Database Design Project

HealthOne Medical Database

Project Scenario:

You are a small database consulting company specializing in developing databases for the medical industry. You have just been awarded the contract to develop a data model for a database application system for a mid-size health insurance company to keep track of health claims including patient information, provider(doctor) information, information about patient visits to their doctor as well as prescription drugs prescribed to patients.

Information such as patient name, address, phone, email etc. are needed as well as who each patient's primary care doctor is, their insurance ID number and insurance company name. We also want information on each doctor such as their specialty and what hospitals they are affiliated with as well as their phone, address etc. Regarding the hospitals themselves we will need to know where they are located and how to contact them.

The prescriptions given to each patient by a health-care provider also need to be tracked in this particular database at this time to determine claim eligibility including some basic information on the drug being prescribed to make sure there are no conflicts with a patient's other prescriptions. We need to know each drug's name, purpose/use and possible side effects.

Eventually, the database will be used to track trends and for some extrapolative modeling based on the accumulated data. The database will be accessible in English only right now, although plans include making it available in multiple languages eventually.

Step 1: Determining entities, attributes, UIDs

Based on the business scenario stated above you will identify the database needs, and then create a conceptual data model to support these needs.

1. Review the types of information a medical database may contain. Read articles and check the internet to understand the challenges of tracking this information.

2. Research medical information to better understand your topic. Look for the type of information you will need to track.

3. Build a list of business needs, rules and assumptions based on your scenario, research, and objectives.

Answer:

- **Business Needs**

1. Patient's Accurate Data: The database must store every patient's accurate data like personal information such as full name, social security number, address, phone number and email address, insurance details such as insurance ID, name of insurance organization and their primary care doctor (personal doctor).

2. Doctor's Accurate Data: The database must maintain information on each doctor, including ID, full name, specialty, contact details and affiliations with hospitals. Every patient may have more than one doctor depending on their health condition and needs, and every doctor may have affiliations with various hospitals.

3. Prescription Management: The database must obtain information about each patient's prescriptions including drug name, purpose, dosage, date of prescription and dosage frequency. It should be able to identify any conflicts between medications, such as duplicate medicines or expired prescriptions.

4. Patient's Medical File: The database must maintain every valid information about patient's doctor visits, hospitalization, diagnosis and possible prescriptions.

5. Hospital Information: The database must store hospital details like name, location, contact details and affiliated doctors. The database should have the ability to associate hospitals with doctors easily. The association should be multiple to multiple.

6. Security: The database must ensure the privacy and security of patient data, adhering to relevant healthcare regulations. Access to sensitive data must be restricted to authorized personnel based on roles and permissions.

7. Reports and Trend Analysis: The database system should be able to collect data on diagnoses, prescriptions and visits for trend analysis and reports. It should support querying historical data to identify medical trends, track patient health and healthcare costs through time.

8. Flexibility and Scalability: The database should be flexible to accommodate future expansions, such as multilingual support, tracking additional healthcare services or new businesses rules. It needs to be scalable as the patient base, doctor network and insurances companies grow.

Business Rules

1. Patient's Primary Care Doctor (Personal Doctor):
    a. A patient can have only one primary care doctor at any given time.
    b. A doctor can be the primary care doctor for multiple patients.

2. Data Integrity:
    a. Each entity patient, doctor, hospital, prescription must have a unique identifier (primary key) that is referenced by foreign keys in other tables.
    b. The database system must enforce data consistency, ensuring that no false records exist, such as a prescription without an associated patient or doctor.

3. Doctor's Specialty:
    a. A doctor's specialty field should be mandatory. In this way the database ensures the accuracy of every doctor's information and helps to categorize the type of care the patient needs.
    b. This field may also enable analysis for future study and track the needs of every hospital department grow in the future.

4. Doctor – Hospital Affiliations:
    a. A doctor can have affiliations with multiple hospitals.
    b. A hospital can have affiliations with multiple doctors.

5. Hospital Location:
    a. A hospital's location field should be mandatory to enable easy contact and references.

6. Prescriptions Conflicts and Limitations:
    a. A patient can receive multiple prescriptions from multiple doctors, but the database system must check for conflicts, such as duplicate prescriptions or medications.
    b. Prescriptions must be querying by drug name, dosage, and frequency to ensure proper usage.
    c. Prescription medications and services provided during doctor visits should be in accordance with the benefits of the insurance company.
    d. Only medically necessary procedures and prescriptions that align with the patient's insurance coverage should be considered for executional approval.

7. Security Controls:
    a. All sensitive operations like adding or modifying patient data, must be logged for auditing purposes.
    b. The database system must have role-based access control to restrict access to sensitive data, such as patient medical history.
8. Language Support:
    a. While the database system initially will be built in English, the data should be stored as text data in a way that can be easily translated in other languages.
    b. International medical and pharmaceutical terminology should be used in data related to medical specialties, diagnoses, and prescriptions.

Assumptions

1. Data Accuracy:
    a. Patient, Doctor and Hospital data should be accurate, complete and consistent.
    b. Insurance company details and credentials should be verified and standardized before logging into the system.

2. Patient Data Management:
    a. Patient data will be updated regularly to reflect changes in personal data, doctor visits and prescriptions.

3. Health Insurance Requirements:
    a. The insurance company should provide all the required information for each patient such as the health coverage and healthcare costs.

4. Regulatory Compliance:
    a. The insurance companies must comply with the relevant healthcare regulations, in order to ensure that patient data is managed and stored securely.
    b. The database system should be designed to allow easy updates to remain compliant with possible legal changes and regulations over time.

4. Develop a list of potential entities including their attributes, the attribute's optionality as well as a possible UID for each entity.

