Міністерство освіти і науки України

Харківський національний університет радіоелектроніки

Факультет Комп’ютерні науки

Кафедра Системотехніки

**МІЖДИСЦИПЛІНАРНИЙ КУРСОВИЙ ПРОЕКТ**

**ПОЯСНЮВАЛЬНА** **ЗАПИСКА**

тема: Інформаційна система «Аптека»

(тема роботи)

з дисципліни Проектування високонавантажених систем зберігання даних

(назва дисципліни)

Керівник

(підпис, дата, посада, прізвище, ініціали)

Студент ІТКНу 16-1 31.05.2018 Байкалов М.О.

(група, підпис, дата, прізвище, ініціали)

Робота захищена з оцінкою «\_\_\_\_\_\_\_\_\_»

«\_\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2018 р.

Комісія:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(підпис, посада, прізвище, ініціали)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(підпис, посада, прізвище, ініціали)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(підпис, посада, прізвище, ініціали)

Харків 20 18

Харьковский национальный университет радиоэлектроники

**Факультет** Компьютерные науки\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Кафедра** Системотехники\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Направления подготовки** 6.050101 – Компьютерные науки\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Курс** 2у **группа** ИТКНу-16-1 **семестр** 4

**ЗАДАНИЕ**

**на курсовое проектирование**

**студенту** Байкалову Максиму Александровичу

(фамилия, имя, отчество)

**1. Тема работы:** Информационная система «Аптека»

**2. Срок сдачи студентом законченной работы** 31.05.18

**3. Исходные данные к проекту:** Разработать серверную часть информационной системы «Аптека». Серверная часть должна представлять собой 2 варианта реализации базы данных, разработанной для платформы СУБД MySQL c использованием таблиц типа MyISAM и InnoDB. Бизнес-функции системы для незарегистрированных пользователей: просмотр имеющихся товаров: поиск лекарств по названию, применению и т.д. (полнотекстовый поиск); просмотр информации по выбранному препарату; регистрация на сайте. Бизнес-функции системы для зарегистрированных пользователей: добавление лекарств в корзину; оформление заказа; вход в систему с определением статуса «user»; просмотр оформленных заказов и их состояний. Бизнес-функции системы для администраторов: вход в систему с определением статуса «admin»; добавление информации о препаратах; просмотр заказов с фильтрацией по статусу («подтвержден», «не подтвержден», «в исполнении», «доставка», «выполнен» и т.д.); изменение статуса выбранного заказа; Операционная система – Windows ХР или выше, программное обеспечение: утилита командной строки MySQL Command Line Client; программный пакет Workbench.

**4. Содержание пояснительной записки (перечень вопросов, которые подлежат разработке):** обеспечить целостность данных, создав необходимые триггеры для таблиц типа MyISAM и исполнив соответствующие инструкции SQL для таблиц типа InnoDB; описать функции интерфейса клиентской части высоконагруженной информационной системы «Аптека», реализующие основные бизнес-процессы; разработать SQL-запросы в виде процедур, функций, триггеров, представлений, необходимые для реализации бизнес-процессов на стороне сервера MySQL (включая полнотекстовый поиск); разработать транзакцию для реализации одного из основных бизнес-процессов на стороне сервера MySQL (для таблиц типа InnoDB); провести исследования и принять обоснованные решения по оптимизации доступа к высоконагруженным базам данных с помощью индексов (включая полнотекстовый поиск) и учета специфики их использования для таблиц типа MyISAM и InnoDB; провести масштабирование высоконагруженных баз данных (с использованием таблиц типа MyISAM и InnoDB) с изменением структуры данных для горизонтального и вертикального шардинга и обоснованием принятых решений. Разработать модификации процедур, функций, триггеров, транзакций для каждого варианта масштабирования; провести сравнительный анализ двух вариантов реализации базы данных (с использованием таблиц типа MyISAM и InnoDB) высоконагруженной информационной системы «Аптека» с принятием решений и разработкой рекомендаций по их использованию.

**5. Перечень графического материала (с точным определением обязательных чертежей):**

физическая модель базы данных на платформе сервера MySQL в виде ER-диаграммы согласно нотации IDEF1X (либо в виде EER-диаграммы, созданной с помощью программного пакета WorkBench) с обязательным указанием первичных и внешних ключей, типа данных, атрибутов «NULL», «NOT NULL»; таблицы запросов на выборку для обоснования и проверки разработанных процедур, функций, триггеров, представлений, транзакций; таблицы планов EXPLANE выполнения SQL-запросов для индексов, уникальных, кластерных и составных индексов; таблицы выборки данных для определения селективности составных индексов; таблицы выборки данных для обоснования выбора длины префикса при полнотекстовом поиске; измененные структуры данных физической модели базы данных при проведении масштабирования (горизонтального и вертикального шардинга).

**6. Дата выдачи задания:** 19.05.18

**Руководитель работы**

(подпись) (фамилия, имя, отчество)

**Студент Байкалов Максим Александрович**

(подпись) (фамилия, имя, отчество)

**КАЛЕНДАРНИЙ ПЛАН**

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **Название этапов курсового проекта** | **Срок выполнения** | **Примечание** |
|  | Анализ предметной области |  |  |
|  | Определение основных бизнес-функций высоконагруженной информационной системы |  |  |
|  | Определение функций интерфейса клиентской части информационной системы |  |  |
|  | Разработка серверной части информационной системы |  |  |
|  | Логическое и физическое моделирование данных. |  |  |
|  | Создание и заполнение высоконагруженных баз данных с таблицами типа MyISAM и InnoDB |  |  |
|  | Разработка поддержки целостности данных |  |  |
|  | Реализация бизнес-функций информационной системы на стороне сервера MySQL (процедур, функций, триггеров, представлений, транзакций) |  |  |
|  | Оптимизация запросов к высоконагруженным базам данных |  |  |
|  | Масштабирование баз данных |  |  |
|  | Сравнительный анализ двух вариантов реализации базы данных (с использованием таблиц типа MyISAM и InnoDB) |  |  |
|  | Рекомендации по использованию баз данных |  |  |

**Руководитель работы**

(подпись) (фамилия, имя, отчество)

**Студент Байкалов Максим Александрович**

(подпись) (фамилия, имя, отчество)

**«\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_** 20\_18\_ г.

РЕФЕРАТ

Пояснительная записка к курсовому проекту: 27 с., 26 рис., 4 приложения, 4 источника информации.

