Spark Pharmacy Prescription Management System

Created by Kathryn Flinn and Jahmel Phillips

LINK TO APP: classwork.engr.oregonstate.edu:50348/

Step 3 Draft Feedback:

From a TA via Canvas:

Matthew Harangody offered the following feedback:

> "Per the project guidelines you actually need to be able to do full CUD on your intersection table (this is the only entity you need CUD on). My advice to your group: to save both time coding and to meet the project guidelines, eliminate the CUD on the other entities and implement it on PrescriptionMeds. It is totally fine to leave it, but it will just take you more time to implement (although you do 100% still need to implement CUD on the PrescriptionMeds entity).
>
> Also, please add citations on both the DDL and DML file even if you did not use any outside resources."

From Peers on Ed Discussion (edited for brevity):

1. Alexander Lott noted the following:

   > "While there is an M:M relationship between the Meds table (which has UI fields for updating) and the Prescriptions table, there are no SQL statements in the DML that would update the PrescriptionMeds intersection table to reflect any updates to the Meds table."

   Alexander also had the following suggestions:

   > i. "[Giving the column names] aliases so that their displayed names are more user-friendly (e.g. displaying "dosageForm" as "Form of Dosage" in the Meds table)."
   >
   > ii. Removing the extra column header overhang that is present in all tables except Prescriptions.
   >
   > iii. "[Replacing the update Med menu] with "Edit Quantity" buttons/fields alongside each row of the table, similar to the delete buttons on the Prescriptions table"

2. Lyle Feinberg had the following notes:

   > "I found it a bit unintuitive to track a prescription to it's medication, and it may worth adding custom columns to some of the tables to make it more digestible. But I know that is not what this assignment is about, overall great job!!"

Lyle also noted that our Update operation is not on the intersection table or its dependent tables, so it does not meet the requirement that Update functionality must be implemented for an M:M relationship.

3. Michael Valderrama, like the others, noted the absence of Update functionality for an M:M relationship, and suggested implementing a confirmation dialogue for the Delete functionality to prevent accidental deletions.
4. Joseph Musgrove noted that our Update and Delete functionality are not made for our intersection table, and suggested stylizing our website a bit more.

Actions Based on Feedback:

Based on the feedback, we decided to implement the following changes:

1. **Changes to CUD functionality:** we decided to scrap our original CUD functionality and instead implement it for the PrescriptionMeds table *only*.
2. **Tweaks to UI:**
    a. Removing the column overhang
    b. Generally cleaning up the website and adding unique CSS rules.

Future Actions:

These are actions we hope to implement, but did not have enough time for before the due date.

1. **Updates to SELECT functionality for clarity**
    a. Implementing aliases for certain columns for clarity.
    b. Implementing a JOIN that would allow the Prescriptions page to show all associated PrescriptionMeds.
        i. This would negate the need for a separate PrescriptionMeds page.

Step 2 Draft Feedback:

From a TA via Canvas:

Matthew Harangody offered the following feedback:

"Make sure to add descriptive comments in your DDL. For example: -- Creates Patients table, etc.

The prescriptionID fk in sales needs to be marked unique in order to enforce the one to one relationship."

From Peers on Ed Discussion (edited for brevity):

1. Alexander Richmond pointed out some discrepancies between our database outline, ERD, and schema:

   > "In the outline, the Prescriptions table is missing its relationship to the Sales table. In the outline, the Sales table specifies a relationship with Patients, but this is not shown in the ERD or schema."

   Additionally, Alexander suggested adding a uniqueness constraint to the Meds table to prevent duplicate data:

   > "Not exactly a dependency issue, but records in the Meds can be duplicated due to a lack of any unique constraints. This may lead to duplicated data. While there may be instances of medications having the same name with different dosage forms, a useful constraint would be to ensure those two values combined are unique (similar to a first name + last name as a unique full name)."