Answer:

- **PATIENT:**
  - Attributes >

    # SS_NUMBER (UID)
    * PR_DOC_ID (required) (FK to DOCTOR)
    * P_NAME (required)
    * P_ADDRESS (required)
    * P_PHONE (required)
      P_EMAIL (optional)
    * INSURANCE_ID (required)
    * INSURANCE_NAME (required)

  - Optionality: The attribute SS_NUMBER is required as a primary key for the entity PATIENT and referred to the Social Security Number of each patient which is unique.
    The attribute PR_DOC_ID is required because each patient must have at least one primary care doctor and is a foreign key to the entity DOCTOR.
    The attributes P_NAME, P_ADDRESS and P_PHONE are also required for communication.
    The attributes INSURANCE_ID and INSURANCE_NAME are required to ensure the details of insurance coverage.
    The UID for the entity PATIENT will be the attribute SS_NUMBER .

- **DOCTOR:**
  - Attributes >

    # DOCTOR_ID (UID)
    * D_NAME (required)
    * SPECIALTY (required)
    * D_ADDRESS (required)
    * D_PHONE (required)
    * H_AFFILIATION (required) (FK to HOSPITAL)

  - Optionality: The attribute DOCTOR_ID is required as a primary key for the entity DOCTOR which is unique.
    The attribute SPECIALTY is also required.

The attributes D_ADDRESS and D_PHONE are required for communication.
The attribute H_AFFILIATION is required because each doctor must have at least one affiliated hospital.
The UID for the entity DOCTOR will be the attribute DOCTOR_ID which is a unique number for each doctor.

- **PRESCRIPTION:**
  - Attributes >

    # PRES_ID (UID)
    * SS_NUMBER (required) (FK to PATIENT)
    * DOCTOR_ID (required) (FK to DOCTOR)
    * DRUG (required)
    * DOSAGE (required)
    * QUANTITY (required)
    * EXPIRE_DATE (required)

  - Optionality: All of the attributes are required.
    The UID for the entity PRESCRIPTION will be the attribute PRES_ID which is a unique number for each prescription.

- **MEDICAL_FILE:**
  - Attributes >

    # SS_NUMBER (UID) (FK to PATIENT)
    * DOCTOR_ID (required) (FK to DOCTOR)
    * PRES_ID (required) (FK to PRESCRIPTION)
    * V_DATE (required)
    * V_REASON (required)
      DIAGNOSIS (optional)

  - Optionality: The attributes SS_NUMBER, DOCTOR_ID and PRES_ID foreign keys for the entities PATIENT, DOCTOR and PRESCRIPTION accordingly.
    The attributes V_DATE and V_REASON are required for tracking.
    The attribute DIAGNOSIS is optional because may not always be provided.
    In this entity the possible UID will be the attribute SS_NUMBER which will serve as both primary key for the MEDICAL_FILE entity and as a foreign key that references the SS_NUMBER in the PATIENT entity.

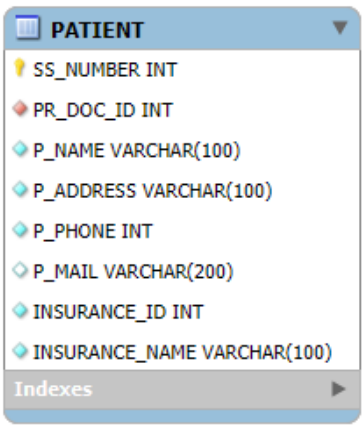
- **HOSPITAL:**
  - Attributes >

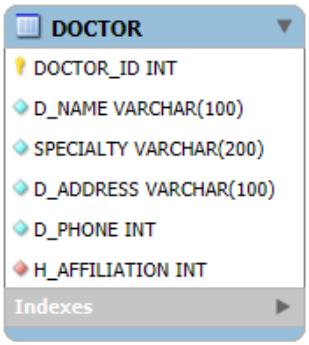
    # HOSPITAL_ID (UID-PK)
    * H_NAME (required)
    * H_LOCATION (required)
    * H_PHONE (required)

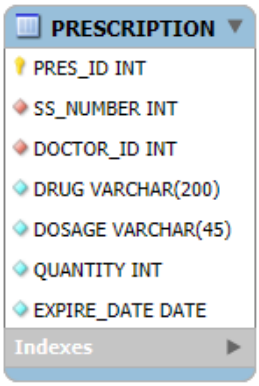
  - Optionality: All the attributes are required.
    The UID for the entity HOSPITAL will be the attribute HOSPITAL_ID which is a unique number for each hospital.

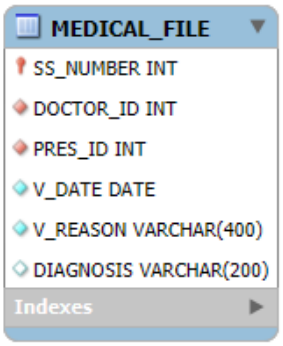
5. Create a preliminary Entity Relationship Diagram (ERD) that meets these needs and objectives. (Note : Create entities only, relationships will be added in Step 3)

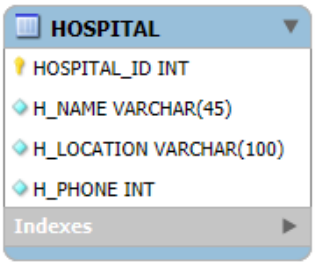
Answer:











Step 2: Supertypes and subtypes

As stated in the scenario we need to track the visits a patient makes to their doctor. Some patient visits are related to a new issue/illness, some are follow up visits to an existing diagnosis and some visits are routine "well patient" visits or checkups. We would like to be able to track which type of visit each instance is so we can keep specific information regarding the visit. For example :

1. For a new issue/illness visit we will store an initial diagnosis.
2. For follow-up visits we need to keep track of the patient's status regarding the diagnosis.
3. For routine checkups we need to track patient vital information such as current blood pressure, height and weight.

