Connor Shields
CS 340
Project Step 2
5/2/18

Medical Office Database

**Project Outline**

I will be making a database that represents important information that might be stored in the database of a medical office. In my case, the entities would at least include Patients, Doctors, Appointments, and Insurance Carriers.

**Database Outline, in words**

The entities in my database are:

1. Patient – patients are the people who go to the medical office for treatment. They have a relation with every other entity. Patients have the following attributes:
    a. id: an integer that is an automatically incremented id number. It is the primary key and can't be null
    b. fname: First name of the patient, a string with max 255 chars. Can't be null.
    c. lname: Last name of the patient, a string with max 255 chars. Can't be null.
    d. Primary Provider: an integer that is a foreign key to the ID of the provider to which the patient belongs. Can't be null, and it must be a provider that exists in the database.
    e. Insurance Carrier: insuranceID of the patient's insurance carrier. It is an integer that Defaults to null. The number must point to a carrier that already exists in the database.
    f. Treatments completed: An integer -- number of incomplete appointments the patient is attached to. Defaults to 0, but cannot be null.
    g. Active: a boolean that represents the patient's status – true (1) if the patient is an active patient, and false (0) if the patient is not active. A patient who has had an appointment in the last 3 years should have an Active status of true. May also be set to false if the patient decides that she doesn't wish to visit this office anymore. Defaults to false. Can't be null.

2. Providers – The doctors who work in the medical clinic. They have a relationship with the patient, as well as the treatments they can perform.
    a. provider_id: an auto-incrementing integer, which is the primary key and can't be null
    b. fname: first name of the doctor, a string with max 255 chars. Can't be null.
    c. lname: Last name of the doctor, a string with max 255 chars. Can't be null.
    d. patients: number of patients who are linked to this provider. An integer. This number defaults to 0, but it cannot be null.
    e. specialty: specialty of the doctor. A string of max 255 characters that can't be null. It defaults to 'General'.

f. room number: The number of the room that this provider operates in. An integer that cannot be null. This room number is specific to this provider (can't be shared). Must exist in the database.

3. Appointments – the appointments that are performed by a provider for a patient.
    a. idNum: the distinct id number of the appointment. An auto-incrementing integer that can't be null. Primary key.
    b. provider: id of the provider associated with this appointment. An integer that can't be null. Must already exist in the database.
    c. patient: integer representing the id of the patient associated with this appointment. Can't be null. Must already exist in the database.
    e. cost: a float(10, 2) representing the total cost of this appointment. Defaults to 0. Can't be null.
    f. Insurance portion: a float(10, 2) representing the portion that insurance will pay. Cannot be null. Defaults to 0.
    g. Patient portion:a float(10, 2) representing the portion that patient will pay. Cannot be null. Defaults to 0.
    h. Status: enumerated data type: defaults to "scheduled". Can also be "complete" or "broken". Cannot be null.
4. Insurance Carrier – the carrier that can be associated with a patient.
    a. subscribers: an integer representing the number of subscribers to this plan. Cannot be null. Defaults to 0
    b. coverage: an integer representing the percent that this plan will cover for every patient appointment. Cannot be null. Defaults to 0.
    c. name: name of the insurance company. Cannot be null. A string of max 255 characters.
    d. Carrier ID: Auto-incrementing integer representing the ID that the insurance provides to be used to associate it. Cannot be null. This is the primary key.
5. Room: The room that a provider uses to perform their operations.
    a. Room Number: an integer. Cannot be null. Does not have a default value. This is the primary key.
    b. Provider Number: an integer representing the ID number of the provider associated with this room. Can be null. Can only be occupied by one provider. Must exist in the database.

**The Relationships in my database are:**

1. A patient has a primary provider. A patient can only have one *primary* provider, but a provider can have many patients. Therefore, the patient and provider entities are a one-to-many relationship

2. Appointments have a patient. An appointment can only have one patient associated with it, but a patient can have many appointments that they're scheduled for. This is a one-to-many relationship between the patient and appointment entities.

3. Patients have insurance: A patient can have many insurance carriers (primary, secondary, tertiary, etc), and an insurance carrier can have many patients associated with it. Therefore, this is a many-to-many relationship.

4. Provider to Room: A provider can only have one room, and a room can only be associated with one provider at a time. Therefore, this is a one-to-one relationship.

5. Appointments are covered by insurance carriers. An appointment can be covered by many insurance companies, and many insurance companies can cover many appointments. Therefore, this is a many-to-many relationship.