   Lastly, Alexander suggested splitting the Meds.dosageStrength attribute into two attributes to make it easier to interact with the attribute through sorting or filtering:

   > "dosageStrength as a varchar can lead to inconsistent data entry that is difficult to filter and sort. A more consistent way to enter this would be with 2 attributes: dosageStrength as an integer, and dosageUnit as a varchar."

2. Blaise Skoletsky said that most of our work looked good, but pointed out that "there could be issues with deletion, due to the absence of Cascade statements."

3. Grace Meyer also pointed out the differences between the outline, ERD, and schema:

   > "In the overview, you said there is a 1:M relationship between patients and sales, but there is no line connecting the patients table directly to the sales table. The relationship is instead facilitated through the Prescriptions table. Your outline says this: "This relationship is facilitated through including a patientID foreign key in the Sales table." Neither diagrams show a patientID in the sales table; instead, it is in the prescription table."

   Grace also suggested changing the 1:1 relationship between Prescriptions and Sales to a 1:M relationship to account for Prescriptions that have multiple refills and reduce data redundancy.

4. Ryan Davidson pointed out that there were differences between the overview, ERD, and schema, but suggested reviewing Alexander and Grace's reviews instead of reiterating what had already been said. Ryan also suggested changing Patients.phoneNumber's type to VARCHAR or BIGINT, as the default INT type is too small to support numbers greater than 2,147,483,647.

Actions Based on Feedback:

Based on the feedback from our peers, we decided to implement the following changes:

1. Editing the database overview such that it aligns with the ERD and schema. The ERD and schema were changed in Step 2 of the draft, but some of these changes were not reflected in the overview. Consequently, we will go back to make sure the attributes and relationships in the overview match those in the ERD and schema.
2. Changes to attributes and relationships:
    a. Making an attribute called Meds.dosageUnit to support Meds.dosageStrength. Meds.dosageStrength can then have a DECIMAL type, and Meds.dosageUnit can be a VARCHAR attribute to indicate the units attached to the dosageStrength (e.g. mg, mL, etc.)
    b. Adding a uniqueness constraint to the Meds table to prevent duplicate data. For instance, adding a constraint such that the combination of the medication name, dosageStrength, dosageType, and dosageUnit must be unique could help reduce possible data redundancy.
    c. Changing Patients.phoneNumber's type to be BIGINT to allow it to support all 10-digit phone numbers.
    d. Moving the numRefills attribute from PrescriptionMeds to Prescriptions, since Sales interacts directly with Prescriptions as opposed to PrescriptionMeds.
    e. Changing the Prescription and Sales relationship from 1:1 to 1:M, such that the same Prescription can be associated with multiple Sales to account for refillable Prescriptions.
3. Changes to the DDL SQL file:
    a. Adding CASCADE clauses to our DDL SQL document. (add more on this…)
    b. Adding more comments to the document. Additionally, we will add any citations as needed, or indicate that no extra sources were used and that the work is original.

Step 1 Final Draft Feedback:

From a TA:

"Great work Group 18!

Please include a Citations section at the bottom of your report, even if you didn't use any. The rubric is confusing which is why I did not take any points off this time.

Please reach out to me via teams with any questions or concerns.

Thanks!"

Action Based on Feedback:

Based on the feedback, we decided to add a Citations section to the document. Although we have not used any sources yet, we can add to this section in the future as necessary.

Independent of the feedback, we also decided to implement the following changes:

1. We **removed the SaleItems table** to simplify our project.
2. Because of the removal of the SaleItems table, we also **removed the M:N relationship between Sales and Meds**, such that Sales and Meds can only interact indirectly through the Prescription table now.
3. This change also led us to **remove the PatientID attribute from the Sales table** to eliminate redundancy between the Sales and Prescriptions tables and to avoid potential delete anomalies.
4. We also **added a subTotal attribute to the PrescriptionMeds table** to allow the Sales table to calculate the total cost of a Sale by finding the sum of all PrescriptionMed subtotals associated with a given Prescription.
5. Because of these changes, we also **updated the ER Diagram.**

Step 1 Rough Draft Feedback:

From a TA:

"Great start group 18!