Modify the ERD using a supertype/subtype structure within the Visit entity.

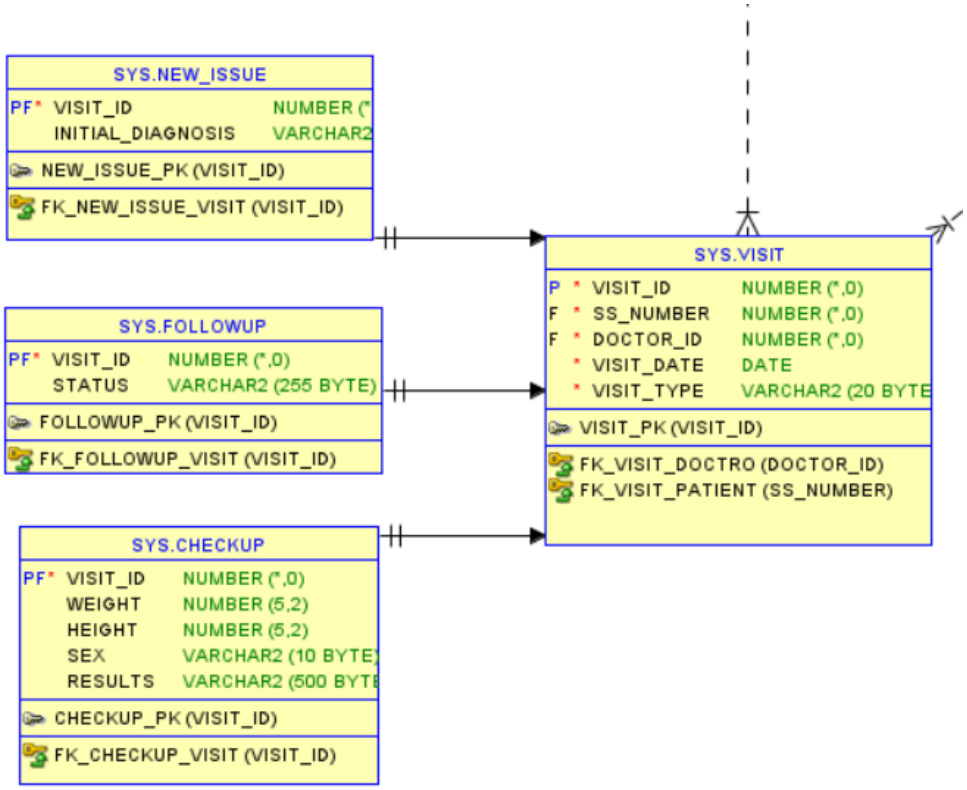
Answer:

To accommodate the differentiation between visit types:

- New issue/illness
- Follow-up
- Checkups

the VISIT entity is treated as a supertype with the subtypes:

- NEW_ISSUE
- FOLLOWUP
- CHECKUP



Step 3: Relationships

While creating your entities you should have been thinking about what relationships the entities would have with each other. Create the relationships between your entities including the relationship's optionality and cardinality.

1. Write out the ERDish for each of the relationships.

Answer:

- Each PATIENT must have exactly one DOCTOR as their primary care doctor. The relationship is mandatory, 1.
- Each DOCTOR can be the primary care doctor for more than one PATIENT. The relationship is optional, M.
- A DOCTOR must be affiliated with at least one HOSPITAL. The relationship is mandatory, M.
- A HOSPITAL can have more than one affiliated DOCTOR. The relationship is optional, M.
- A PATIENT may or may not have PRESCRIPTION. The relationship is optional, M.
- A PRESCRIPTION must belong to exactly one PATIENT. The relationship is mandatory, 1.
- A DOCTOR may or may not prescribe one or more PRESCRIPTION. The relationship is optional, M.
- Each PRESCRIPTION must be tied to one and only one DOCTOR. The relationship is mandatory, 1.
- A PATIENT can have exactly one MEDICAL_FILE. The relationship is mandatory, 1.
- A MEDICAL_FILE must be tied to exactly one PATIENT. The relationship is mandatory, 1.
- A PATIENT can have one or more VISITs. The relationship is optional, M.
- A VISIT must belong to exactly one PATIENT. The relationship is mandatory, 1.
- A DOCTOR can be associated with one or more VISITs. The relationship is optional, M.
- A VISIT must be associated with one and only DOCTOR. The relationship is mandatory, 1.
- A VISIT must belong to exactly one of the subtypes: NEW_ISSUE, FOLLOWUP or CHECKUP. The relationship is mandatory, 1.
- Each subtype must be associated with one and only VISIT. The relationship is mandatory, 1.
- A VISIT must belong to exactly one MEDICAL_FILE. The relationship is mandatory, 1.
- A MEDICAL_FILE can document one or more VISITs. The relationship is optional, M.

