Design Document

By Kelvin Mafurendi

Video overview: Database design for a Pharmacy Drug Dispensing software

Scope

This project is a database for a Pharmacy Dispensing software. It will include:

- General Drug information like name, type and expiry information
- General patient information like name, allergies, medical aid etc
- A patient's medical history in relation to this particular pharmacy/chain of pharmacies sharing the database
- The medical history will be in the form of transactions performed and prescription given to a particular patient.
- General staff information for the pharmacy and their dispensing history in the pharmacy/group of pharmacy
- General pharmacy identification information like name,location etc
- Prescription information like prescription number, prescribing Doctor's information
- Drug inventory information, which drugs are available in what quantities
- Drug suppliers information
- Drug dispensing activity including date and time, patient etc

However, compliance information and the financial accounting aspects of the pharmacy is out the scope of this database.

Functional Requirements

This database will be able to support:

- CRUD functions for patients,drug,staff members, and suppliers
- Keep track of all staff member dispensing activity on the platform
- Track patient medical history with all pharmacies that have access to the database
- Keep track of drug inventory
- Keep track of patients' prescriptions
- Drug expiration tracking
- The database will not keep track of sales to the extent of financial accounting i.e calculating total sales, profits nor will it store compliance information associated with each of the drugs

Representation

Entities are captured in SQLite tables with the following schema.

Entities

pharmacies

The `pharmacies` table includes:

- `id` specifies the pharmacy ID as `INTEGER`, the `PRIMARY KEY` constraint is applied on this column.
- `name` specifies the name of the pharmacy as `TEXT`, since `TEXT` is appropriate for name fields.
- `phone_number` specifies the contact phone number for the pharmacy as `NUMERIC`, since it is a number but not an integer.
- `email` specifies the pharmacy's email address as `TEXT`, since it is a string of text.
- `address` specifies the physical address of the pharmacy as `TEXT`, since the address is a string of text.
- `state` specifies the state/province the pharmacy is in as `TEXT`, which again is appropriate for string fields
- `country` specifies the country in which the pharmacy is as `TEXT`, since country names are strings.

All columns in the pharmacies table are required hence they are all `NOT NULL`.

patients

Patient information is captured in the `patients` table as follows:

- `id` specifies the ID for each patient as `INTEGER` and has the `PRIMARY KEY` contraint applied.
- `first_name` specifies a patient's first name as `TEXT` since a name is a string of text.
- `last_name` specifies the patient's surname as `TEXT` which is a string of text.
- `DOB` specifies a patient's date of birth as `NUMERIC` this is a date in the format yyyy-mm-dd which should be `NUMERIC` according to SQLite documentation
- `gender` specifies the gender of the patient as `TEXT`, gender is expressed as a string of text.
- `id_number` specifies the identity number/passport number for the patient as `TEXT`, since this can be a mix of numbers and letters. Every patient has got a `UNIQUE` id number.
- `contact_number` specifies the patient's phone number as NUMERIC, since it's a string of numbers and sometimes with a few characters like '+' or '-'.
- `email` specifies the patient's email address as `TEXT`, since it is a string of text characters.
- `allergies` specifies any allegies the patient might have as `TEXT`, since it is a string of text.
- `medical_aid_name` specifies the name of a patient's medical insurance provider as `TEXT`, since it is a name.
- `medical_aid_number` specifies the patient's medical aid number as `NUMERIC`, since different medical aid providers issue medical aid numbers differently it is safe to just make this of `NUMERIC` type.
- `pre_existing_condition` captures the patient's pre-exisitng conditions if any as `TEXT`, since this will be a string of text.

