

НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ
УНИВЕРСИТЕТ
ФАКУЛЬТЕТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ

Базы данных

Информационная система аптеки

Кондренко К.П., группа 21203

17 марта 2024 г.

Содержание

1	Задание	1
1.1	Описание предметной области	1
1.2	Запросы к базы данных	1
2	Таблицы и связи	3
2.1	Описание таблиц	3
3	Создание таблиц	5
4	Ограничения по поддержанию целостности	9

1 Задание

Разработать структуру базы данных для информационной системы аптеки и реализовать приложение в архитектуре клиент-сервер, выполняющее операции внесения данных в базу данных, редактирование данных и запросы.

1.1 Описание предметной области

Аптека продает медикаменты и изготавливает их по рецептам. Лекарства могут быть разных типов:

1. Готовые лекарства: таблетки, мази, настойки.
2. Изготавливаемые аптекой: микстуры, мази, растворы, настойки, порошки.

Различие в типах лекарств отражается в различном наборе атрибутов, их характеризующих. Микстуры и порошки изготавливаются только для внутреннего применения, растворы для наружного, внутреннего применения и для смешивания с другими лекарствами и мази только для наружного применения. Лекарство различны также по способу приготовления и по времени приготовления. Порошки и мази изготавливаются смешиванием различных компонент. При изготовлении растворов и микстур ингредиенты не только смешивают, но и отстаивают с последующей фильтрацией лекарства, что увеличивает время изготовления.

В аптеке существует справочник технологий приготовления различных лекарств. В нем указываются: идентификационный номер технологии, название лекарства и сам способ приготовления. На складе на все медикаменты устанавливается критическая норма, т.е. когда какого-либо вещества на складе меньше критической нормы, то составляются заявки на данные вещества и их в срочном порядке привозят с оптовых складов медикаментов.

Для изготовления аптекой лекарства, больной должен принести рецепт от лечащего врача. В рецепте должно быть указано: ФИО, подпись и печать врача, ФИО, возраст и диагноз пациента, также количество лекарства и способ применения. Больной отдает рецепт регистратору, он принимает заказ и смотрит, есть ли компоненты заказываемого лекарства. Если не все компоненты имеются в наличии, то делает заявки на оптовые склады лекарств и фиксирует ФИО, телефон и адрес необслуженного покупателя, чтобы сообщить ему, когда доставят нужные компоненты. Такой больной пополняет справочник заказов - это те заказы, которые находятся в процессе приготовления, с пометкой, что не все компоненты есть для заказа. Если все компоненты имеются, то они резервируются для лекарства больного. Покупатель выплачивает цену лекарства, ему возвращается рецепт с пометкой о времени изготовления. Больной также пополняет справочник заказов

в производстве. В назначенное время больной приходит и по тому же рецепту получает готовое лекарство. Такой больной пополняет список отданных заказов.

Ведется статистика по объемам используемых медикаментов. Через определенный промежуток времени производится инвентаризация склада. Это делается для того, чтобы определить, есть ли лекарства с критической нормой, или вышел срок хранения или недостача.

1.2 Запросы к базы данных

1. Получить сведения о покупателях, которые не пришли забрать свой заказ в назначенное им время и общее их число.
2. Получить перечень и общее число покупателей, которые ждут прибытия на склад нужных им медикаментов в целом и по указанной категории медикаментов.
3. Получить перечень десяти наиболее часто используемых медикаментов в целом и указанной категории медикаментов.
4. Получить какой объем указанных веществ использован за указанный период.
5. Получить перечень и общее число покупателей, заказывавших определенное лекарство или определенные типы лекарств за данный период.
6. Получить перечень и типы лекарств, достигших своей критической нормы или закончившихся.
7. Получить перечень лекарств с минимальным запасом на складе в целом и по указанной категории медикаментов.
8. Получить полный перечень и общее число заказов находящихся в производстве.
9. Получить полный перечень и общее число препаратов требующихся для заказов, находящихся в производстве.
10. Получить все технологии приготовления лекарств указанных типов, конкретных лекарств, находящихся в справочнике заказов в производстве.
11. Получить сведения о ценах на указанное лекарство в готовом виде, об объеме и ценах на все компоненты, требующиеся для этого лекарства.
12. Получить сведения о наиболее часто делающих заказы клиентах на медикаменты определенного типа, на конкретные медикаменты.
13. Получить сведения о конкретном лекарстве (его тип, способ приготовления, названия всех компонент, цены, его количество на складе).