БАЗА ДАННЫХ, СИСТЕМА УПРАВЛЕНИЯ БАЗАМИ ДАННЫХ, MYSQL, МАСТАБИРОВАНИЕ, ШАРДИНГ, ОПТИМИЗАЦИЯ SQL ЗАПРОСОВ, ИНДЕКСЫ, ИНТЕРФЕЙС ДОСТУПА, СИСТЕМА УЧЕТА, ЭЛЕКТРОННАЯ КОММЕРЦИЯ

Объектом исследований курсового проекта является процесс электронной коммерции аптеки, оформления, учета и исполнения заказов, документирования всех учетных данных.

Предметом исследований курсового проекта являются информационные технологии и программные методы создания серверной части высоконагруженной информационной системы автоматизации учета продаж аптеки.

Цель исследований: разработка серверной части высоконагруженной информационной системы «Аптека».

Методы исследований – системный подход, методы структурного анализа и моделирования реляционных баз данных, методы реляционной алгебры и реляционного исчисления.

В работе проведено проектирование двух вариантов серверной части высоконагруженной информационной системы «Аптека». Проведена оптимизация SQL-запросов по критерию минимизации времени доступа к данных с учетом специфики высоконагруженных систем. В соответствии с проведенным анализом проведено обоснованное масштабирование структур данных, проведен сравнительный анализ и разработаны рекомендации по использованию каждого варианта серверной части высоконагруженной информационной системы.

Область применения – поддержка электронной коммерции аптеки.

Содержание

[ВВЕДЕНИЕ 6](#_Toc515528248)

[1 АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ 7](#_Toc515528249)

[1.1 Анализ предметной области и краткая постановка задачи 7](#_Toc515528250)

[1.2 Определение основных бизнес-функций высоконагруженной информационной системы 7](#_Toc515528251)

[1.3 Определение функций интерфейса клиентской части информационной системы 7](#_Toc515528252)

[2 РАЗРАБОТКА СЕРВЕРНОЙ ЧАСТИ ВЫСОКОНАГРУЖЕННОЙ ИНФОРМАЦИОННОЙ СИСТЕМЫ 9](#_Toc515528253)

[2.1 Логическое и физическое моделирование данных 9](#_Toc515528254)

[2.2 Разработка поддержки целостности данных 10](#_Toc515528255)

[2.3 Реализация бизнес-функций информационной системы на стороне сервера MYSQL 11](#_Toc515528256)

[3 ОПТИМИЗАЦИЯ ВЫСОКОНАГРУЖЕННЫХ БАЗ ДАННЫХ 12](#_Toc515528257)

[ВЫВОДЫ 14](#_Toc515528258)

[ПЕРЕЧЕНЬ ССЫЛОК 15](#_Toc515528259)

[ПРИЛОЖЕНИЕ А Правила целостности для таблиц типа InnoDB 16](#_Toc515528260)

[ПРИЛОЖЕНИЕ Б Тригеры для таблиц типа MyISAM 19](#_Toc515528261)

[ПРИЛОЖЕНИЕ В Процедуры и функции для реализации бизнес-функций БД 23](#_Toc515528262)

[ПРИЛОЖЕНИЕ Г Индексирование полей таблиц 25](#_Toc515528263)

# ВВЕДЕНИЕ

В настоящее время все большее внимание уделяется техническим задачам поддержки работы высоконагруженных систем хранения данных. Причиной этому является развитие технологий и появление существенного количества пользователей сети Интернет, как следствие – появление сервисов, которые направлены на удовлетворение запросов этих пользователей, а учитывая безусловный рост кол-ва новых пользователей и потребностей – речь идёт о необходимости создания систем, которые будут выдерживать такие нагрузки. В связи с этим задачи масштабирования серверных решений по нагрузке и оптимизации доступа к информации, хранящейся в базах данных, является актуальной.

Курсовое проектирование направлено на углубление и обобщение теоретических знаний по дисциплине «Проектирование высоконагруженных систем хранения данных», и на создание серверной части высоконагруженной информационной систем, используя при этом ранее изученный материал.

Работа над проектом поможет углубить знания, полученные на лекциях и практических занятиях, поможет справиться с дальнейшими дисциплинами программы обучения, включая аттестационную работу бакалавра, а также будет полезным багажом знаний для дальнейшей профессиональной деятельности.

1. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ
   1. Анализ предметной области и краткая постановка задачи

Аптека – это место, куда редко обращаются по радостным причинам (хоть бывают и такие). Создавая систему для подобного рода деятельности, необходимо особенно тщательно отнестись к вопросу быстрого отклика на запросы посетителей, а также обеспечить максимально быструю работу базы данных на стороне сервера – в общем сделать всё, чтобы процесс покупки препаратов был максимально быстр и комфортен.

* 1. Определение основных бизнес-функций высоконагруженной информационной системы

Задача создания данной системы заключается в реализации процесса покупки товара через сеть Интернет. Включающимся в эту задачу функционалом является также просмотр каталога препаратов, поиск по названию, возможность зарегистрироваться для оформления и отслеживания уже оформленных заказов.

Помимо функционала для потенциальных покупателей, необходимо реализовать возможность добавления новых препаратов в систему, возможность просмотра заказов с указанием фильтра состояния («не подтвержден», «подтверждён», «оплачен», «доставка», «получен»), а также функцию изменения статуса конкретного заказа.

* 1. Определение функций интерфейса клиентской части информационной системы

Форма поиска – текстовое поле, в которое пользователь будет вводить наименования нужного препарата.

Форма регистрации – текстовое поле для ввода логина, два текстовых поля для ввода пароля (ввод и повторный ввод для подтверждения), кнопка «Регистрация».

Кнопка «Добавить в корзину», «Корзина», «Оформить заказ», «Вход» и «Выход» из учетной записи, для зарегистрированных пользователей – кнопки «Архив заказов» и фильтр по статусу заказа.

Форма оформления заказа – Текстовые поля для заполнения информации о пользователе, кнопка «Оформить».

1. РАЗРАБОТКА СЕРВЕРНОЙ ЧАСТИ ВЫСОКОНАГРУЖЕННОЙ ИНФОРМАЦИОННОЙ СИСТЕМЫ
2. Логическое и физическое моделирование данных

Основой для информационной системы является проектирование БД. Именно БД позволяет эксплуатировать ИС, выполнять её текущее обслуживание, модифицировать и обрабатывать информацию о работе предприятия, принимать информационные потоки.