Feedback

I would consider removing some entities from your outline for the scope of this course and the shortened summer term. Insurance type would be an easy one to remove at this point.

The reason I suggest this is because, per the project requirements:

> "Put another way, if you had 4 entities that were implemented as 5 tables in a database, then we expect roughly 5 web app pages as a front end UX for each of the 5 tables in your database.
> One exception is oftentimes it works better for the UX to have a single web page for a Many-to-Many relationship between 2 tables (so the user doesn't have to view two pages to complete a transaction in both tables). So in that case, if you had 4 entities that were implemented as 5 tables, with 1 many-to-many relationship between 2 of those tables, and the 2 tables in that M:M were managed on a single web page, then we expect 4 web pages in the project. "

Since there is a 1:1 relationship between sales and prescriptions, you only need totalAmount in on of those tables. Sales probably makes more sense. But if one is an estimated amount and the other is the actual sale amount it works the way it is.

Please reach out to me via Teams if you have any questions.

Thanks!"

From Peers:

1. "Hey Kathryn and Jahmel! Your project looks great and I appreciate how detailed your database outline is. The main thing that I might change is the Entity-Relationship diagram. I think it might be important to highlight all of the attributes of the entities in the diagram. You did a great job of highlighting the relationships between the entities though, and overall I like the direction for your project. Great work!"

2. "Hey Kathryn and Jahmel,

   I really appreciate the amount of effort you guys put towards each relationships that works with the Patients and the prescriptions, there is M:N relationships and it really shows the connection between each of these entities. I also noticed you utilized every form of relationship, from 1:1, 1:M, and M:N highlighting how each part connects toward the patients and their medications and everything revolving around them. I feel there isn't much for me to pick, I would say maybe with the Diagram there could be distinctions to be made to show the relationships in each one, even though you mentioned them prior in the earlier section maybe the diagram can reflect those changes as well? Overall Great job, I appreciate it."

3. "Hello Kathryn and Jahmel,

   I read your draft about Spark Pharmacy's Prescription Management System. Thank you for the clear, organized submission. This is my thorough feedback, which comes from the project review guidelines.

   Overview - What problem does this solve?
   Your overview explains the problem well - Spark Pharmacy fills about 300 prescriptions each day - it serves around 10,000 patients. You propose a database-backed website - this site helps manage patient records, medications, prescriptions, insurance along with sales. The proposal includes PrescriptionMed in addition to SaleItem for many-to-many relationships. That shows careful design.

Suggestion - Briefly mention the difficulties Spark Pharmacy now faces. For example, the pharmacy may track prescriptions or sales in ways that do not work well or have risks; this will show the need for a database-backed site more clearly.

Use of Specific Facts
You support your proposal with facts - these facts include the number of prescriptions per day and the size of the patient group. This shows the database's scope plus supports the need for a well-made solution.

Entities - Do at least four describe single concepts?
You named seven entities. Each one represents a single idea:

Patients

Meds

Prescriptions

PrescriptionMed (junction)

InsuranceType

Sales

SaleItem (junction)

Each of these entities works for a main part of the system - this part is good.

Entity Details - Purpose, Attributes, Relationships
Each entity has a plain purpose. It has attributes with data types and rules. The relationships are also clear.
I particularly liked how you wrote about the relationships and the purpose of foreign keys. That made it easy to follow.
Small suggestion - You might want to check the one-to-one relationship between Sales besides Prescriptions. Does this always hold in actual use?

Relationships - One-to-Many also Many-to-Many
You used one-to-many and many-to-many relationships correctly. You also made proper junction tables, which are PrescriptionMed or SaleItem.
You explained everything well with foreign keys. There are no problems here.