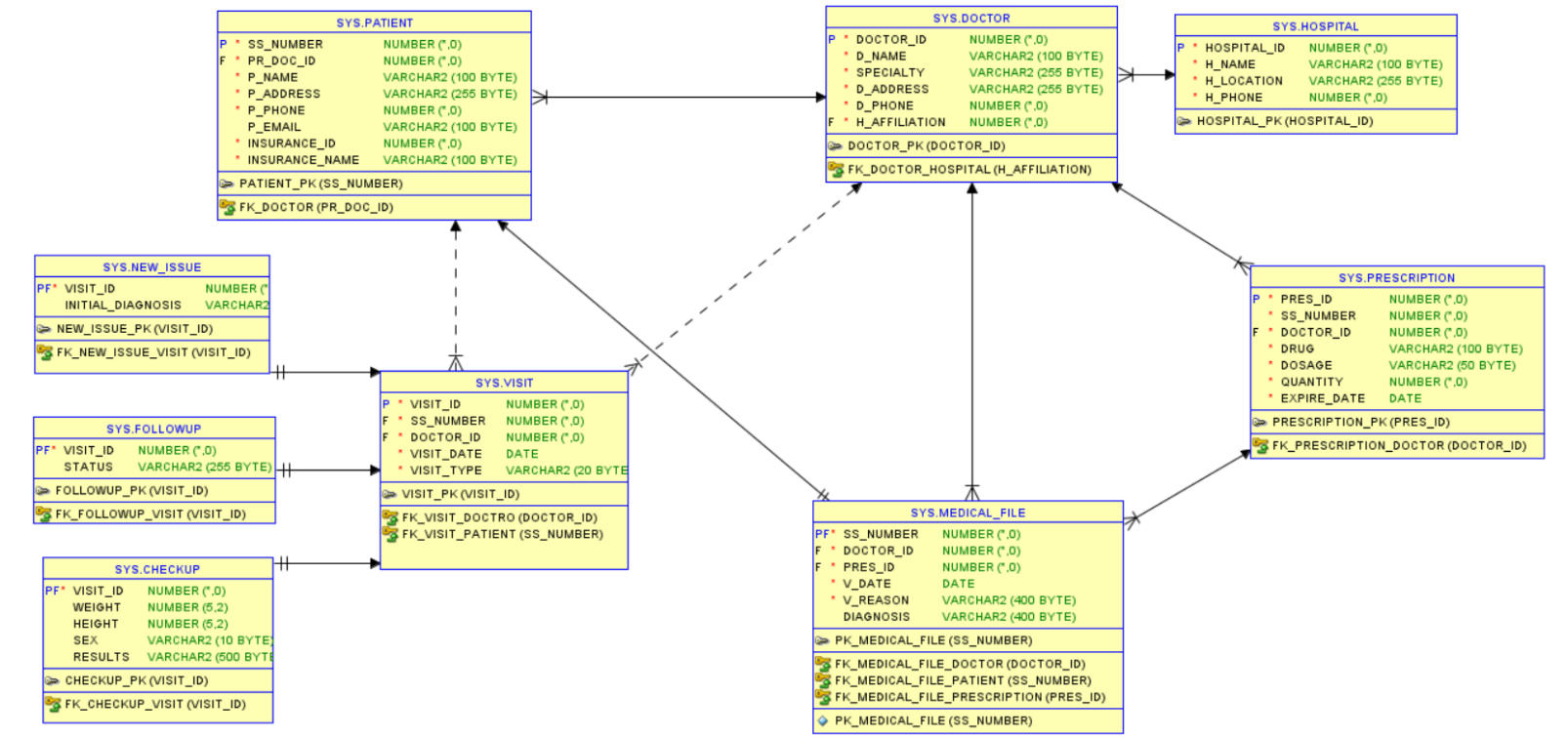
2. Some relationships will be transferrable and some non-transferrable, be sure to illustrate this point on the ERD. For example, once a prescription is written for a patient it cannot be transferred to another patient.

3. The possible relationship types are : 1-to-1, 1-to-many and many-to-many.

4. Any many-to-many relationships will need to be resolved. For example, each doctor may be affiliated with many hospitals and each hospital may have many doctors affiliated with it. We need to make sure this many-to-many relationship is resolved so that we can track which doctors are affiliated with which hospitals.

Modify the ERD to include the relationships.

Answer:



Step 4: Normalization

Ensure that each entity has been normalized to third normal form – this means:

1. 1sT normal form states that all attributes have a single value - no multivalued attributes. For example : each patient can only have one primary doctor, each doctor can only have one specialty etc.

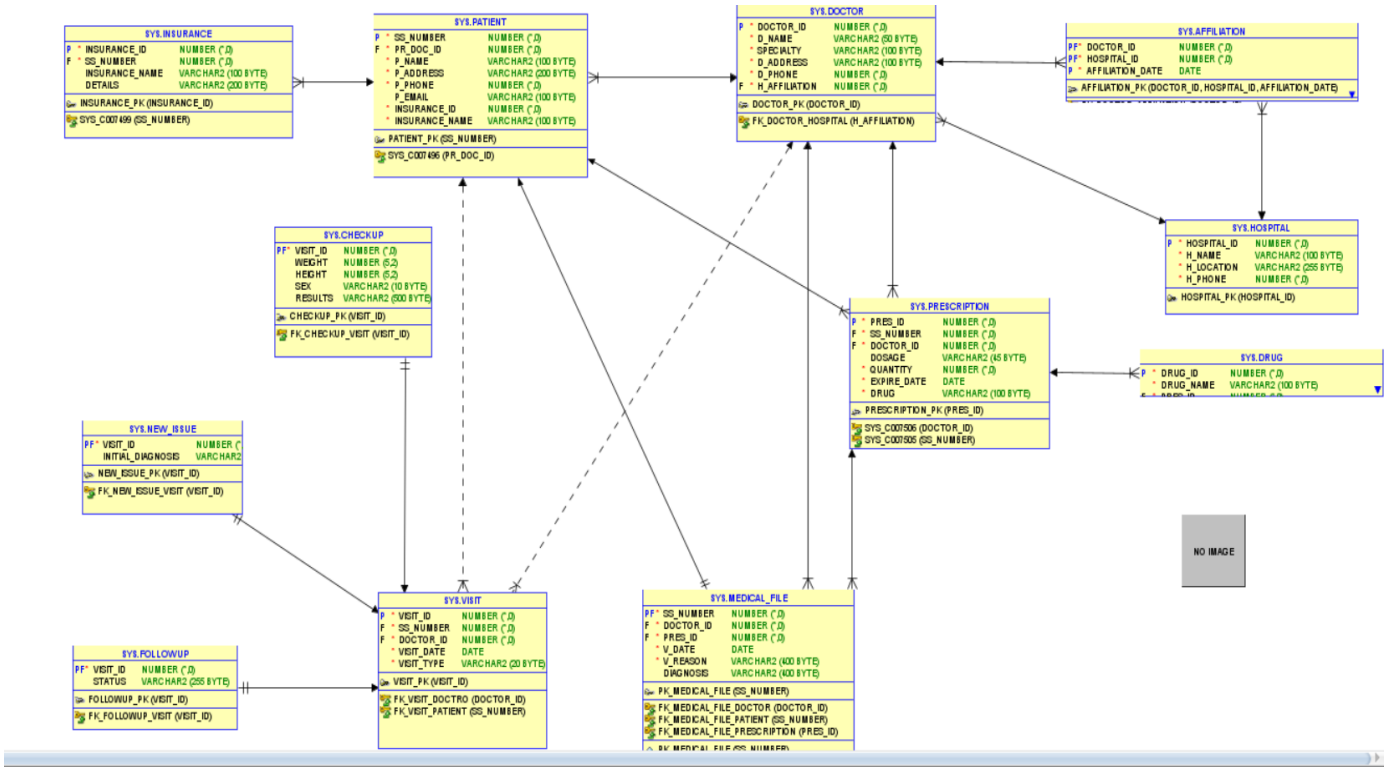
2. 2nd normal form says that all attributes must be dependent on the entire key of the entity. For example, we need to know each drug's name, purpose and side effects but if we include this in the Prescription entity it will be dependent only on what drug is prescribed not who it's for or what doctor prescribed it – so it does not belong in the same entity as the prescription information itself.