Для предметной области «Аптека» была разработана модель базы данных в виде ER-диаграммы (Рисунок 2.1), в которой присутствуют следующие сущности:

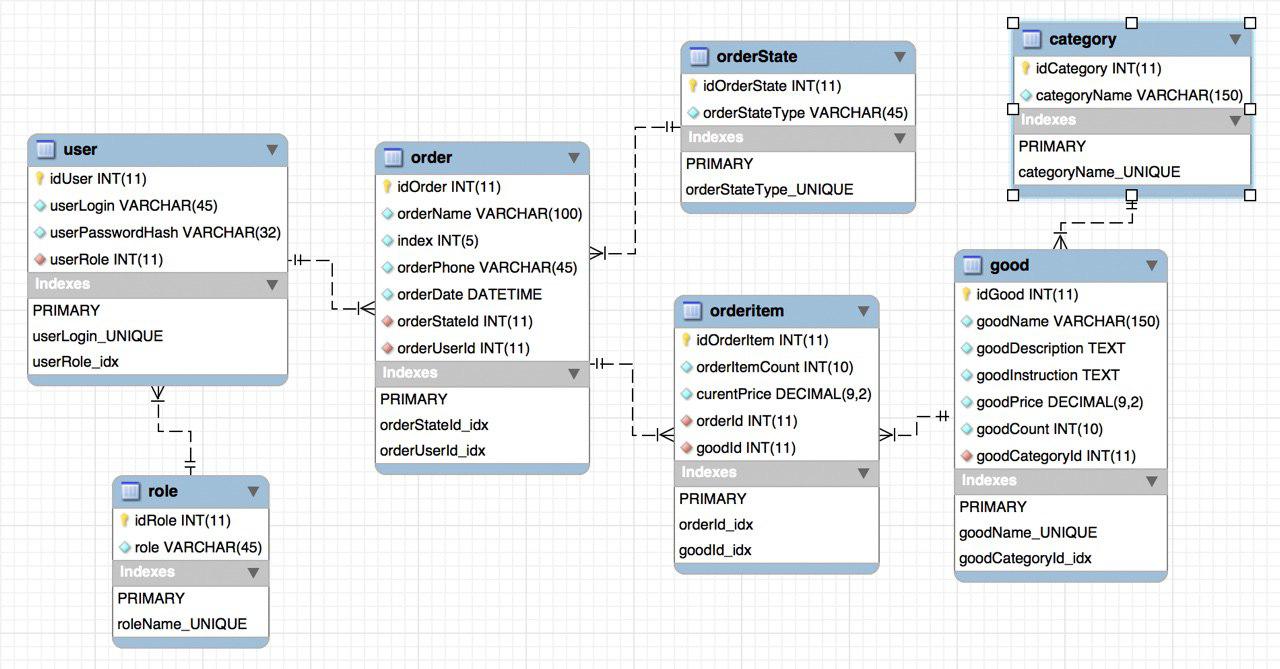


Рисунок 2.1 – ER-диаграмма для таблиц типа InnoDB

а) «user» - сущность, которая хранит в себе логин и пароль пользователя, а также id роли пользователя в сети.

б) «role» - содержит описание категорий пользователей – обычного пользователя и администратора, ключевое поле idRole и само описание – Role;

в) «orderState» - содержит перечень состояний заказа, которые могу быть («не подтвержден», «подтверждён», «оплачен», «доставка», «получен»), поля – idOrderState(ключевое) и orderStateType.

г) «category» - содержит перечень возможных категорий препаратов, поля – idCategory(ключевое поле) и categoryName.

д) «order» - информация о заказе – имя покупателя orderName, почтовый индекс Index, номер телефона orderPhone, состояние заказа OrderStateId, тип пользователя orderUserId, дата оформления заказа orderDate.

Также была разработана база данных с типом таблиц MyISAM (Рисунок 2.2).

