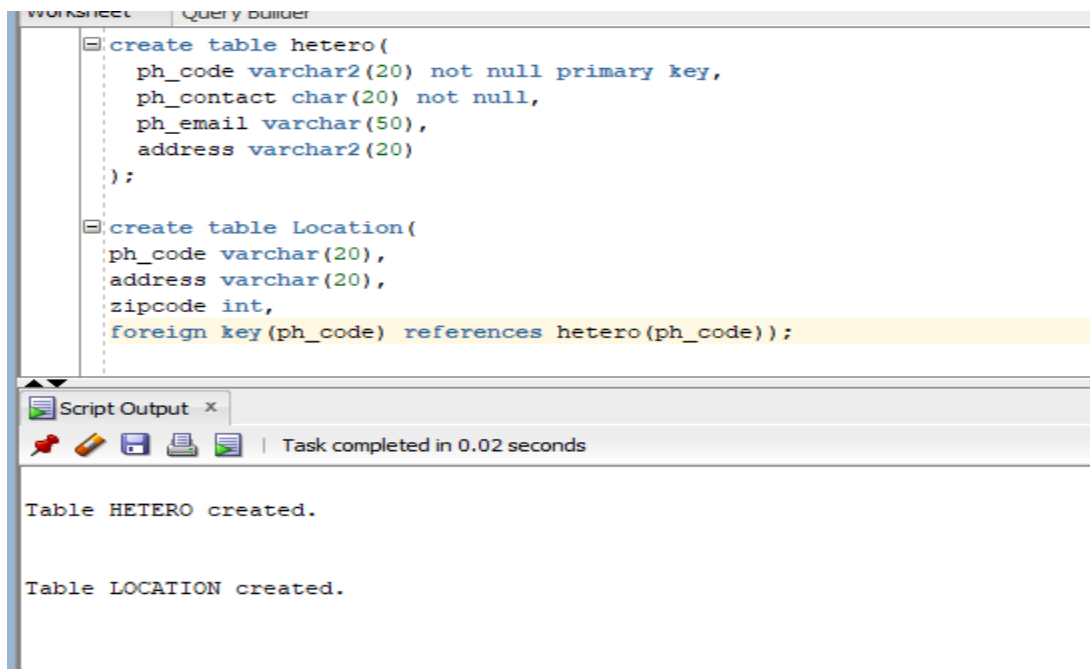
iii. Super Keys: A, B, C, D AB, AC, AD, BC, BD, CD, ABC, ABD, ACD, ABCD

- iv. The Purchase Table is in BCNF. Since the L.H.S of Functional Dependencies (A, B, D, C) are the super keys.

As the tables are now normalized, most of the tables are in BCNF.

The table Hetero is Decomposed and is split into 2 tables, where one table is Hetero, and the other table is Location table.

Therefore, the old hetero table is dropped, and new Hetero table is created. Also the Location table is also created. Respective screenshots are as below.



```
create table hetero(
  ph_code varchar2(20) not null primary key,
  ph_contact char(20) not null,
  ph_email varchar(50),
  address varchar2(20)
);

create table Location(
  ph_code varchar(20),
  address varchar(20),
  zipcode int,
  foreign key(ph_code) references hetero(ph_code));
```

Script Output x

Task completed in 0.02 seconds

Table HETERO created.

Table LOCATION created.

NOTE: As per our Database, only the Hetero table is changed and the rest of the tables are already in BCNF, As Hetero table is in 2NF, we have Normalized Hetero table into BCNF such that two tables- Hetero and Location tables are created.

Updated Description of our Hetero Pharmacy Database:

Hetero Pharmaceuticals is a service-oriented company for medicinal drugs. Hetero provides pharma services across metro cities in INDIA. The Headquarters is located at Hyderabad. Each pharmacy has details of patients, prescriptions, doctors, day-to-day transactions, Inventory of Medicines, Insurance details. Additionally, its database holds information about the nearby clinics, which serve as entities in the whole database and have their respective attributes.