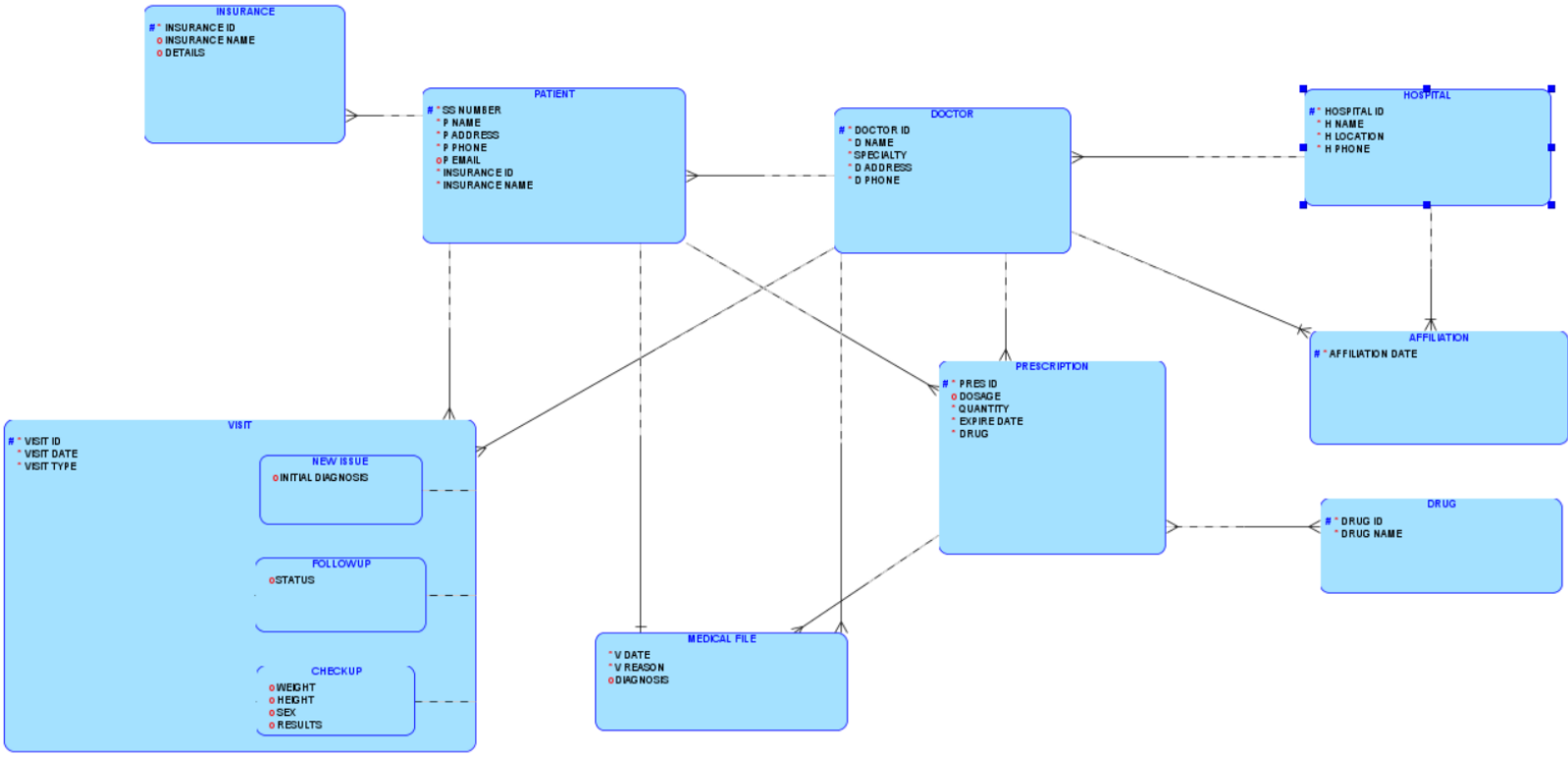
3. 3rd normal form states that no non-UID attribute can be dependent on another non-UID attribute. For example : A patient's insurance ID number will determine what insurance company they are insured with. The ID number determines the insurance company's name.