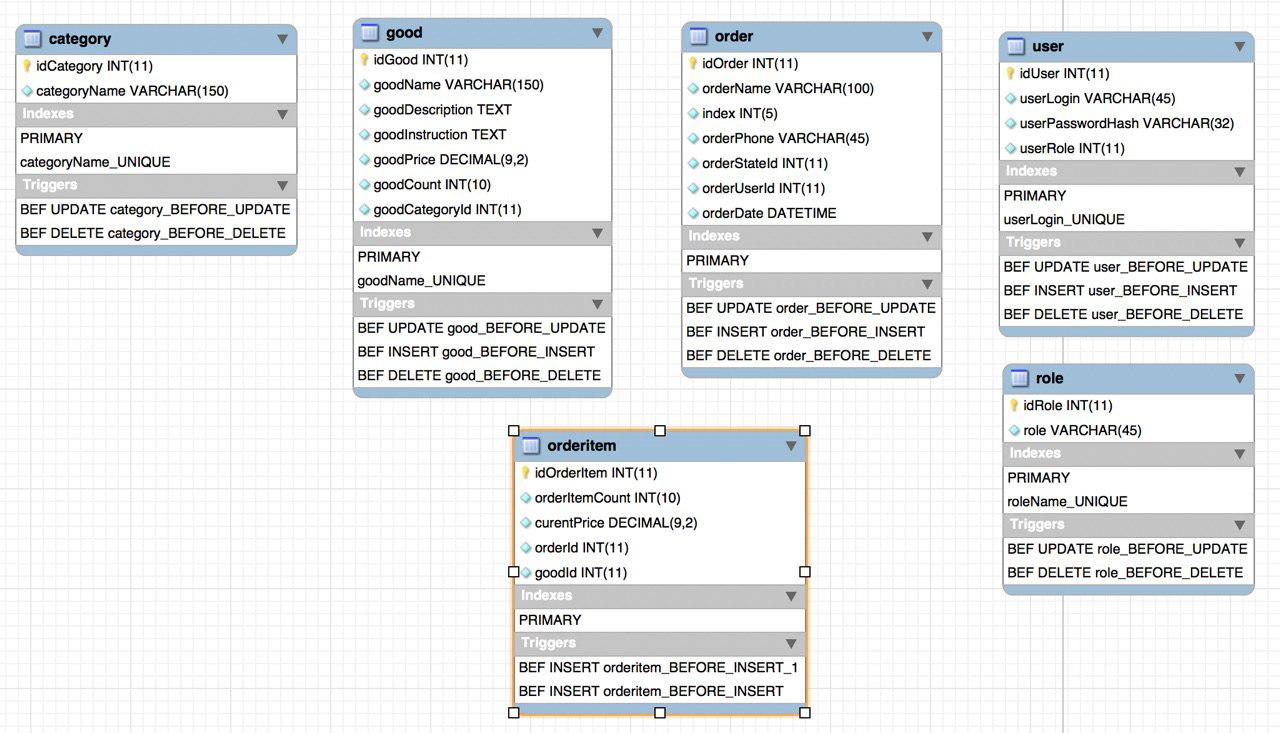


Рисунок 2.2 – ER диаграмма для таблиц типа MyISAM

1. Разработка поддержки целостности данных

Логическая модель данных была преобразована в физическую путем выбора базы данных и задания правил целостности (Приложение А). Они необходимы для описания действий сущностей относительно друг друга, то есть, какие изменения происходят в сущности, если связанная с ней другая сущность была изменена или удалена.

После построения схемы физической модели необходимо задать правила целостности данных, то есть характеристики для каждого атрибута каждой сущности, чтобы обеспечить соответствие информации её внутренней логике, структуре и всем заданным правилам. Целостность БД не гарантирует достоверности содержащейся в ней информации, но обеспечивает, по крайней мере, правдоподобность этой информации, отвергая заведомо невероятные, невозможные значения.

Так как таблицы типа “MyISAM” не поддерживают ограничения внешних ключей, целостность данных обеспечивают соответствующие триггеры (Приложение Б).

1. Реализация бизнес-функций информационной системы на стороне сервера MYSQL

Для реализации бизнес-функций, описанных в п. 1.3 были реализованы соответствующие хранимые процедуры и хранимые функции, представленные в Приложении В.

1. ОПТИМИЗАЦИЯ ВЫСОКОНАГРУЖЕННЫХ БАЗ ДАННЫХ

Для того, чтобы база данных работала быстрее, необходимо провести оптимизацию. Существует несколько способов оптимизировать базу данных. Один из способов – это уменьшение количества подстановочных таблиц, что упрощает алгоритмы запросов и процедур и ускоряет их работу. Измененная схема БД представлена ниже:

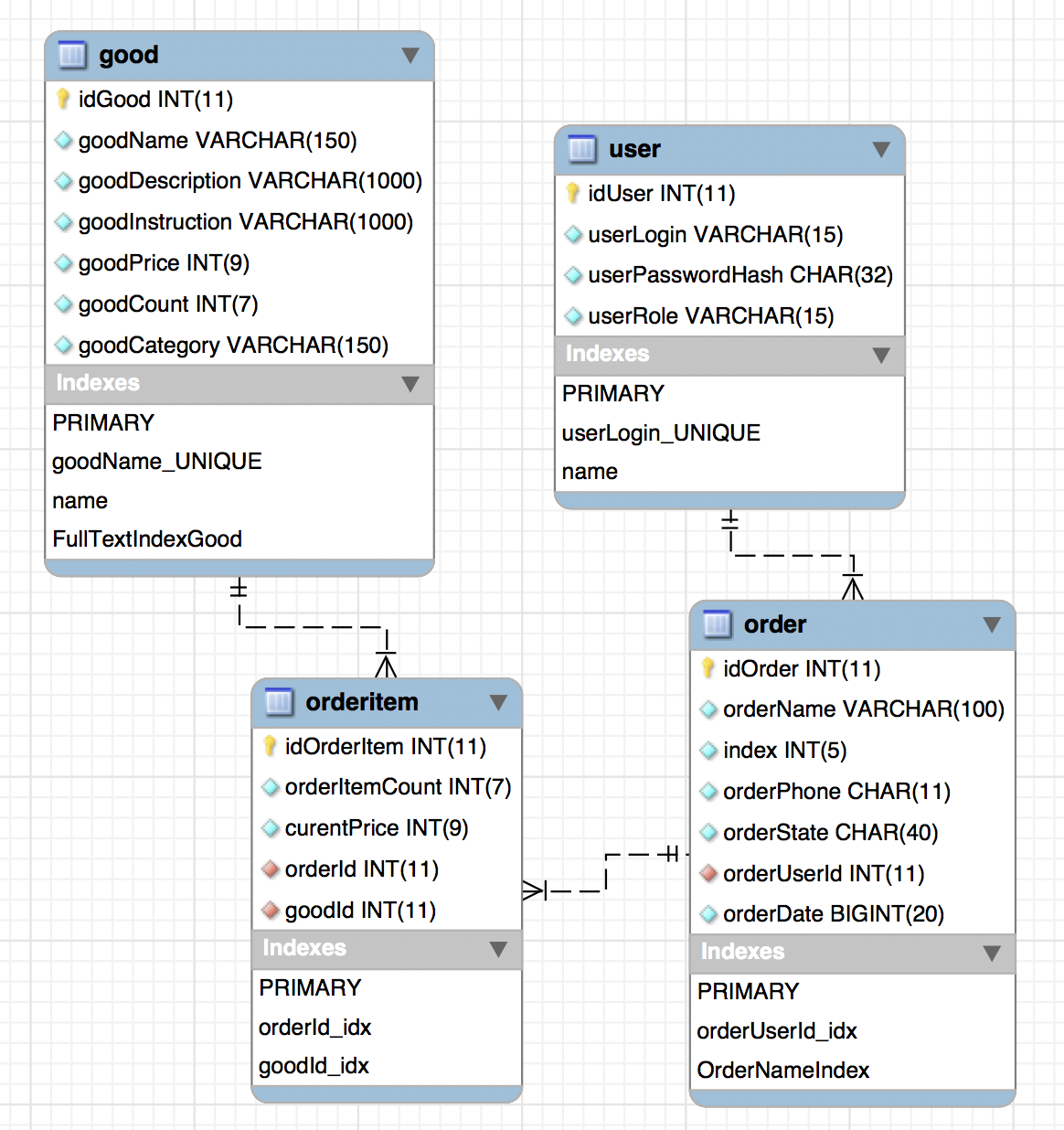


Рисунок 3.1 – Оптимизированная ER-диаграмма для таблиц типа «InnoDB»