All the other fields are required hence `NOT NULL` except for `allergies`, `medical_aid_name`, `medical_aid_number` and `pre_existing_condition` because not all patients have these. `medical_aid_number` should be `UNIQUE` for each patient.

staff

The pharmacy employees responsible for dispensing drugs are stored in the `staff` table:

- `id` specifies the ID for each staff member as `INTEGER` and has the `PRIMARY KEY` contraint applied.
- `first_name` specifies the first name of a staff member as `TEXT`, since it is a name field.
- `last_name` specifies a staff member's surname as `TEXT`. This too is a name field.
- `pharmacy_id` specifies the ID of the pharmacy for which the staff member works as `INTEGER`. The `FOREIGN KEY` constraint is applied to this column because it references the `id` column in the `pharmacies` table.
- `role` specifies the role of the employee at the pharmacy as `TEXT`, roles are expressed as strings of text.
- `password` specifies the staff member's password when they login to the software as `TEXT`, since it will be a string of mixed characters. In this project, I will overlook the fact that passwords will have to be hashed before being stored on the database.

All columns are required in this table hence they are all `NOT NULL`.

suppliers

Drug suppliers are recorded in the `suppliers` table:

- `id` specifies the ID for each supplier as `INTEGER` and has the `PRIMARY KEY` contraint applied.
- `name` specifies the name of the supplier as `TEXT`, since a name field is a string of text. Supplier names ought to be `UNIQUE`.

- `phone_number` specifies the contact phone number for the supplier as `NUMERIC`, since it is a number but not an integer.
- `email` specifies the supplier's email address as `TEXT`, since it is a string of text.
- `address` specifies the physical address of the supplier as `TEXT`, since the address is a string of text
- `state` specifies the state/province the supplier is in as `TEXT`, which again is appropriate for string fields
- `country` specifies the country in which the supplier is as `TEXT`, since country names are strings.

All fields in this table are required hence `NOT NULL`.

drugs

The drugs information is recorded in the `drugs` table:

- `id` specifies the drug ID as `INTEGER`, the `PRIMARY KEY` constraint is applied on this column
- `name` specifies the name of the drug as `TEXT`, since `TEXT` is appropriate for name fields. EACH drug has a `UNIQUE` name.
- `supplier_id` specifies the ID of the drug supplier as `INTEGER`. A `FOREIGN KEY` constraint is applied on this field because it references the `id` column in the `suppliers` table.
- `stock_number` specifies the unique in store ID for each drug in the pharmacy as `NUMERIC`, this will usually be a long number. Each drug has a `UNIQUE` stock number.
- `form` specifies the form of drug as `TEXT`, since type is expressed as a string of text i.e is the drug a capsule, tablet, liquid etc.
- `batch_number` specifies the drug's batch number as `NUMERIC` because although it isi a number batch numbers are usually formatted differently.
- `dosage` specifies the dosage of the drug as `TEXT`, since this will contain strings which are a mixture of numbers and text.
- `manufacturer` specifies the manufacturer of the drug as `TEXT`, since this is a name field.

- `expiry_date` specifies the expiry date of the drug as `NUMERIC`, since this is a date data type.
- `in_stock` specifies the quantity of the drug that is available in stock as `INTEGER`, because this specifies the number of single units of the drug.
- `requires_prescription` specifies in `TEXT` (yes or no) whether or not a drug requires prescription, again this is a string of text.
- `cost` specifies the unit buying price for the drug from the supplier as `NUMERIC`, since this represents money.
- `price` specifies the unit selling price in the pharmacy as `NUMERIC`, since again this is money.
- `barcode` specifies the barcode for the drug as `NUMERIC` to enable barcode scanning, because it's a string of numbers. The barcode for each drug ought to be `UNIQUE`.
- `location_in_store` specifies as `TEXT` the location in the pharmacy where the drug is kept.

ALL columns are required for each drug hence `NOT NULL`, except on those where the primary and foreign key constraints are applied.

prescriptions

All prescriptions are recorded in the `prescriptions` table:

- `id` specifies the ID for each prescription as `INTEGER`, the `PRIMARY KEY` constraint is applied on this column.
- `patient_id` specifies the ID of the patient as `INTEGER`, this column has a `FOREIGN KEY` constraint as it references the `id` column in the `patients` table.
- `drug_id` specifies the ID of a prescribed drug as `INTEGER`, again a `FOREIGN KEY` constrain is applied here as it references the `id` column in the `drugs` table.
- `diagnosis` specifies the doctor's diagnosis as `TEXT`, since it is a string of text.
- `dosage_duration` specifies the number of days the patient ought to be taking a particular prescribed drug as `INTEGER` since the column represents a whole number of days.