Modify the ERD to incorporate all 3 stages of normalization

Answer:

- From the table PERSON attributes INSURANCE_ID and INSURANCE_NAME became a separate entity INSURANCE with the above attributes. Each PATIENT must have only one INSURANCE, but each INSURANCE can have more than one PATIENT.
- From the table DOCTOR the attribute H_AFFILIATION became a separate entity AFFILIATION with attributes DOCTOR_ID, HOSPITAL_ID and AFFILIATION_DATE. Each DOCTOR can be affiliated with multiple HOSPITAL, and each HOSPITAL can affiliate with multiple DOCTOR.
- From the table PRESCRIPTION the attributes drug became a new separate entity DRUG with attributes DRUG_ID, DRUG_NAME and SIDE_EFFECTS. Each PRESCRIPTION can contain more than one DRUG, and each DRUG can be included in more than one PRESCRIPTION.
- All the tables are in 1NF because all the attributes are not multi-valued attributes.
- They are in 2NF since their primary key uniquely determines all the other attributes.
- It is in 3NF because there are no transitive dependencies, and there are no partial dependencies because there's only one candidate key.
- The tables are in BCNF.


SYS.INSURANCE
INSURANCE_ID NUMBER (*,0)
SS_NUMBER NUMBER (*,0)
INSURANCE_NAME VARCHAR2 (100 BYTE)
DETAILS VARCHAR2 (200 BYTE)
INSURANCE_PK (INSURANCE_ID)
SYS_C007499 (SS_NUMBER)
SYS.PATIENT
SS_NUMBER NUMBER (*,0)
PR_DOC_ID NUMBER (*,0)
P_NAME VARCHAR2 (100 BYTE)
P_ADDRESS VARCHAR2 (200 BYTE)
P_PHONE NUMBER (*,0)
P_EMAIL VARCHAR2 (100 BYTE)
INSURANCE_ID NUMBER (*,0)
INSURANCE_NAME VARCHAR2 (100 BYTE)
PATIENT_PK (SS_NUMBER)
SYS_C007496 (PR_DOC_ID)
SYS.DOCTOR
DOCTOR_ID NUMBER (*,0)
D_NAME VARCHAR2 (100 BYTE)
SPECIALTY VARCHAR2 (100 BYTE)
D_ADDRESS VARCHAR2 (100 BYTE)
D_PHONE NUMBER (*,0)
H_AFFILIATION NUMBER (*,0)
DOCTOR_PK (DOCTOR_ID)
FK_DOCTOR_HOSPITAL (H_AFFILIATION)
SYS.AFFILIATION
DOCTOR_ID NUMBER (*,0)
HOSPITAL_ID NUMBER (*,0)
AFFILIATION_DATE DATE
AFFILIATION_PK (DOCTOR_ID, HOSPITAL_ID, AFFILIATION_DATE)
SYS.HOSPITAL
HOSPITAL_ID NUMBER (*,0)
H_NAME VARCHAR2 (100 BYTE)
H_LOCATION VARCHAR2 (255 BYTE)
H_PHONE NUMBER (*,0)
HOSPITAL_PK (HOSPITAL_ID)
SYS.CHECKUP
VISIT_ID NUMBER (*,0)
WEIGHT NUMBER (5,2)
HEIGHT NUMBER (5,2)
SEX VARCHAR2 (10 BYTE)
RESULTS VARCHAR2 (500 BYTE)
CHECKUP_PK (VISIT_ID)
FK_CHECKUP_VISIT (VISIT_ID)
SYS.PRESCRIPTION
PRES_ID NUMBER (*,0)
SS_NUMBER NUMBER (*,0)
DOCTOR_ID NUMBER (*,0)
DOSAGE VARCHAR2 (45 BYTE)
QUANTITY NUMBER (*,0)
EXPIRE_DATE DATE
DRUG VARCHAR2 (100 BYTE)
PRESCRIPTION_PK (PRES_ID)
SYS_C007505 (DOCTOR_ID)
SYS_C007505 (SS_NUMBER)
SYS.DRUG
DRUG_ID NUMBER (*,0)
DRUG_NAME VARCHAR2 (100 BYTE)
SYS.NEW_ISSUE
VISIT_ID NUMBER (*,0)
INITIAL_DIAGNOSIS VARCHAR2
NEW_ISSUE_PK (VISIT_ID)
FK_NEW_ISSUE_VISIT (VISIT_ID)
NO IMAGE
SYS.VISIT
VISIT_ID NUMBER (*,0)
SS_NUMBER NUMBER (*,0)
DOCTOR_ID NUMBER (*,0)
VISIT_DATE DATE
VISIT_TYPE VARCHAR2 (20 BYTE)
VISIT_PK (VISIT_ID)
FK_VISIT_DOCTRO (DOCTOR_ID)
FK_VISIT_PATIENT (SS_NUMBER)
SYS.MEDICAL_FILE
SS_NUMBER NUMBER (*,0)
DOCTOR_ID NUMBER (*,0)
PRES_ID NUMBER (*,0)
V_DATE DATE
V_REASON VARCHAR2 (400 BYTE)
DIAGNOSIS VARCHAR2 (400 BYTE)
PK_MEDICAL_FILE (SS_NUMBER)
FK_MEDICAL_FILE_DOCTOR (DOCTOR_ID)
FK_MEDICAL_FILE_PATIENT (SS_NUMBER)
FK_MEDICAL_FILE_PRESCRIPTION (PRES_ID)
SYS.FOLLOWUP
VISIT_ID NUMBER (*,0)
STATUS VARCHAR2 (255 BYTE)
FOLLOWUP_PK (VISIT_ID)
FK_FOLLOWUP_VISIT (VISIT_ID)