ERD - Logical View
The PDF did not contain the ERD. But based on your descriptions, the structure sounds logical and well-planned. When you upload your ERD, make sure it matches the writing - it should also include correct cardinalities as well as relationship lines.

Consistency
The names are consistent.

Entities are plural (for example, Patients, Meds)

Attributes are singular (for example, firstName, dosageForm)

Capitalization is right and consistent.

Watch for a few small formatting problems, which include spaces between bullets and attribute explanations that do not match.

Final Comments
Your draft is very good. It is well thought out, detailed as well and structured. When you upload the ERD, plus perhaps tighten the explanation of why the system is needed now, you will have a very good base for the rest of the project.

Originality Statement
This feedback comes from my work.

Best,
Ben."

4. "Hi Group 18,

   I thought your idea was unique and interesting.

   I think the overview does a good job describing the problem that needs to be solved by the database. It sounds like this database will facilitate all pharmacy operations by tracking patients, their medications/insurance, and interactions between pharmacy staff and those patients. The overview does do a good job of giving us a scale at which the database is to be used. The only thing I might add would be to alter the term "regular patients." I would guess that would mean yearly patients for the pharmacy, but it might be a good addition to add a term to spell out what that means.

I think that the entities are well done in this proposal. The database tracks patients, medications, insurance types, and sales. Additionally, there is a table between the medications and prescription tables as well as a table between the sales and medications tables to facilitate M:M relationships. The only alteration I would suggest is considering making the patients able to have multiple insurance types. Aside from that, this structure is intuitive and makes sense to me.

I think that the attributes for each of these entities are relevant to the data they are describing. Each entity table contains several attributes that list important information like medication names, sale dates, and insurance discount rates. I think that all of the most relevant attributes are accounted for but the list could be expanded if the pharmacy needed to track additional things. I think it also is appropriate for the doctor attribute in a prescription to never be able to be set to NULL.

The relationships between the entities make sense. Each patient has one Insurance type, and may have many sales or prescriptions or both through the optional M:M relationships. Additionally, medications can optionally be part of many prescriptions and sales, and sales or prescriptions can have at least one to many medications. This is denoted by the two M:M relationships between these medications and prescriptions/sales. These relationships make sense based on my limited knowledge of pharmacy operation.

I think that the naming conventions are well done. They appear to follow a plural scheme for entity tables as well as a Camel Case type naming for attributes. The only suggestion I would make would be to add a more descriptive name to the first attribute in the "PrescriptionMed" table than ID.

I think this proposal was really interesting. I do not know much about the inner workings of a pharmacy, but I could see how the design of this database could solve a real problem that exists in that industry. All of my suggestions are just small things and feel free to disregard them if needed. Overall, I think this proposal is great.

All content was written by me. Structure was sourced from the Project Step 1 Review Canvas Page (I can provide a citation if needed.). Post is replying to the document posted above."

Actions Based on Feedback:

Based on the feedback we received, we decided to take the following actions:

1. Removing the InsuranceType entity to simplify our project.

2. Updating the ERD to show the attributes of each entity to make the relationships between entities clearer.
3. Removing the totalCost attribute from our Prescription entity.
4. Adding more detail to the project overview to illustrate why our fictional pharmacy would benefit from a database-driven backend, and editing vague wording.
5. Making the wording of entity attribute descriptions more consistent.

We decided **against** implementing the following suggestions:

1. Allowing Patients to have multiple InsuranceTypes. Since we removed the InsuranceType entity, this suggestion is no longer applicable to our project.
2. Changing the 1:1 relationship between Sales and Prescriptions. For simplicity, we wanted this to stay as a 1:1 relationship.

Upgrades to the Draft Version:

As mentioned above, we will be updating the draft document by doing the following:

1. Removing all mentions of the InsuranceType entity, including removing it from the ERD.
2. Updating the ERD to include each entity's attributes.
3. Removing the totalCost attribute from the Prescription entity.
4. General editing for clarity and consistency, specifically in the Overview and Database Outline sections.