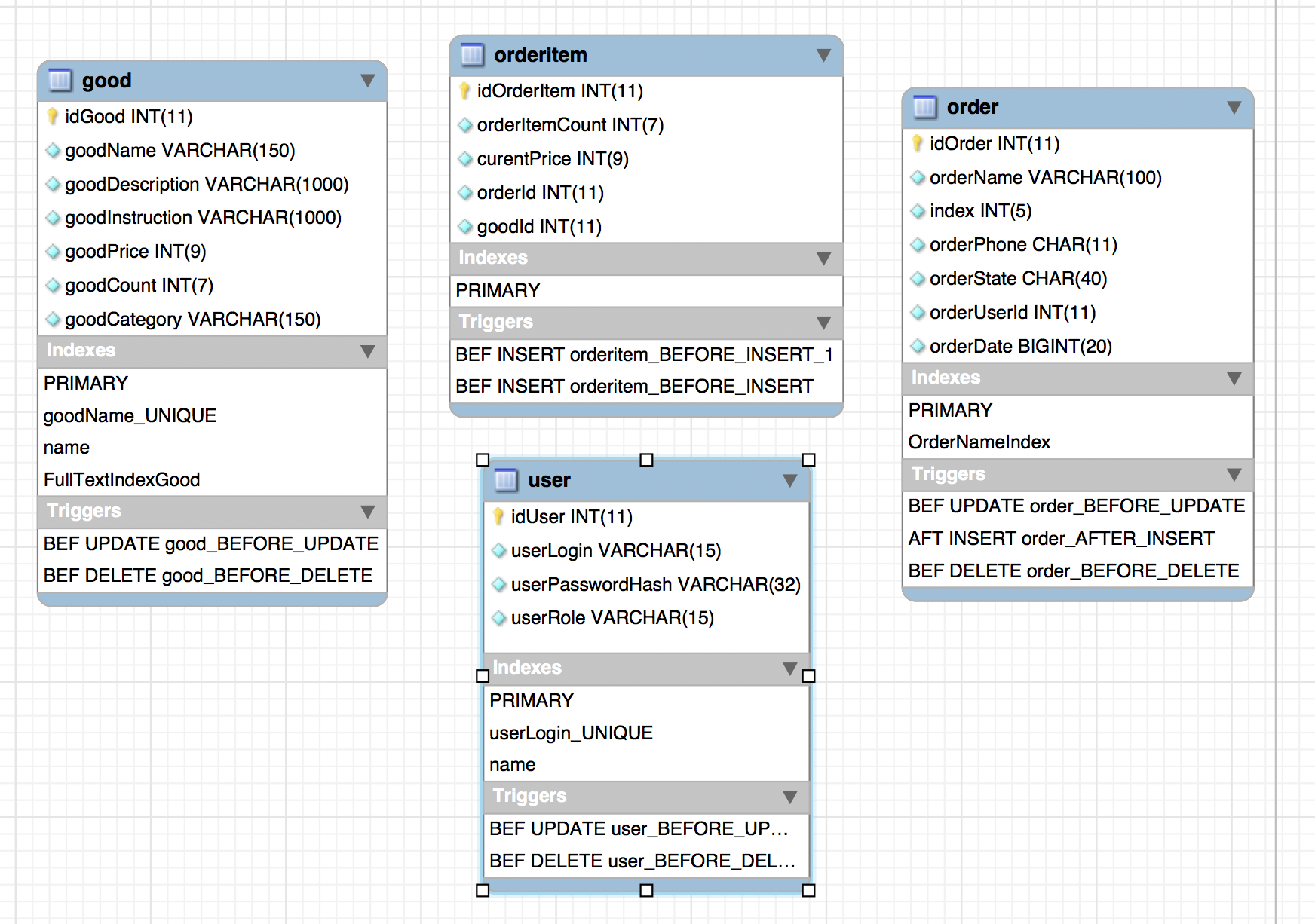


Рисунок 3.2 - Оптимизированная ER-диаграмма для таблиц типа «InnoDB»

Ещё один из способов оптимизации, который помогает ускорить работу базы данных – это индексация полей. В данном случае были использованы разные виды индексирования, в том числе и полнотекстовый поиск для поля инструкции к препарату (good.goodDescription) в хранимой процедуре find. Реализация индексирования представлена в Приложении Г.

ВЫВОДЫ

В ходе выполнения работы было выполнено проектирование серверной части высоконагруженной базы данных «Аптека». Для создания БД было использована среда визуального проектирование баз данных «MySQL Workbench» - инструмент, который позволяет наглядно представить модель базы данных в графическом виде, а также является удобным редактором SQL запросов, позволяющий сразу же отправлять их к серверу и получать ответ в виде таблицы.

Была разработана база данных для двух типов таблиц − «MyISAM» и «InnoDB». Для осуществления целостности БД в «MyISAM» были реализованы триггеры, которые устранят возможные искажения данных, в «InnoDB» были описаны ограничения в виде связей (внешних ключей). Для всех типов таблиц были использованы ограничения NOT NULL (исходя из требований системы).

Для устранения из базы избыточных функциональных зависимостей между атрибутами (полями таблиц) была проведена оптимизация путем сокращения количества таблиц и проведения индексирования отдельных полей.

В ходе выполнения курсового проекта были углублены знания и закреплены практические навыки работы с программой Workbench, а также обретены практические навыки работы с высоконагруженными базами данных.

Полученные знания и навыки будут использованы при выполнении дальнейших работ в программе университета, а также в дальнейшей профессиональной деятельности.

ПЕРЕЧЕНЬ ССЫЛОК

1. Методические указания к междисциплинарного курсового проекта по для студентов направления подготовки 6.050101 - Компьютерные науки [Электронное издание] / Сост .: Ю.В. Мищеряков, З.А. Имангулова., Л.В. Колесник - Харьков: ХНУРЭ, 2016. - 30 с.
2. Справочник по MySQL [Электронный ресурс] - Режим доступа URL: https://dev.mysql.com
3. Справочник по Workbench [Электронный ресурс] - Режим доступа URL: https:// dev.mysql.com/doc/workbench/en/
4. Работа с базами данных: учебное пособие / сост. О. Н. Евсеева, А. Б. Шамшев. – Ульяновск: УлГТУ, 2009. – 170 с.