INSURANCE
INSURANCE ID
INSURANCE NAME
DETAILS
PATIENT
SS NUMBER
P NAME
P ADDRESS
P PHONE
P EMAIL
INSURANCE ID
INSURANCE NAME
DOCTOR
DOCTOR ID
D NAME
SPECIALTY
D ADDRESS
D PHONE
HOSPITAL
HOSPITAL ID
H NAME
H LOCATION
H PHONE
AFFILIATION
AFFILIATION DATE
PRESCRIPTION
PRES ID
DOSAGE
QUANTITY
EXPIRE DATE
DRUG
DRUG
DRUG ID
DRUG NAME
VISIT
VISIT ID
VISIT DATE
VISIT TYPE
NEW ISSUE
INITIAL DIAGNOSIS
FOLLOWUP
STATUS
CHECKUP
WEIGHT
HEIGHT
SEX
RESULTS
MEDICAL FILE
V DATE
V REASON
DIAGNOSIS

Step 5: Arcs

Each prescription issued by a doctor must be refillable or non-refillable. It can't be both.

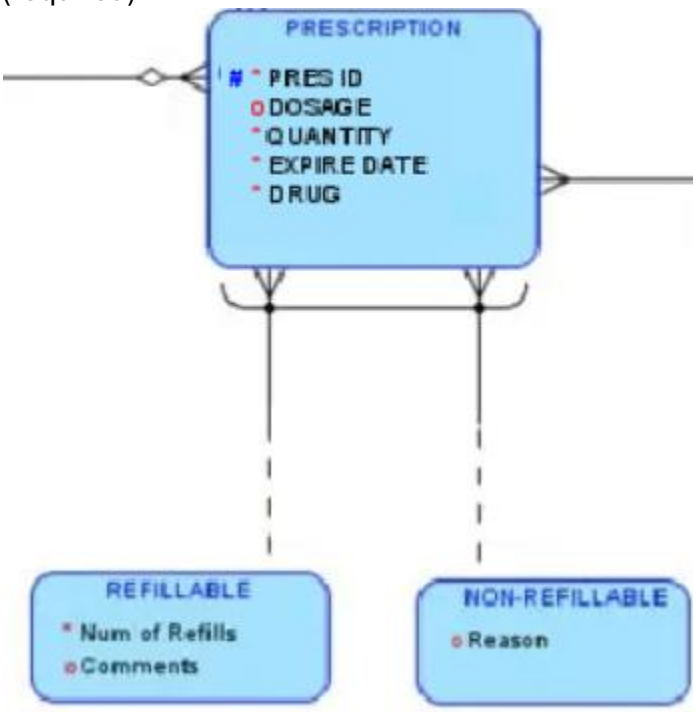
Modify the ERD to make this distinction using an arc. Refillable prescriptions will have information about the number and size of refills. All prescriptions will need information about the date, dosage and duration of the RX.

Answer:

- The PRESCRIPTION entity should be modified to include information about whether the prescription is refillable or not. If the prescription is refillable, we need to store additional attributes related to the refills like frequency, refill size, etc.
- An Arc will be created to separate refillable prescriptions from non-refillable ones. The Arc will only apply to prescriptions that are marked as refillable, and the Arc will define the attributes specific to refillable prescriptions.
- The entity REFILL is linked to the entity PRESCRIPTION via an Arc that claims if IS_REFILLABLE=TRUE and it has the attributes below.

REFILL:

- Attributes >
  - * REFILL_COUNT (required)
  - * REFILL_FREQUENCY (required)
  - * REFILL_SIZE (required)
  - * REFILL_DATE (required)

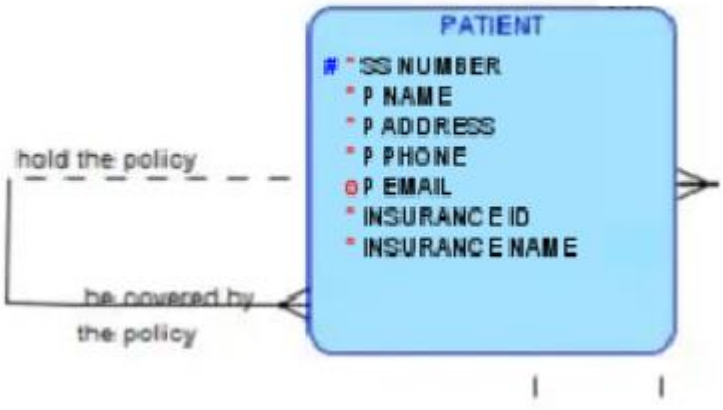


Step 6: Recursive Relationships

Some patients in the patient entity may be part of the same family and be covered by the same insurance – we would like to designate a field in the patient entity showing who is the insurance holder for each patient – this field would be the patient ID number of the person holding the insurance for the family.

Modify the ERD to include a recursive relationship on the Patient entity showing the insurance owners role.

