Chapter 4 Team 10

Sagar Devkate, Carlos Flores, Christina Hyman, Daulet Kapezov, Zallocco Filippo

CS 623: Database Management Systems

Dr. William Buttgieg

October 17, 2022

**Step 4.1** - Map the E-R Diagram



Patient (patientNo, name, address, phone, dateOfBirth, sex, *insuranceCo, policyNo*, relationshipToInsured)

Appointment (*patientNo*, apptdate, appttime, reason, *staffNo*, *visitNo*)

Visit (visitNo, *patientNo*, visitdate*,* visittime, duration, reason, visitType, visitCost, *staffNo, roomNo)*

Staff (staffNo, name, title, specialty, address, phone)

Availability (*staffNo*, availDate, startTime, endTime)

InsurancePolicy (company, policyNo, insuredName, policytype, medicalCoPay, labCoPay, pharmacyCoPay, startDate, endingDate)

DiagnosisMenu (diagCode, diagName)

ProcedureMenu (procCode, procName, cost)

Room (roomNo, roomType, condition)

PrescriptionScript (scriptNo, *visitNo*, dateWritten, itemPrescribed, quanityPrescribed, directions, numberRefills)

LabTest (testNo,  *prescriptionNo*, *testType*, testDate, testTime, cost, result)

PrescriptionMedication (RXNumber, *scriptNo*, drugDispensed, dateDispensed, quantityDispensed, refillsRemaining, cost)

ProcedurePerformed (*visitNo*,  *procCode*, result)

Diagnosis (*visitNo, diagCode*, dateOnset, symptoms, severity, prognosis)

Referral (refNo, *visitNo* refTo, reason)

Bill (invoiceNo, billDate, totalAmount, dueDate, *patientNo,*amountPaid)

Charge (*invoiceNo*, serviceType, serviceDate, amountCharged)

Payment (*invoiceNo*, date, amountPaid, *patientPayer, insuranceCoPayer, insurnacePolNoPayer)*

The Explanation.

The relationship sets are: Have, Covers, Consists, Lists, Makes, Pays, Participates, Creates, Schedules, Executed, Requires, Involves, Provides, Produces, Results, Determines, Recommends, States, Specifies, Follows, Gives.

**Have:** Have is a one-to-many relationship between Patient and Insurance Policy. Therefore, the primary key of the Insurance Policy table is included in the Patient table as a foreign key.

**Covers**: The one-to-many Covers relationship between Insurance Policy and Payment can be represented by placing the primary key of Insurance Policy in the Payment table, so we add insurnacePolNoPayer to that table using italics to show that it is a foreign key.

**Consists**: The Consists relationship is a one-to-many relationship between Bill and Payment. The Consists relationship is represented by placing the primary key of Bill in Payment and forms a foreign key in Payment and is part of the primary key. This is shown by italicizing and underlining the attribute invoiceNo.

**Lists**: The Lists relationship is a one-to-many relatinship. The bill should have foreign key which refers to the charge’s primary key.

**Mak**es: One patient can cover many different payments, so there should be a foreign key in the payment, which refers to the patient.

**Pay** is a one-to-many relationship between the entities Patient and Bill. Because a patient can have multiple bills, we have a foreign key inside the table Bill that refers back to Patient payments

**Participates**: This is a one-to-many relationship because a patient can go to multiple visits in each period. Thereby, patientNo is in the Visit table as a foreign key.

**Creates**: This is a one-to-many relationship between the entities Patient and Appointment. Consequently, patientNo is in the table of Appointment as a foreign key.

**Schedules**: Staff schedule appointments, and Schedules is a one-to-one relationship between the entities appointment and visit. There follows that staffNo is a foreign key inside the table Appointment.

**Execute**: the many-to-one relationship Execute(d) characterizing Visit and Room entities implies that there are several visits per day in a clinic and that each clinic must have one room. Therefore, roomNo will qualify as a foreign key inside the table-Entity.

**Requires**: the many-to-one relationship-Requires featuring the entities Appointment and Staff means that, on any given day, a clinic can have multiple appointments and that for each appointment there is at least one member of the medical staff. Hence, staffNo will be a foreign key inside the Appointment table as it comes from the entity staff.

**Involves**: There exist numerous Visits in a day at a given clinic that a member of the medical staff can have. Like Requires, the relationship-Involves has a many-to-one nature. There cannot be any visit without a physician. Thus, StaffNo will be included inside the table for Visit as a foreign key.

**Provides**: the one-to-many relationship between Staff and Availability indicates that each member of the staff is available multiple times, or shifts, at the clinic. Furthermore, availability is a dependent entity, which stems from the entity staff. Therefore, staffNo is included in the table Availability as a foreign key.

**Produces**: The one-to-many Produces will be represented by a foreign key. Therefore, we need to put visitNo under the PerscriptionScript table and designate it as the foreign key. The visitNo foreign key is indicated by the italics in the schema below.

**Results**: The Results relationship has already been represented by placing the primary key of Visit in ProcedurePerformed. This is shown by placing visitNo under the ProcedurePerformed table and designating it as a foreign key and part of the primary key by italicizing and underlining visitNo in the schema below.

**Determines**: The Determines relationship has already been represented by placing the primary key of Visit in Diagnosis. This is shown by placing visitNo under the Diagnosis table and designating it as a foreign key and part of the primary key by italicizing and underlining visitNo in the schema below.

**Recommends**: The Recommends relationship has already been represented by placing the primary key of Visit in Referral. This is shown placing visitNo under the Referral table and designating it as a foreign key and part of the primary key by italicizing and underlining visitNo in the schema below.

**States**: The states relationship is one-to-many. We represent this relationship by placing a foreign entity in the table of Prescription Script representing many Lab Test. The foreign key will be testNo

**Specifies**: The Specifies relationship is one-to-many. This is represented using a foreign key of RXNumber within the prescription script table.

**Follows**: The follows relationship is many-to-one. We represent this relationship by placing the “one” entity Procedures Menu in the table for “many” side, this will make procCode the foreign key.

**Gives**: The Gives relationship is many-to-one. We represent it by placing a “one,” Diagnosis Menu in the table for the “many” side, diagCode will be the foreign key.

A picture containing text

Description automatically generated