ПРИЛОЖЕНИЕ А  
Правила целостности для таблиц типа InnoDB

CREATE TABLE Medicine\_InnoDB2.good (

idGood INT NOT NULL AUTO\_INCREMENT,

goodName VARCHAR(150) NOT NULL,

goodDescription VARCHAR(1000) NOT NULL,

goodInstruction VARCHAR(1000) NOT NULL,

goodPrice INT(9) UNSIGNED NOT NULL,

goodCount INT(7) UNSIGNED NOT NULL,

goodCategory VARCHAR(150) NOT NULL,

PRIMARY KEY (`idGood`)) ;

CREATE TABLE Medicine\_InnoDB2.user (

idUser INT NOT NULL,

userLogin VARCHAR(15) NOT NULL,

userPasswordHash CHAR(32) NOT NULL,

userRole VARCHAR(15) NOT NULL,

PRIMARY KEY (`idUser`),

UNIQUE INDEX userLogin\_UNIQUE (`userLogin` ASC));

CREATE TABLE Medicine\_InnoDB2.orderitem (

idOrderItem INT NOT NULL AUTO\_INCREMENT,

orderItemCount INT(7) UNSIGNED NOT NULL,

curentPrice INT(9) NOT NULL,

orderId INT NOT NULL,

goodId INT NOT NULL,

PRIMARY KEY (`idOrderItem`));

CREATE TABLE Medicine\_InnoDB2.order (

idOrder INT NOT NULL,

orderName VARCHAR(100) NOT NULL,

index INT(5) NOT NULL,

orderPhone CHAR(11) NOT NULL,

orderState char(40) NOT NULL,

orderUserId INT NOT NULL,

orderDate bigINT NOT NULL,

PRIMARY KEY (`idOrder`));

ALTER TABLE Medicine\_InnoDB2.orderitem

ADD INDEX orderId\_idx (`orderId` ASC);

ALTER TABLE Medicine\_InnoDB2.orderitem

ADD CONSTRAINT orderId

FOREIGN KEY (`orderId`)

REFERENCES Medicine\_InnoDB2.order (`idOrder`)

ON DELETE RESTRICT

ON UPDATE CASCADE;

ALTER TABLE Medicine\_InnoDB2.orderitem

ADD INDEX goodId\_idx (`goodId` ASC);

ALTER TABLE Medicine\_InnoDB2.orderitem

ADD CONSTRAINT goodId

FOREIGN KEY (`goodId`)

REFERENCES Medicine\_InnoDB2.good (`idGood`)

ON DELETE RESTRICT

ON UPDATE CASCADE;

ALTER TABLE Medicine\_InnoDB2.order

ADD INDEX orderUserId\_idx (`orderUserId` ASC);

ALTER TABLE Medicine\_InnoDB2.order

ADD CONSTRAINT orderUserId

FOREIGN KEY (`orderUserId`)

REFERENCES Medicine\_InnoDB2.user (`idUser`)

ON DELETE RESTRICT

ON UPDATE CASCADE;

ALTER TABLE Medicine\_InnoDB1.user

ADD INDEX userRole\_idx (`userRole` ASC);

ALTER TABLE Medicine\_InnoDB1.user

ADD CONSTRAINT userRole

FOREIGN KEY (`userRole`)

REFERENCES Medicine\_InnoDB1.role (`idRole`)

ON DELETE RESTRICT

ON UPDATE CASCADE;

ALTER TABLE Medicine\_InnoDB1.order

ADD INDEX orderStateId\_idx (`orderStateId` ASC);

ALTER TABLE Medicine\_InnoDB1.order

ADD CONSTRAINT orderStateId

FOREIGN KEY (`orderStateId`)

REFERENCES Medicine\_InnoDB1.orderState (`idOrderState`)

ON DELETE RESTRICT

ON UPDATE CASCADE;

ALTER TABLE Medicine\_InnoDB1.orderitem

ADD INDEX orderId\_idx (`orderId` ASC);

ALTER TABLE Medicine\_InnoDB1.orderitem

ADD CONSTRAINT orderId

FOREIGN KEY (`orderId`)

REFERENCES Medicine\_InnoDB1.order (`idOrder`)

ON DELETE RESTRICT

ON UPDATE CASCADE;

ALTER TABLE Medicine\_InnoDB1.orderitem

ADD INDEX goodId\_idx (`goodId` ASC);

ALTER TABLE Medicine\_InnoDB1.orderitem

ADD CONSTRAINT goodId

FOREIGN KEY (`goodId`)

REFERENCES Medicine\_InnoDB1.good (`idGood`)

ON DELETE RESTRICT

ON UPDATE CASCADE;

ALTER TABLE Medicine\_InnoDB1.order

ADD INDEX orderUserId\_idx (`orderUserId` ASC);

ALTER TABLE Medicine\_InnoDB1.order

ADD CONSTRAINT orderUserId

FOREIGN KEY (`orderUserId`)

REFERENCES Medicine\_InnoDB1.user (`idUser`)

ON DELETE RESTRICT

ON UPDATE CASCADE;

# ПРИЛОЖЕНИЕ Б Тригеры для таблиц типа MyISAM

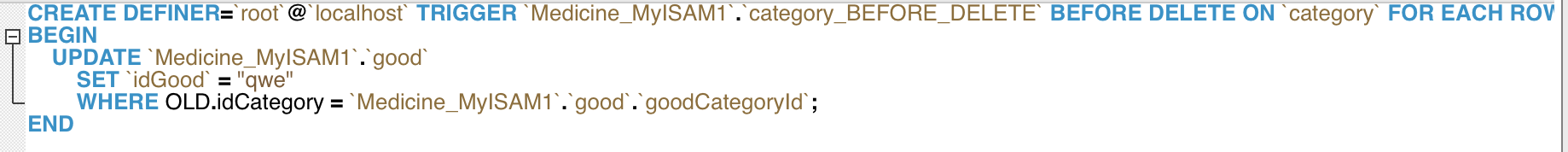


Рисунок Б.1 – Триггер, срабатывающий перед удалением из таблицы «category», отменяет удаление в случае если в таблице «good» есть хоть одна запись, которая ссылается на эту категорию