There are several entities used in this project and those are listed below,

Entity:

Hetero has enormous database as its spread across several locations in India. These entities comprise of strong and weak entities which are as listed below,

1. **HETERO** : Hetero is an Indian pharmacy around which our project revolves and here are the important aspects that are to be considered with Hetero. Each pharmacy has a set of attributes. Hetero is Strong Entity.

Attributes: Ph_Code, Ph_Contact, Ph_Email, Ph_Address, Ph_Zipcode

Primary key: Pharmacy code (ph_code). Here, each pharmacy store has a unique code to identify them among the listed pharmacies of that company.

Relationship: Hetero Pharmacy Store is in One-One relationship with the Location, One-Many with Employee_data, Many-Many relationship with Inventory, One-Many with prescription as every pharmacy can receive multiple prescriptions from patients, One-Many relationships with employee since there are multiple employees in a single pharmacy and Many-Many relationships with inventory because there are several Medicines/Supplies in an inventory and store needs, several of such supplies.

2. **Location:** Location stores the information of the Address of pharmacy branches that are located at respective address. Location is a strong Entity.

Attributes: ph_code, Address, zipcode.

Relationship: Location Entity is in One-One many relationships with Hetero Entity.

3. **Employee_data:** Employees are individuals working across several stores and they include cashiers, store managers etc... Each employee has a set of attributes. Employees data is considered as strong entity as this an independent to pharmacy.

Attributes: Emp_id, Employee Name, Designation, Employee Contact, Salary, Gender, Pharmacy Code.

Primary key: Employee Identity Number (Emp_id). Each employee would have a unique Employee_ID when compared to other employees in the store.

Relationship: This Employee_Data Entity is in One-One relationship with Payroll Table, Many-One relationship with Hetero Pharmacy store as there are several employees working in a single store.

4. **Payroll:** Payroll stores every employee salary data within it. Payroll too has a unique id; payroll includes all payments made to employee from pharmacy.

Attributes: id, emp_id, workingHours, Start_date, end_date.

Primary Key: Id is the primary key in payroll which can uniquely define all other existing attributes in the payroll.

Empid is the foreign key from Employee_data.

Relationships: One-One relationship with Employee_data.

5. **Inventory:** Inventory is the excess stock/surplus supplies present in warehouse or storage unit for every pharmacy company from where the supplies are sent out to every store depending upon the requirement. Inventory stores the data of the quantity of medicines and related information with respect to medicines, vaccines etc. Each inventory possess a specific set of attributes.

Attributes: Item identity Number, Item name, Required_Quantity, Availaility, presently_available_quantity.

Primary Key: Item identity Number (**Item_Id**). Each medicine is identified by specific identity number for identification as several medicines from several companies have same or similar compositions and this id is used to distinguish them easily.

Relationship: This entity shares Many-Many relationship with Hetero Pharmacy since the inventory holds data for several medicines and the pharmacy possess several medicines and requires similar medicines, It stores the information of Medical Item name and the required quantity for each pharmacy location and the existing available quantity.

6. **Prescription:** Prescription is a list of medicine that a patient need to use and that varies with person to person and disease to disease as doctor recommends to the patient.

Attributes: Prescription Identification Number (**Prescript_Id**), **P_Date**, **Ph_Code**, **Doctor_Id**, **Patient_id**.

Primary Key: Prescription Identification Number (**Prescription_Id**).

Relationship: It shares Many-Many relationship with Hetero Pharmacy, there are many pharmacy stores and multiple prescription will be received in each pharmacy store. Prescription shares Ternary Relationship with Doctor and Patient and Prescription shares ternary relationship with Bills and patient.

7. **Patient:** Patients are those that suffer from any ailments and they the crucial for any pharmacy because patients give complete analysis to most of the entities in this E-R representation.

Attributes: Patient Identification Number(**Patient_Id**), **Patient Name**, **Date of Birth**, **Disease**, **Gender**, **Contact**.