2 Таблицы и связи

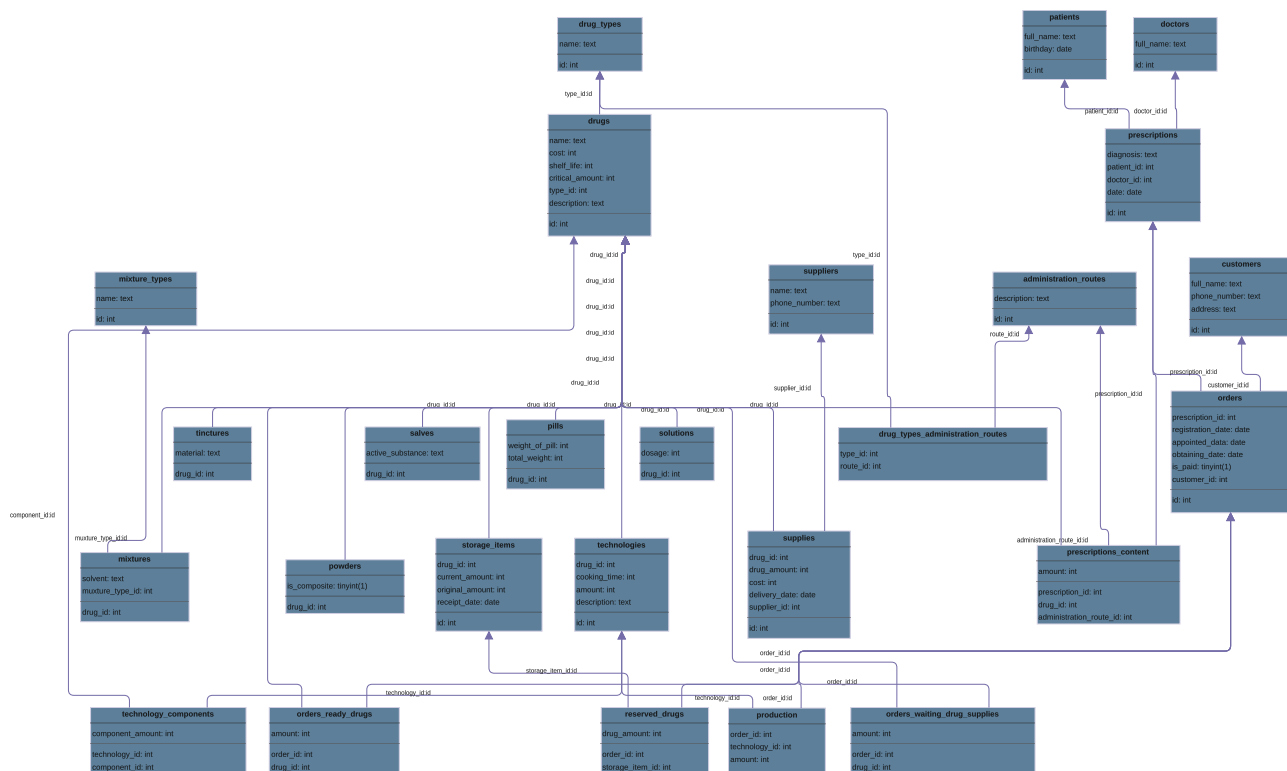


Рис. 1: Графическая схема базы данных

2.1 Описание таблиц

- **administration_routes** — способы применения лекарств (идентификатор способа, название);
- **drug_types** — типы лекарств (идентификатор типа, название);
- **drug_types_administration_routes** — соответствие между типами лекарств и способами их применения (идентификатор типа, идентификатор способа);
- **customers** — клиенты аптеки (идентификатор клиента, ФИО, номер телефона, адрес);
- **doctors** — врачи, которые выписывают рецепты для больных (идентификатор врача, ФИО);
- **patients** — пациенты, то есть те, на кого выписывают рецепты (идентификатор пациента, ФИО, дата рождения);
- **suppliers** — поставщики лекарств в аптеку (идентификатор поставщика, название, номер телефона);
- **prescriptions** — рецепты, выписанные больным врачами (идентификатор рецепта, диагноз, идентификатор пациента, идентификатор врача, дата);
- **mixture_types** — типы микстур (идентификатор типа, название);

- **drugs** — лекарства (идентификатор лекарства, название, стоимость, срок годности, критическая норма, идентификатор типа — из *drug_types*, описание);
- **mixtures** — микстуры (идентификатор лекарства — из *drugs*, растворитель, идентификатор типа микстуры — из *mixture_types*);
- **pills** — таблетки (идентификатор лекарства — из *drugs*, масса одной таблетки, масса пачки таблеток);
- **powders** — порошки (идентификатор лекарства — из *drugs*, составной порошок или нет);
- **salves** — мази (идентификатор лекарства — из *drugs*, действующее вещество);
- **solutions** — растворы (идентификатор лекарства — из *drugs*, концентрация);
- **tinctures** — настойки (идентификатор лекарства — из *drugs*, материал);
- **prescriptions_content** — состав рецептов (идентификатор рецепта — из *prescriptions*, идентификатор лекарства — из *drugs*, количество лекарства, способ применения — из *administration_routes*);