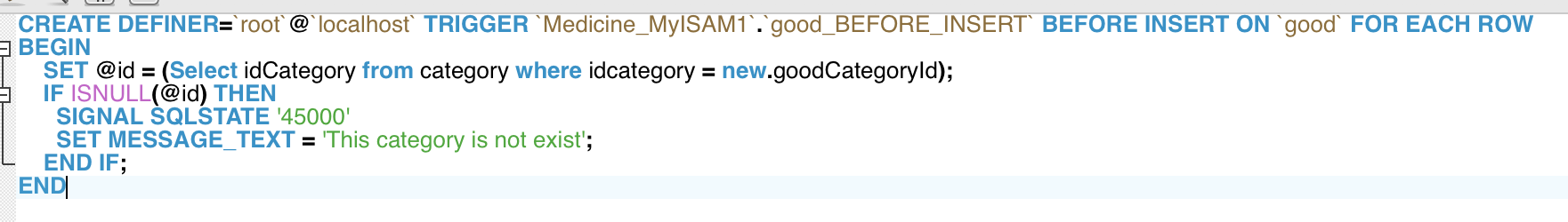
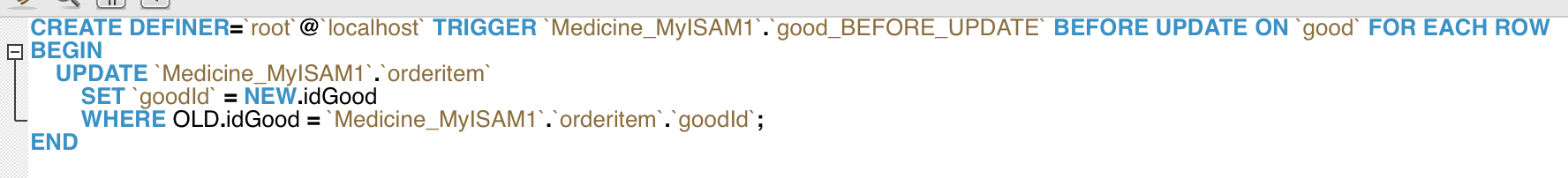


Рисунок Б.2 – Триггер, который срабатывает при вставке в таблицу «good», и выдает ошибку, если указывается категория препарата, которой не существует в таблице «category»

Рисунок Б.3 – Триггер, который срабатывает при обновлении первичного ключа таблицы «good», и изменяет внешние ключи таблицы

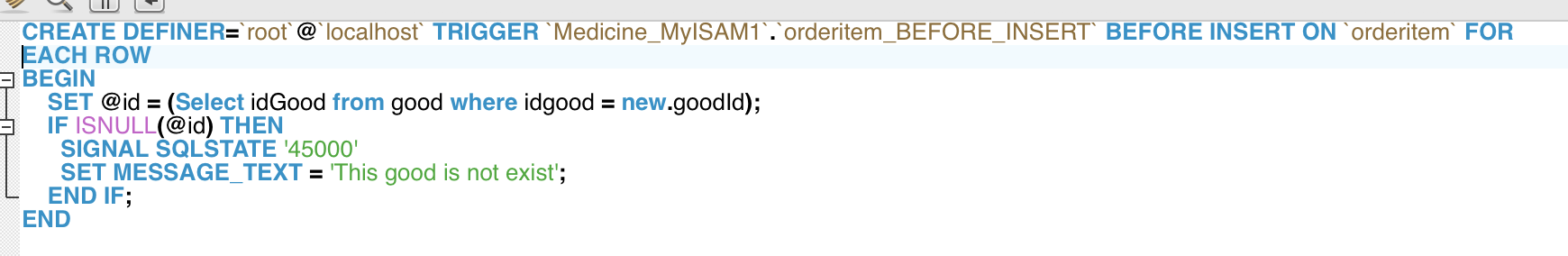


Рисунок Б.4 – Триггер, который срабатывает при добавлении записи в таблицу «orderItem», отменяя добавление в случае если препарата нет в таблице «good»

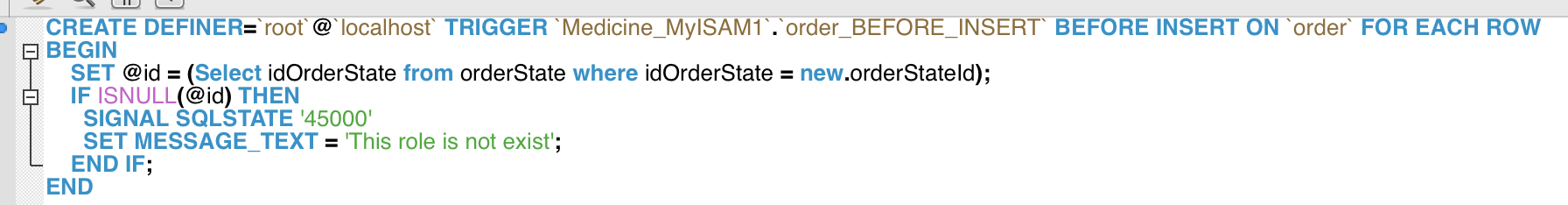


Рисунок Б.5 – Триггер, который срабатывает при добавлении записи в таблицу «order», и не допускает создание записи, если та ссылается на несуществующий IdOrderState.

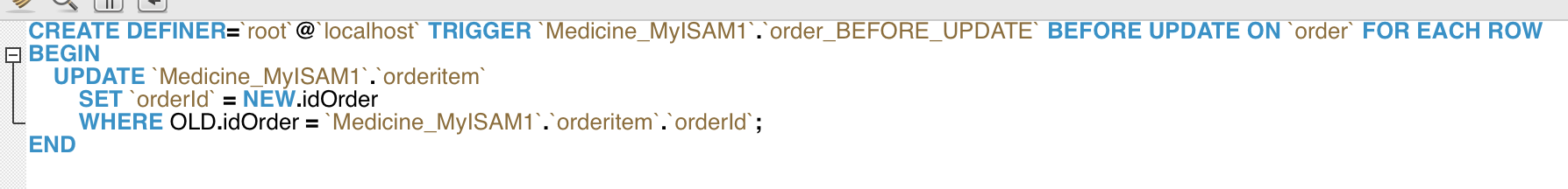


Рисунок Б.6 – Триггер, который срабатывает при обновлении таблицы «order», и изменяет значение внешнего ключа таблицы «orderItem» на новое значение первичного ключа «IdOrder»