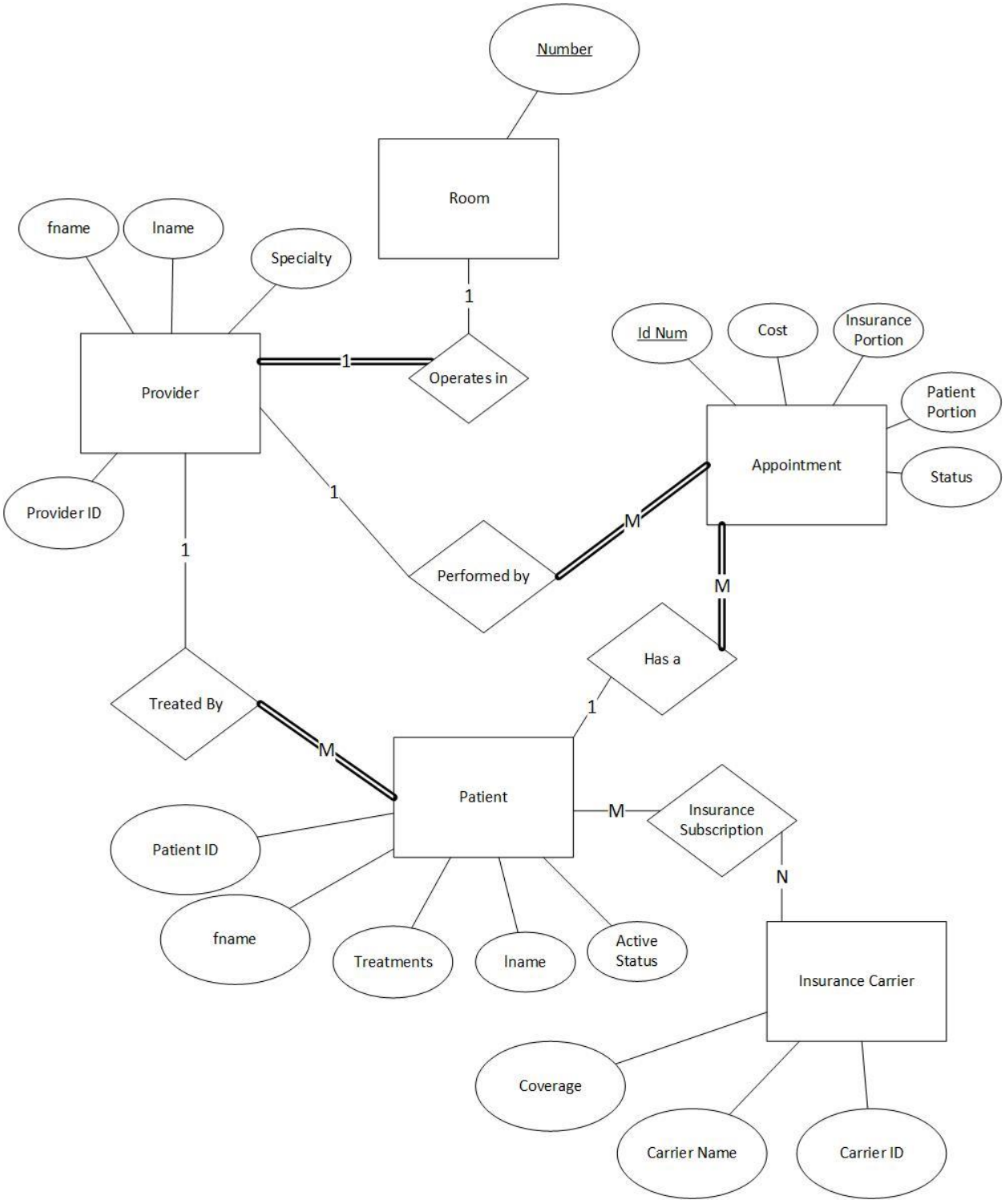
Changes made based on feedback:

-The only feedback I received, aside from adding data types, was to consider whether or not a patient should exist if she doesn't have an existing appointment. I decided to keep my outline the way it was, because many medical offices may have patients (or potential patients) who have created an account with the office, but still haven't scheduled an appointment. Instead of adding a constraint to the patient field regarding their appointments, I decided to add another field, the Active property. This way, it's easy to keep track of patients who are "active" and those who are not, which directly correlates to whether or not they have had a recent appointment, or if they've had an appointment at all.