Primary key: Patient Identification Number (**Patient_Id**).

Relationship: Patient has ternary relationship with Prescription and doctor and patient possess ternary relationship with bills and prescription, Also the patient table is in Many-Many relationships with Insurance.

8. **Doctor:** Doctor may or might not be present in the pharmacy, but he is the only authorized person to prescribe medication to patient. The following are the attributes.

Attributes: Doctor Registration Identity Number(**doctor_id**) **Doctor_Name**, **Specialization**, **Contact**.

Primary Key: Doctor Registration Identity Number (**Doctor_id**).

Relationship: Doctor has only one Ternary Relationship with prescription and patient.

9. **Insurance:** Every patient has medical insurance that covers partial or full payment for the patient or their dependents. Each patient can have one or many insurances depending upon their necessity and preference. The attributes are as follows.

Attributes: Insurance Company, **Amount**, **Insurance_id**.

Primary Key: Insurance_id, For every patient, insurance is mandatory and so a unique insurance_id is given to each patient.

Relationship: Insurance has One-Many relation with bills and possess Many-Many relationship with Patient.

10. **Bills:** This Entity describes the transactions made by patient at pharmacy during purchase. Each medicine has a price depending on the drug and dosage and so this entity stores the

cost price of the medicines and stores the information of day-to-day payments related to buying and selling the medicines.

Attributes: Transaction Identity Number (trans_id), Bills Amount, insurance_id.

Primary Key: Transaction Identity Number (Trans_Id).

Relationship: Billing entity has Many-One relationship with insurance and billing possess ternary relationship with prescription and patient.

RELATIONSHIP TABLES:

TERNARY RELATIONS.

- 11. TREATMENT:** Treatment is relationship table which notes the token_number of patients who are visiting the store on each day, also this table stores the details of prescription such as prescription_id, patient_id, doctor_id.

Attributes: Token_no, prescription_id, patient_id, doctor_id.

This is a ternary relationship table for patient, doctor, and prescription.

- 12. PURCHASE:** This is a ternary relationship for patient, billing & Prescription. This entity stores the information of the purchases that were made by the Patient/Customer.

Attributes: Purchase_id, Trans_id, Prescription_id, Patient_id.

MANY TO MANY RELATIONS.

- 13. INSURANCE CLAIM :** Insurance claim stores the information of Insurance and Validity of the patient.

Attributes: Patient_id, insurance_id, validity.

Relationship: This is a many-many relations ship entity from Patient to Insurance and vice-versa.

- 14. STOCK:** Stock table stores information of items present in all pharmacy locations with respective to each department.

Attributes: dept_id, ph_code, Item_id.

Primary Key: Dept_id is the primary key in the Stock table. This dept_id attribute can uniquely define other attributes in the table.

TERNARY RELATIONSHIPS:

1. Doctor-Patient-Prescription
2. Patient-Prescription-Billing

ONE-MANY or MANY-ONE RELATIONSHIPS:

1. Hetero Pharmacy to Employee_data
2. Prescription to Hetero Pharmacy
3. Bills to Insurance

MANY-MANY RELATIONSHIPS:

1. Inventory to Hetero Pharmacy
2. Patient to Insurance

ONE-ONE RELATIONSHIP:

1. Employee_data to Payroll
2. Location to Hetero Pharmacy

CONSTRAINTS:

The set of rules, ensures that when an authorized user modifies the database, they do not disturb the data consistency, A constraint is a rule that is used for optimization purposes.

Primary key: A primary key constraint is a column or combination of columns that has the same properties as a unique constraint. You can use primary key and foreign key constraints to define relationships between tables.