- **storage_items** — позиции лекарств на складе (идентификатор позиции, идентификатор лекарства — из *drugs*, текущее количество лекарства в позиции на складе, исходное количество лекарства в позиции на складе; дата получения);
- **supplies** — поставки лекарств от поставщиков (идентификатор поставки, идентификатор лекарства — из *drugs*, количество лекарства, общая стоимость, идентификатор поставщика — из *suppliers*, дата поставки);
- **technologies** — справочник технологий приготовления лекарств (идентификатор технологии, идентификатор лекарства — из *drugs*, время приготовления, количество приготавливаемого лекарства, инструкция);
- **technology_components** — лекарства, требуемые для приготовления лекарств по технологиям (идентификатор технологии — из *technologies*, идентификатор лекарства, требуемого для технологии — из *drugs*, количество данного лекарства, требуемого для технологии);
- **orders** — заказы (идентификатор заказа, идентификатор рецепта — из *prescriptions*, дата регистрации, назначенная дата получения заказа, реальная дата получения заказа, оплачен ли заказ, идентификатор клиента — из *customers*);
- **orders_ready_drugs** — какие лекарства для заказов уже готовы (идентификатор заказа — из *orders*, идентификатор лекарства — из *drugs*, количество лекарства);
- **orders_waiting_drug_supplies** — поставки каких лекарств нужны для заказов (идентификатор заказа — из *orders*, идентификатор лекарства — из *drugs*, количество лекарства);
- **reserved_drugs** — какие лекарства со склада зарезервированы для заказов (идентификатор заказа — из *orders*, идентификатор позиции склада — из *storage_items*, количество лекарства);
- **production** — какие лекарства сейчас находятся в производстве для заказов (идентификатор заказа — из *orders*, идентификатор технологии приготовления лекарства — из *technologies*, количество процессов приготовления лекарства).