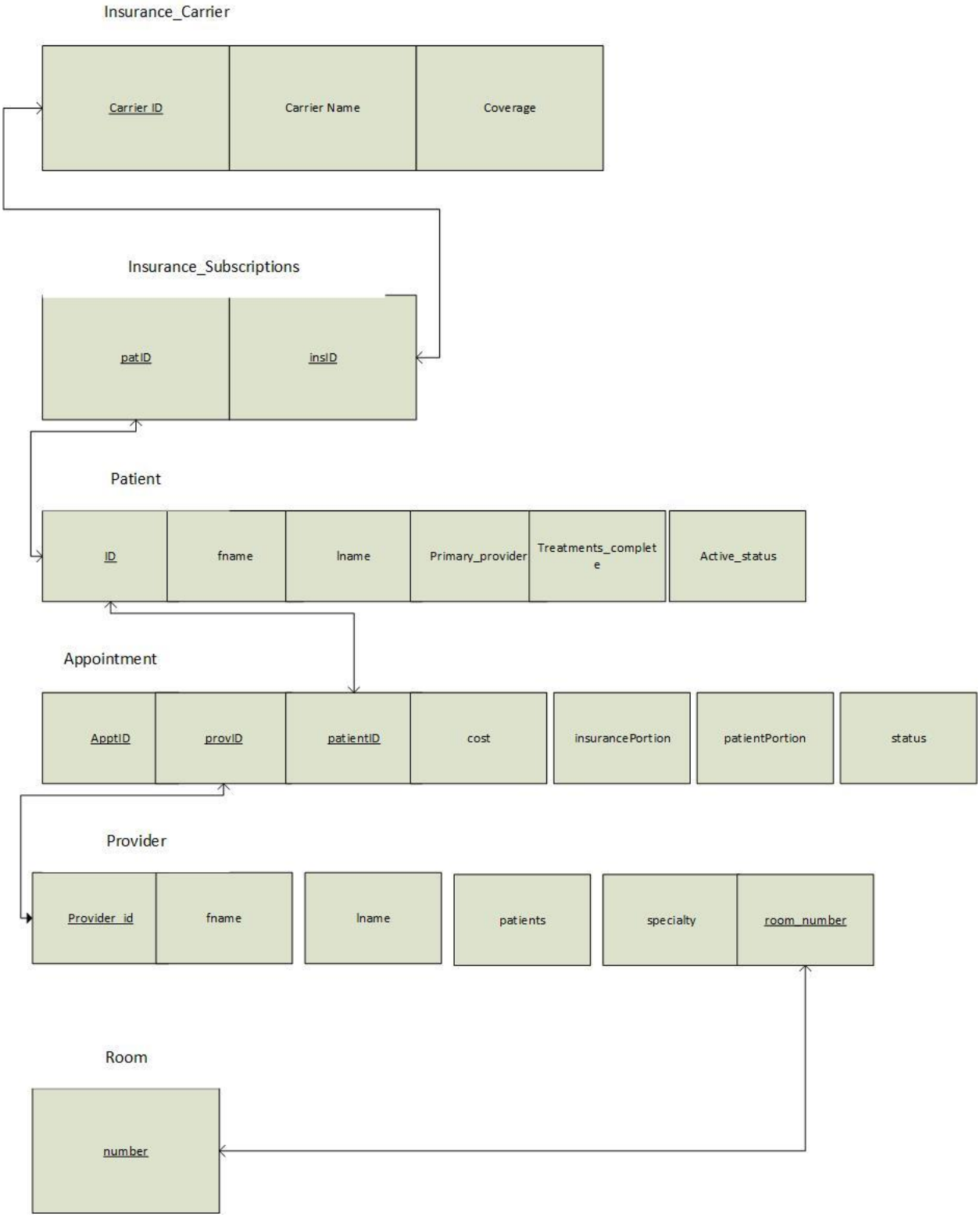
-In my previous outline, the appointment entity had an attribute "insuranceId", which linked it to an insurance carrier. I chose to drop this relationship, and instead opted to rely on the connection to the appointment through the patient. The only reason to have the insurance carrier field attached to the appointment was to track the percentage of payment, but the same effect can be achieved by selecting the insurance's coverage through the linked patient. Doing it this way might call for a slightly more complicated query, but it keeps the schema simpler.

-

**ER Diagram:**


Number
Room
fname
lname
Specialty
1
Provider
1
Operates in
Id Num
Cost
Insurance Portion
Patient Portion
Appointment
Status
Provider ID
1
1
Performed by
M
M
Has a
Treated By
1
M
Patient
M
Insurance Subscription
N
Patient ID
fname
Treatments
lname
Active Status
Insurance Carrier
Coverage
Carrier Name
Carrier ID

**ER Schema:**


Insurance_Carrier
Carrier ID
Carrier Name
Coverage
Insurance_Subscriptions
patID
insID
Patient
ID
fname
lname
Primary_provider
Treatments_complete
Active_status
Appointment
ApptID
provID
patientID
cost
insurancePortion
patientPortion
status
Provider
Provider_id
fname
lname
patients
specialty
room_number
Room
number