Overview:

Spark Pharmacy is a pharmacy based in Corvallis, Oregon that fills an average of 300 prescriptions per day and serves about 10,000 patients. Spark Pharmacy must keep track of high volumes of prescription and medication information, which is not feasible to do with physical paper records alone. In light of this, Spark Pharmacy is looking to develop a database-driven website to keep track of the following information: current *Patients*, the *Meds* the pharmacy supplies, and individual *Prescriptions*, each of which will contain at least one *PrescriptionMed*. Additionally, the pharmacy needs to keep track of *Sales*.

Database Outline:

Patients: contains information about each patient that the pharmacy has seen.

- patientID: INT(PK); Auto-incremented, Not NULL | Unique ID assigned to each patient.
- firstName: VARCHAR(45); Not NULL | A patient's first name.
- lastName:VARCHAR(45); Not NULL | A patient's last name.
- birthDate: DATE; Not NULL | A patient's date of birth.

- phoneNumber: BIGINT | A patient's 10 digit phone number.
- emailAddress: VARCHAR(45) | A patient's email address.

Relationships:

- 1:M relationship with Prescriptions: each Patient can have many prescriptions, but each Prescription can only be associated with a single Patient. This relationship is facilitated using a patientID foreign key *in the Prescriptions table*.

Meds: contains information about each medication the pharmacy can provide. Note that separate dosageForms of the same medication will be listed as different entries in the table.

- medicationID: INT(PK); Auto-incremented, Not NULL | Unique ID assigned to each medication.
- name: VARCHAR(45); Not NULL | The name of the medication.
- dosageForm: VARCHAR(45); Not NULL | The form the medication comes in (e.g. tablet, capsule, liquid).
- dosageStrength: DECIMAL; Not NULL | Amount in each dose (e.g. 100, 12.5, etc.).
- dosageUnit: VARCHAR(2); Not NULL | Unit for dosage (e.g. "mg", "mL", etc.)
- quantity: INT; Not NULL | Gives the total quantity of this medication held at the pharmacy.

Relationships:

- M:N relationship with Prescription: each Med can be part of many different Prescriptions, and each Prescription can have multiple Meds. This relationship will be facilitated through PrescriptionMed (described in more detail below).
    - This results in a 1:M relationship between Meds and PrescriptionMeds, as a Med can be associated with many different PrescriptionMeds, but each PrescriptionMed is only associated with a single Med.

Prescriptions: contains information about each individual prescription filled by the pharmacy.

- prescriptionID: INT(PK); Auto-incremented, Not NULL | Unique ID assigned to each prescription.
- patientID: INT(FK); Not NULL | ID associated with the patient the prescription is for.

- doctorName: VARCHAR(45); Not NULL | Name of doctor prescribing meds (e.g. Dr. Smith).
- dateIssued: DATE; Not NULL | The date the prescription was originally issued.
- numRefills: INT; Default is 0, Not NULL | Number of available refills for the given Prescription.

Relationships:

- M:N relationship with Meds: each Med can be part of many different Prescriptions, and each Prescription can have multiple Meds. This relationship will be facilitated through PrescriptionMed (described in more detail below).
    - This results in a 1:M relationship between Prescriptions and PrescriptionMeds, as a Prescription can be associated with many different PrescriptionMeds, but each PrescriptionMed is only associated with a single Prescription.
- M:1 relationship with Patients: a prescription can only be associated with a single Patient, but a Patient can have several Prescriptions. This relationship is facilitated through the use of the patientID foreign key.
- 1:M relationship with Sales: each Prescription can be associated with several Sales, but each Sale is only associated with a single Prescription.

PrescriptionMeds: holds information on each medication in a given prescription.