Answer:

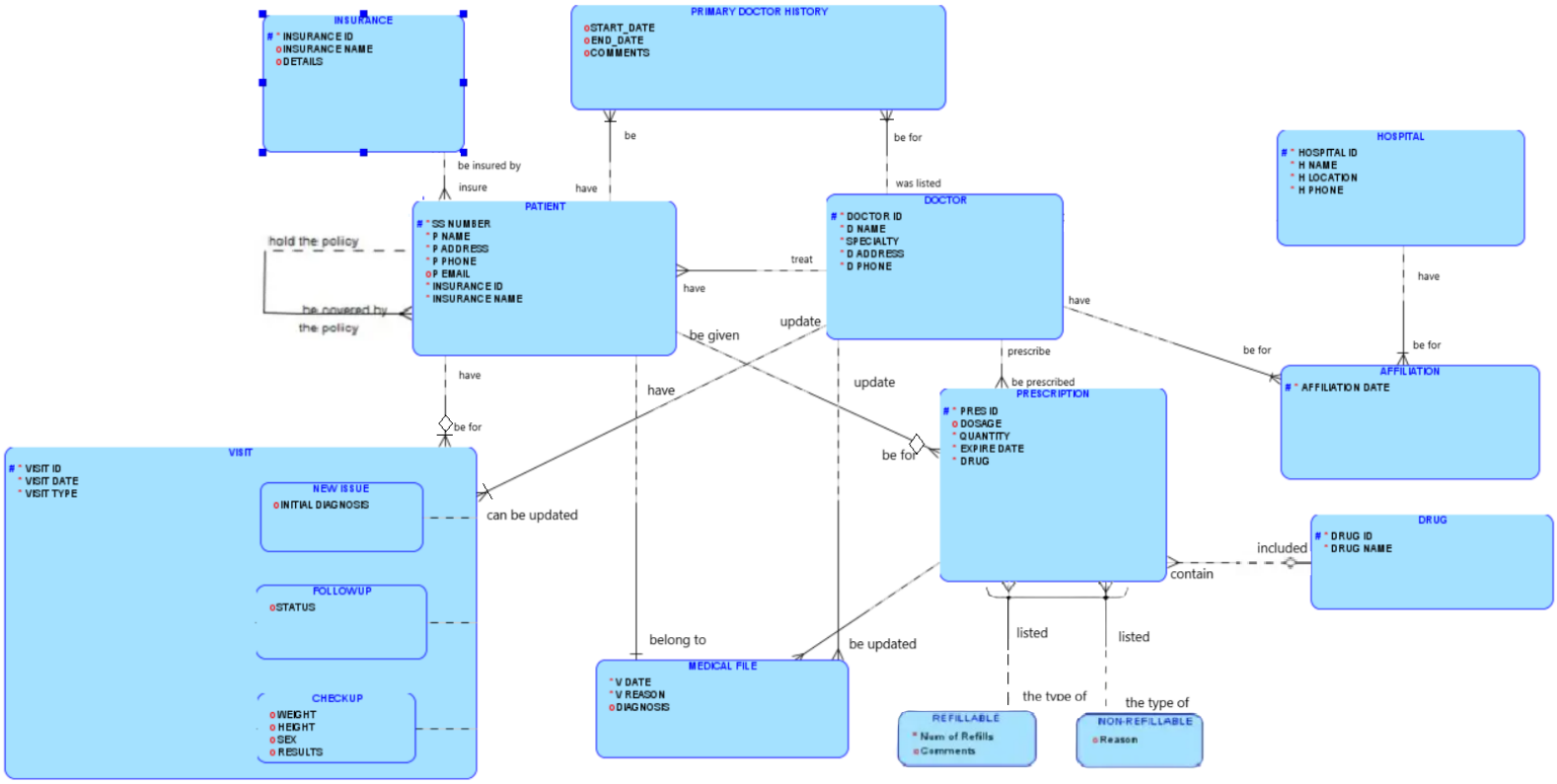


Step 7: Modeling Historical Data

For use in analyzing providers (doctors) and their effectiveness – if a patient changes the primary care doctor, we would like to be able to keep track of these changes. This will also aid in patient care tracking throughout their life. We would like to be able to keep a record of each patient's charts and which doctors may have provided information on them.

Modify the ERD to include an entity showing a history of previous primary care doctors and the dates that the doctor was assigned to a particular patient.

Answer:



Step 8: Basic Mapping

Transform the HealthOne database entities into table diagrams – use suitable naming conventions. Transform relationships into foreign-key columns. Transform the Visit supertype entity using the single-table implementation. Using the following table diagrams, include as many rows as necessary:

Answer:

PATIENT		
pk	*	SS_NUMBER
fk	*	PR_DOC_ID
	*	P_NAME
	*	P_ADDRESS
	*	P_PHONE
	o	P_EMAIL

DOCTOR		
pk	*	DOCTOR_ID
	*	D_NAME
	*	SPECIALTY
	*	D_ADDRESS
	*	D_PHONE

HOSPITAL		
pk	*	HOSPITAL_ID
	*	H_NAME
	*	H_LOCATION
	*	H_PHONE

PRESCRIPTION		
pk	*	PRES_ID
fk1	*	SS_NUMBER
fk2	*	DOCTOR_ID
	*	DOSAGE
	*	QUANTITY
	*	EXPIRE_DATE

MEDICAL_FILE		
pk	*	SS_NUMBER
fk1	*	DOCTOR_ID
fk2	*	PRES_ID
	*	V_DATE
	*	V_REASON
	o	DIAGNOSIS

AFFILIATION		
pk, fk1	*	DOCTOR_ID
pk, fk2	*	HOSPITAL_ID
pk	*	AFFILIATION_DATE

VISIT		
pk	*	VISIT_ID
fk1	*	SS_NUMBER
fk2	*	DOCTOR_ID
	*	VISIT_DATE
	*	VISIT_TYPE

INSURANCE		
pk	*	INSURANCE_ID
fk	*	SS_NUMBER
	o	INSURANCE_NAME
	o	DETAILS

DRUG		
pk	*	DRUG_ID
	*	DRUG_NAME
fk	*	PRES_ID

PRIMARY DOCTOR HISTORY		
pk, fk1	*	SS_NUMBER
pk, fk2	*	DOCTOR_ID
pk	*	START_DATE
	*	END_DATE
	o	COMMENTS