- `date_of_prescription` specifies the date on which the prescription was made as `NUMERIC` since this a date.
- `advice_given` specifies any advice as given to the patient by the doctor as `TEXT`, since advice is a string of text.
- `doctor` specifies the name of the doctor who made the prescription as `TEXT`, since this is a name field.
- `doctor_pr_number` specifies the prescribing doctor's practice number as `NUMERIC`, since this could be a number but formatted differently.

All columns are required hence `NOT NULL` with the exception of `advice_given` as there might not be advice on the prescription. Also those with FOREIGN KEY constraint applied are automatically NOT NULL therefore, no need for redundance.

dispenses

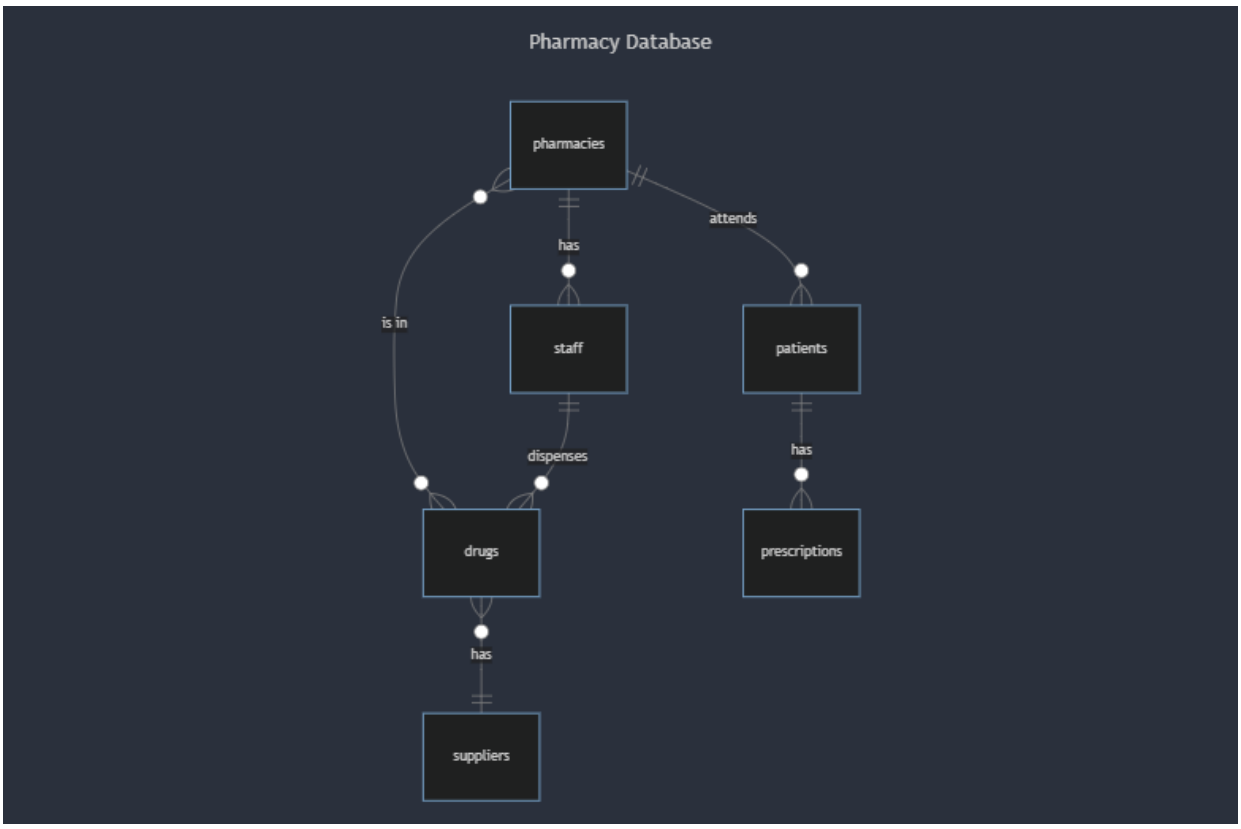
All the drug dispensing sessions are recorded in the `dispenses` table:

- `id` specifies the ID of dispensing session as `INTEGER` and has the `PRIMARY KEY` constrain applied.
- `staff_id` specifies the ID of the staff member who dispensed the drugs as `INTEGER` and the column is constrained by the `FOREIGN KEY` constraint which references the `id` column in the `staff` table.
- `drug_id` specifies the ID of dispensed drug `INTEGER` and the column is constrained by the `FOREIGN KEY` constraint which references the `id` column in the `drugs` table.
- `pharmacy_id` specifies the ID of the pharmacy where the drug was dispensed as `INTEGER` and the column is constrained by the `FOREIGN KEY` constraint which references the `id` column in the `pharmacies` table.
- `prescription_id` specifies the ID of the prescription for which the drug was dispensed as `INTEGER` and the column is constrained by the `FOREIGN KEY` constraint which references the `id` column in the `prescriptions` table.
- `timestamp` specifies the date and time of the drug dispensation as `CURRENT_TIMESTAMP`, because it should record the exact moment at which the transaction was done.

All columns are required but only the `timestamp` column should be marked `NOT NULL` since the rest are automatically unnullable due to the 'key' constraints applied to them.

Relationships

The relationships between the different entities are illustracted by the ER diadgram below:



As illustrated above and with the assumption that this database will be used by a group of pharmacies sharing the same database:

- Each pharmacy in the group could have zero-to-many drugs in their inventory. And the same drug can be in zero-to-many pharmacies. This means that it is possible for a drug to not be available in any pharmacy at a given time and at some other time some or all pharmacies could have a particular drug available. The relationship allows pharmacies to know which among them have a particular drug so that they can refer

their patients if possible or they may have to resale to each other whatever the case maybe.

- At a given time a particular pharmacy could have zero-to-many patients. And a patient can only be in one pharmacy.

- Each pharmacy can employ at a given time zero-to-many staff members who dispense drugs. These could be Pharmacists, Interns etc. And any staff member is assumed to only work for one pharmacy at a given time.
- In a given pharmacy, each staff member can dispense zero-to-many drugs depending on the patient traffic and drug availability.
- Each patient at a given time could have zero-to-many prescriptions and every single prescription can only belong to one patient.
- Each supplier can supply zero-to-many drugs and many suppliers could supply the same drug as well.

Optimizations

In the queries.sql file, I have outlined common queries that users(pharmacy adminstrators,pharmacists etc) might want to perform on the database. Therefore, I have created indexes to optimize these common queries:

- For the `drugs` table, I have indexed the following commonly queried columns based on my queries.sql:
- - `name`,`stock_number`, `expiry_date`,`supplier_id`, `location_in_store`,`in_stock`
- For the `dispenses` table, I have indexed the following columns:
- - `quantity`, `timestamp`, `drug_id`, `patient_id`, `pharmacy_id`, `staff_id`, `prescription_id`
- For the `suppliers` I have indexed the following columns:
- - `name`
- For the `staff` table, the following columns have been indexed:
- - `first_name`, `last_name`
- For the `pharmacies` table:
- - `name`
- For the `patients` table:

- - `title`

Limitations

My database schema does not include the financial accounting aspects which might render it a bit not so useful in a real pharmacy as an all-in-one tool would be more favourable. Although I have elected to use sqlite for this project, a MySQL or Postgresql version of SQL is most appropriate considering the fact that this is supposed to be a shared database among a group of pharmacies. Sqlite would be most appropriate for only one pharmacy where there is no need to replicate the database or a database network that allows concurrent access. Therefore, in the current state, is only useful for individual pharmacy use and can be tranlated to MySQL or Postgresql should scale be a requirement.