- ID: INT(PK); Auto-incremented, Not NULL | Unique ID assigned to each PrescriptionMed.
- prescriptionID: INT(FK); Not NULL | The ID of the prescription the given PrescriptionMed is associated with.
- medicationID: INT(FK); Not NULL | The ID of the Med the given PrescriptionMed is associated with.
- quantityFilled: INT; Not NULL | Amount given to the patient.
- dateFilled: DATE | The date the prescription was filled by the pharmacy.
- subTotal: DECIMAL; Not NULL | Cost of the given prescriptionMed

Relationships:

- PrescriptionMed supports the M:N relationship between Meds and Prescriptions through the use of two foreign keys: medicationID and prescriptionID.
    - This results in a 1:M relationship between Meds and PrescriptionMeds, as a Med can be associated with many different PrescriptionMeds, but each PrescriptionMed is only associated with a single Med.

    - Additionally, this creates a 1:M relationship between Prescriptions and PrescriptionMeds, as a Prescription can be associated with many different PrescriptionMeds, but each PrescriptionMed is only associated with a single Prescription.

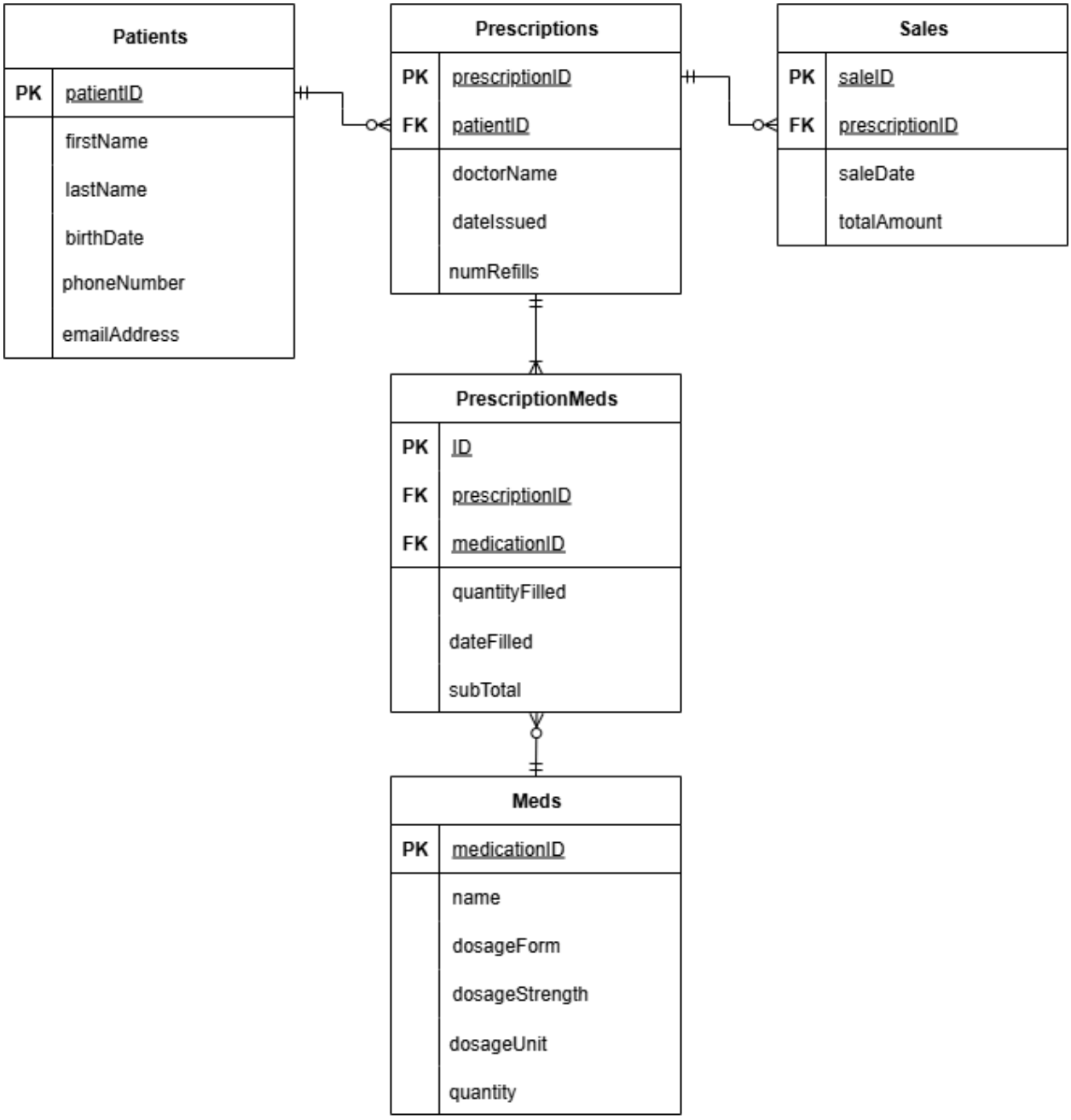
Sales: contains information about the total sale for each prescription.