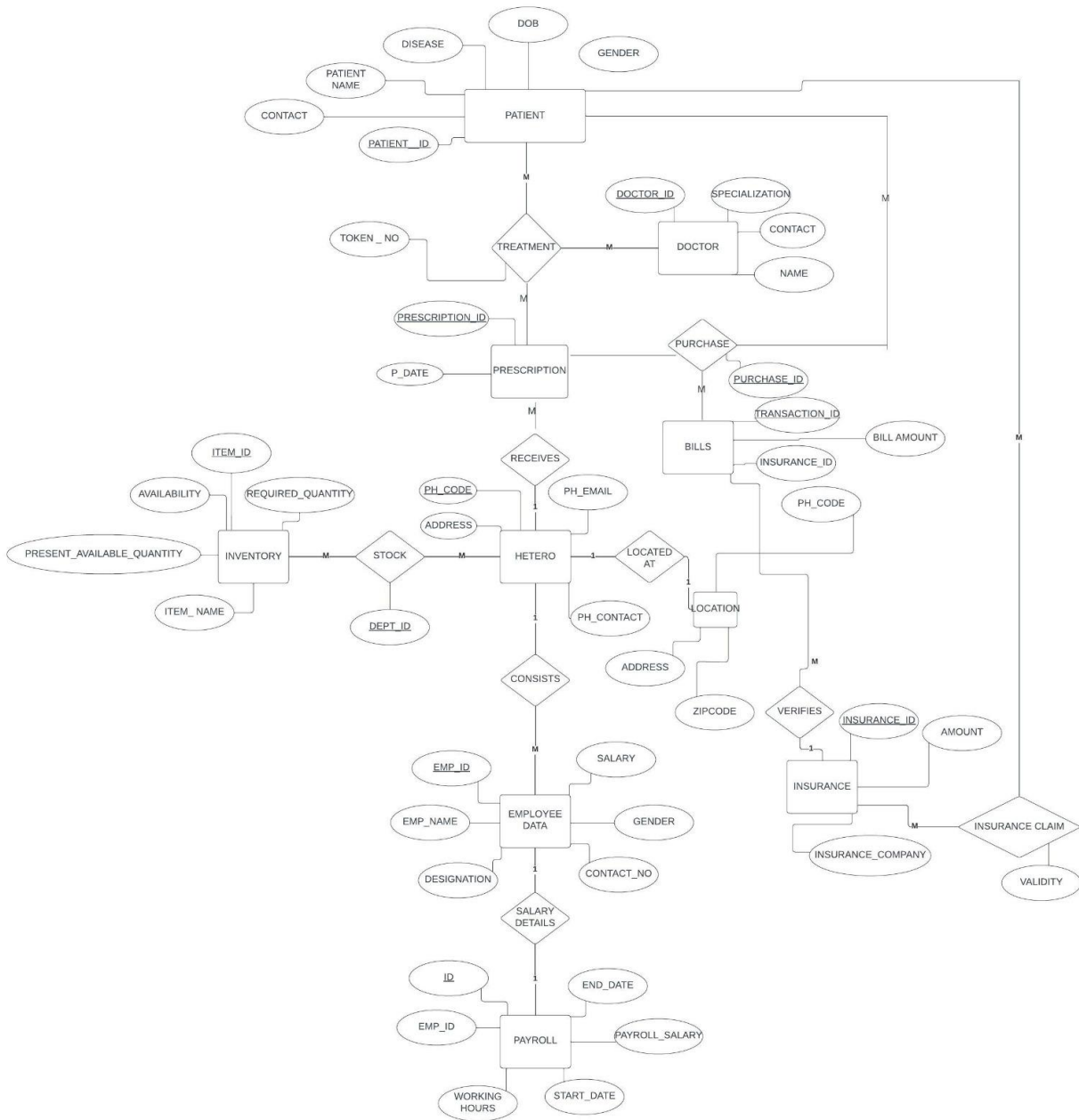
Foreign key: (referential constraint or a referential integrity constraint) is a logical rule about values in one or more columns in one or more tables. For example, a set of tables shares information about a corporation's suppliers. Occasionally, a supplier's name changes. You can define a referential constraint that states the ID of the supplier in a table must match a supplier ID in the supplier information. This constraint prevents insert, update, or delete operations that would otherwise result in missing supplier information.