3 Создание таблиц

Листинг 1: SQL-скрипт для создания таблиц базы данных

```
create table if not exists administration_routes
(
    id          int auto_increment
      primary key,
    description text not null
);

create table if not exists customers
(
    id          int auto_increment
      primary key,
    full_name   text not null,
    phone_number text not null,
    address     text not null
);

create table if not exists doctors
(
    id          int auto_increment
      primary key,
    full_name   text not null
);

create table if not exists drug_types
(
    id          int auto_increment
      primary key,
    name        text not null
);

create table if not exists drug_types_administration_routes
(
    type_id int not null,
    route_id int not null,
    constraint drug_types_administration_routes_administration_routes_id_fk
        foreign key (route_id) references administration_routes (id),
    constraint drug_types_administration_routes_drug_types_id_fk
        foreign key (type_id) references drug_types (id)
);

create table if not exists drugs
(
    id          int auto_increment
      primary key,
    name        text not null,
    cost        int not null,
    shelf_life  int not null,
    critical_amount int not null,
    type_id     int not null,
    description text not null,
    constraint drugs_drug_types_id_fk
        foreign key (type_id) references drug_types (id),
    check ('critical_amount' >= 0),
    check ('cost' > 0),
    check ('shelf_life' > 0)
);

create table if not exists mixture_types
(
    id          int auto_increment
      primary key,
    name        text not null
);

create table if not exists mixtures
(
    drug_id          int auto_increment
      primary key,
    solvent          text not null,
    mixture_type_id int not null,
    constraint mixtures_drugs_id_fk
```

```

        foreign key (drug_id) references drugs (id),
        constraint mixtures_mixture_types_id_fk
        foreign key (mixture_type_id) references mixture_types (id)
);

create table if not exists patients
(
    id            int auto_increment
        primary key,
    full_name text not null,
    birthday    date not null
);

create table if not exists pills
(
    drug_id        int not null
        primary key,
    weight_of_pill int not null,
    total_weight   int null,
    constraint pills_drugs_id_fk
        foreign key (drug_id) references drugs (id),
    check ('total_weight' > 0),
    check ('weight_of_pill' > 0)
);

create table if not exists powders
(
    drug_id        int not null
        primary key,
    is_composite tinyint(1) not null,
    constraint powders_drugs_id_fk
        foreign key (drug_id) references drugs (id)
);

create table if not exists prescriptions
(
    id            int auto_increment
        primary key,
    diagnosis text not null,
    patient_id int not null,
    doctor_id  int not null,
    date       date not null,
    constraint prescriptions_doctors_id_fk
        foreign key (doctor_id) references doctors (id),
    constraint prescriptions_patients_id_fk
        foreign key (patient_id) references patients (id)
);

create table if not exists orders
(
    id            int auto_increment
        primary key,
    prescription_id int not null,
    registration_date date not null,
    appointed_data  date null,
    obtaining_date  date null,
    is_paid         tinyint(1) not null,
    customer_id     int null,
    constraint orders_customers_id_fk
        foreign key (customer_id) references customers (id),
    constraint orders_prescriptions_id_fk
        foreign key (prescription_id) references prescriptions (id)
);

create table if not exists orders_ready_drugs
(
    order_id int not null,
    drug_id  int not null,
    amount   int not null,
    primary key (drug_id, order_id),
    constraint orders_ready_drugs_drugs_id_fk
        foreign key (drug_id) references drugs (id),
    constraint orders_ready_drugs_orders_id_fk
        foreign key (order_id) references orders (id),
    check ('amount' >= 0)
);