- saleID: INT(PK); Auto-incremented, Not NULL | Unique ID assigned to each Sale.
- prescriptionID: INT(FK); Not NULL | ID of the prescription associated with the specific sale.
- saleDate: DATE; Not NULL | Date of sale.
- totalAmount: DECIMAL; Not NULL | Total cost of given Sale, as calculated from the subTotal of each PrescriptionMed associated with the Sale's associated Prescription.

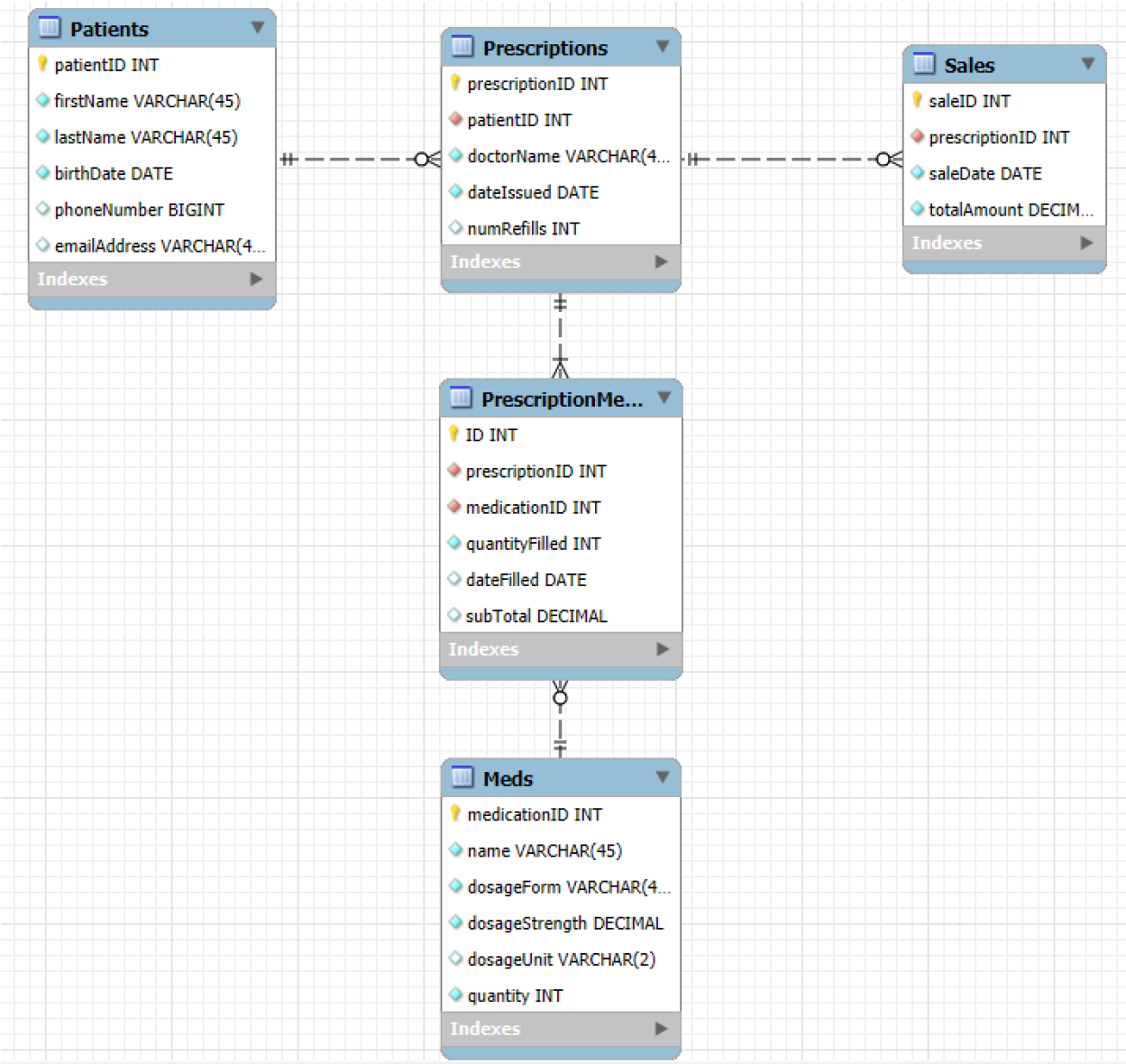
Relationships:

- M:1 relationship with Sales: each Prescription can be associated with several Sales, but each Sale is only associated with a single Prescription.

Entity-Relationship Diagram:



Schema:



Example Data:

Patients					
patientID*	firstName	lastName	birthDate	phoneNumber	emailAddress
1	"John"	"Smith"	"1996-01-10"	1234567890	"john.smith@abc.com"
2	"Carly"	"Jones"	"1972-04-26"	4150000000	NULL
3	"Lily"	"Fisher"	"2002-09-02"	NULL	"lfish@gmail.com"
4	"Riley"	"King"	"1999-02-14"	7891012345	NULL
5	"Chris"	"Jacobs"	"1965-08-29"	NULL	NULL

Prescriptions				
prescriptionID*	patientID*	doctorName	dateIssued	numRefills
1	1	"Abby Connor"	"2025-07-01"	1
2	1	"Abby Connor"	"2025-07-15"	0
3	4	"Stephen Fin"	"2025-06-28"	1
4	5	"Jeff Travis"	"2025-07-24"	1
5	3	"Kelsey Nam"	"2025-06-01"	1

PrescriptionMeds					
ID*	prescriptionID*	medicationID*	quantityFilled	dateFilled	subTotal
1	1	3	50	"2025-07-01"	30.00
2	1	4	25	"2025-07-01"	4.99
3	5	3	50	"2025-06-25"	35.99
4	3	4	20	"2025-06-28"	4.75
5	3	3	20	"2025-06-28"	10.00

Meds					
medicationID*	name	dosageForm	dosageStrength	dosageUnit	quantity
1	"Gabapentin"	"Capsule"	100	"mg"	100
2	"Dicyclomine"	"Capsule"	10	"mg"	250
3	"Ondansetron"	"Pill"	8	"mg"	300
4	"Amoxicillin"	"Liquid"	100	"mL"	125
5	"Lisinopril"	"Tablet"	20	"mg"	200

Sales			
saleID*	prescriptionID*	saleDate	totalAmount
1	1	"2025-07-01"	34.99
2	3	"2025-07-01"	14.75
3	5	"2025-06-28"	35.99
4	1	"2025-07-28"	34.99

Citations

(Add citations as needed)