Unique (unique key constraint): This is a rule that forbids duplicate values in one or more columns within a table. Unique and primary keys are the supported unique constraints. For example, a unique constraint can be defined on the supplier identifier in the supplier table to ensure that the same supplier identifier is not given to two suppliers.

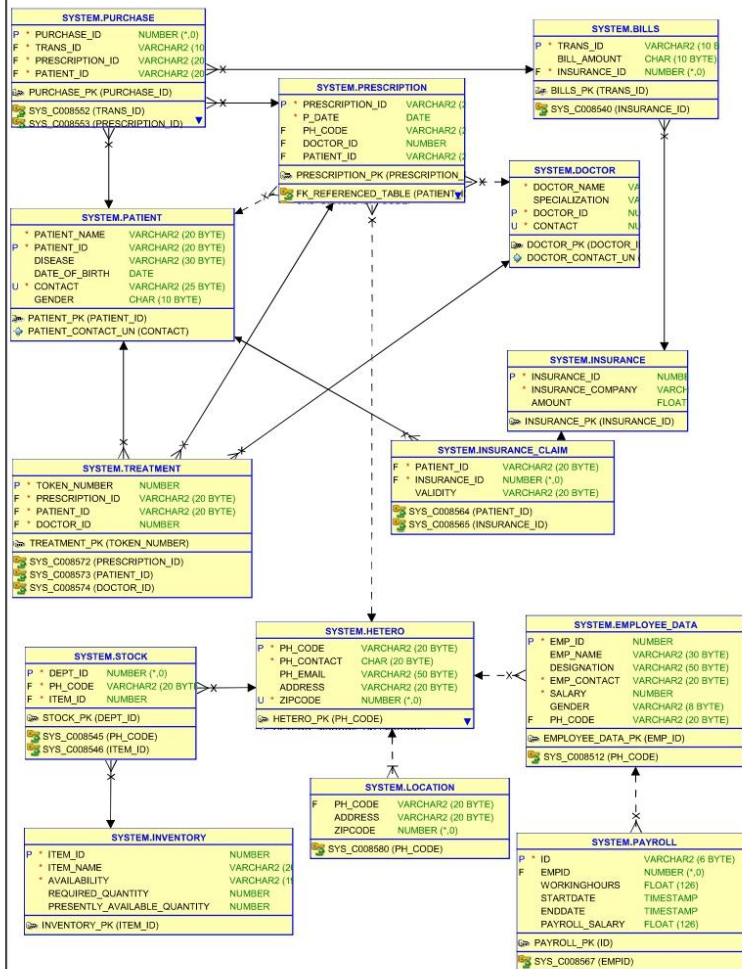
Check: A check constraint (also referred to as a **table check constraint**) is a database rule that specifies the values allowed in one or more columns of every row of a table. Specifying check constraints is done through a restricted form of a search condition.

Not null: A NOT NULL constraint is a rule that prevents null values from being entered into one or more columns within a table.

The Updated ER-Diagram is as follows below:



Relational 1 (Untitled 1)



Individual Contribution:

- I have normalized the Hetero Table from 2nd Normal form to BCNF which means the table Hetero is normalized to Hetero and Location Tables using functional dependencies. The New Hetero Table and Location tables are in BCNF. Hence, I have created those tables in our database removing the old Hetero table. I have followed the process for normalization of Hetero Table.
- I have also verified the normalized Hetero table and Location by checking if the tables are dependency preservation. After checking the dependency preservation the decomposition is loss-less decomposition.
- I have also found the prescription table to be in BCNF by following the process given by professor.
- As the prescription table is already in BCNF, I have moved forward with other process.
- I have helped my teammates with their work and verified the ER Diagram.
- I have distributed the work to teammates, monitored and guided them with their respective distributed work.
- I have cross verified my teammates' works and their tasks and their answers.
- As a team, we have collaborated with each other to complete this project Part 5.