```

```

create table if not exists orders_waiting_drug_supplies
(
    order_id int not null,
    drug_id  int not null,
    amount   int not null,
    primary key (drug_id, order_id),
    constraint orders_waiting_supplies_list_drugs_id_fk
        foreign key (drug_id) references drugs (id),
    constraint orders_waiting_supplies_list_orders_id_fk
        foreign key (order_id) references orders (id),
    check ('amount' > 0)
);

create table if not exists prescriptions_content
(
    prescription_id int not null,
    drug_id         int not null,
    amount          int not null,
    administration_route_id int not null,
    primary key (prescription_id, drug_id, administration_route_id),
    constraint prescriptions_content_administration_routes_id_fk
        foreign key (administration_route_id) references administration_routes (id),
    constraint prescriptions_content_drugs_id_fk
        foreign key (drug_id) references drugs (id),
    constraint prescriptions_content_prescriptions_id_fk
        foreign key (prescription_id) references prescriptions (id),
    check ('amount' > 0)
);

create table if not exists salves
(
    drug_id int not null
        primary key,
    active_substance text not null,
    constraint salves_drugs_id_fk
        foreign key (drug_id) references drugs (id)
);

create table if not exists solutions
(
    drug_id int not null
        primary key,
    dosage int not null,
    constraint solutions_drugs_id_fk
        foreign key (drug_id) references drugs (id),
    check ((0 <= 'dosage') and ('dosage' <= 100))
);

create table if not exists storage_items
(
    id int auto_increment
        primary key,
    drug_id int not null,
    current_amount int null,
    original_amount int not null,
    receipt_date date not null,
    constraint storage_items_drugs_id_fk
        foreign key (drug_id) references drugs (id),
    check ('current_amount' >= 0),
    check ('original_amount' > 0)
);

create table if not exists reserved_drugs
(
    order_id int not null,
    storage_item_id int not null,
    drug_amount int not null,
    primary key (order_id, storage_item_id),
    constraint reserved_drugs_orders_id_fk
        foreign key (order_id) references orders (id),
    constraint reserved_drugs_storage_items_id_fk
        foreign key (storage_item_id) references storage_items (id),
    check ('drug_amount' > 0)
);

```



```

create table if not exists suppliers
(
    id            int auto_increment
        primary key,
    name          text not null,
    phone_number  text not null
);

create table if not exists supplies
(
    drug_id        int not null,
    drug_amount    int not null,
    cost           int not null,
    id             int auto_increment
        primary key,
    delivery_date  date not null,
    supplier_id    int not null,
    constraint supplies_drugs_id_fk
        foreign key (drug_id) references drugs (id),
    constraint supplies_suppliers_id_fk
        foreign key (supplier_id) references suppliers (id),
    check ('cost' > 0),
    check ('drug_amount' > 0)
);

create table if not exists technologies
(
    id            int auto_increment
        primary key,
    drug_id       int not null,
    cooking_time  int not null,
    amount        int not null,
    description   text not null,
    constraint technologies_drugs_id_fk
        foreign key (drug_id) references drugs (id),
    check ('amount' > 0)
);

create table if not exists production
(
    order_id       int not null,
    technology_id  int not null,
    amount         int not null,
    constraint production_orders_id_fk
        foreign key (order_id) references orders (id),
    constraint production_technologies_id_fk
        foreign key (technology_id) references technologies (id),
    check ('amount' > 0)
);

create table if not exists technology_components
(
    technology_id  int not null,
    component_id   int not null,
    component_amount int not null,
    primary key (component_id, technology_id),
    constraint technology_components_drugs_id_fk
        foreign key (component_id) references drugs (id),
    constraint technology_components_technologies_id_fk
        foreign key (technology_id) references technologies (id),
    constraint positive_component_amount_check
        check ('component_amount' > 0)
);

create table if not exists tinctures
(
    drug_id int not null
        primary key,
    material text not null,
    constraint tinctures_drugs_id_fk
        foreign key (drug_id) references drugs (id)
);

```

4 Ограничения по поддержанию целостности

Клиент будет взаимодействовать с базой данной через REST-API с сервером, поэтому все ограничения целостности должны быть учтены лишь на сервере. То есть если клиент пришлёт некорректный запрос к серверу, то сервер провалидирует запрос, установит, что он не корректен, но никаких запросов к базе данных не будет сделано.