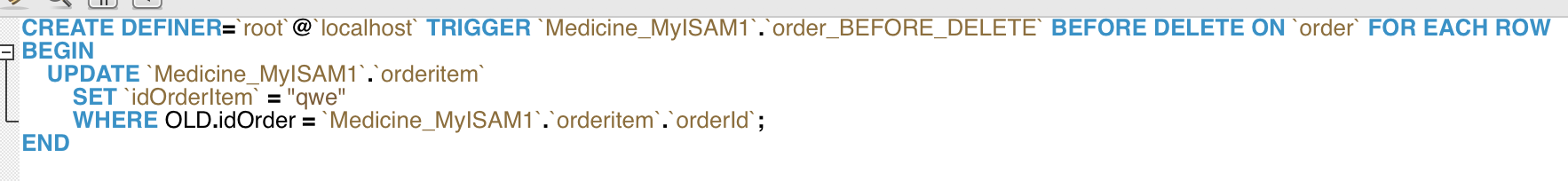


Рисунок Б.7 – Триггер, который срабатывает при удалении из таблицы «order», не позволяет удалить запись из таблицы «order», если существует привязанная к нему запись в таблице «orderItem»

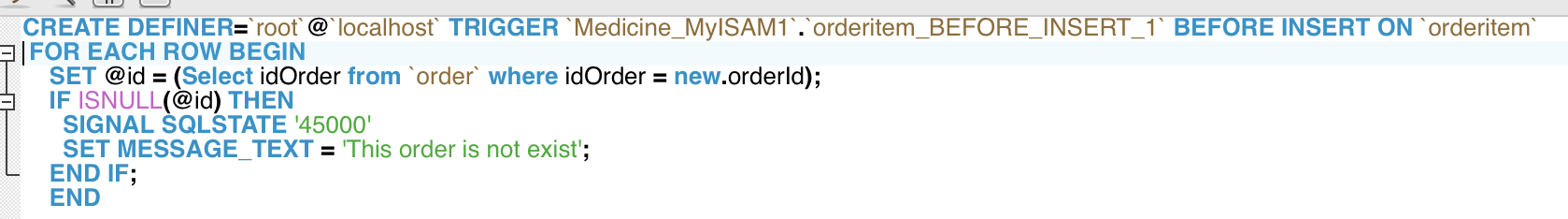


Рисунок Б.8 – Триггер, который срабатывает при добавлении записи в таблицу «orderItem», и не добавляет запись, если не существует соответствующей записи в таблице «order»

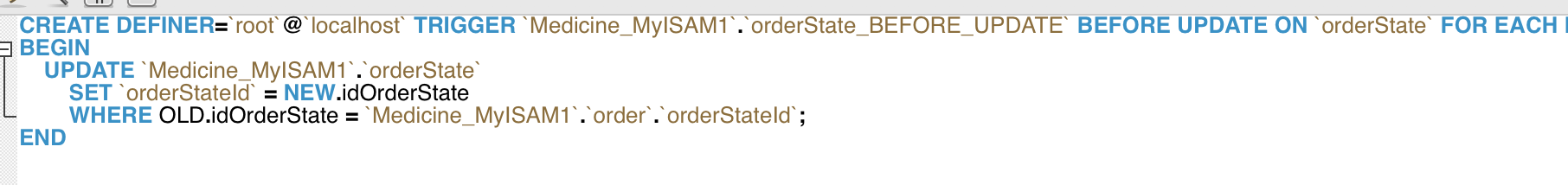


Рисунок Б.9 – Триггер, который срабатывает при обновлении таблицы «orderState», и обновляет внешний ключ в таблице «order»

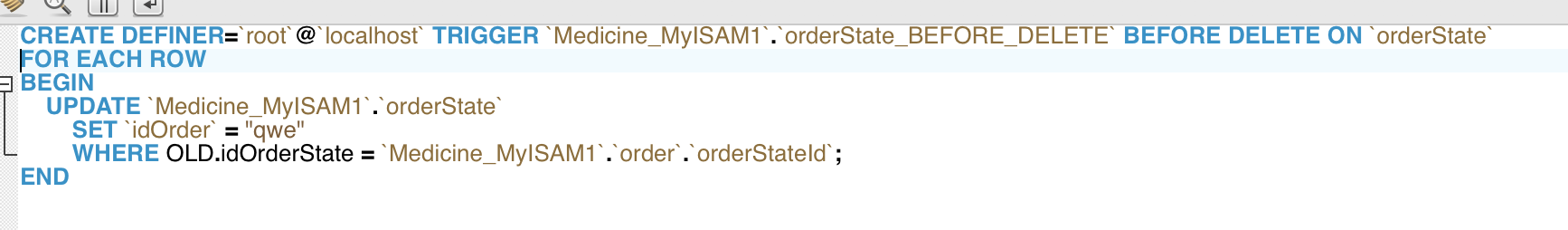


Рисунок Б.10 – Триггер, который срабатывает при удалении из таблицы «orderState», и не допускает удаления, если id статуса присутствует хотя бы в одной из записей таблицы «good»

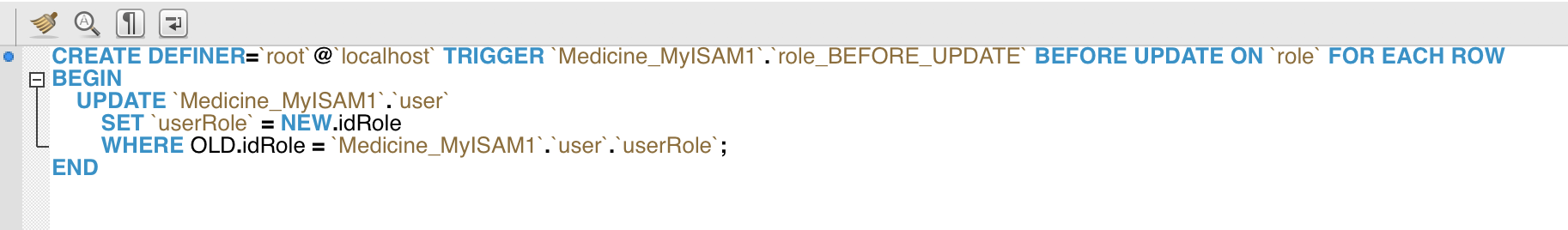


Рисунок Б.11 – Триггер, который срабатывает при обновлении таблицы «role», обновляет так же и индексы в таблице «user»

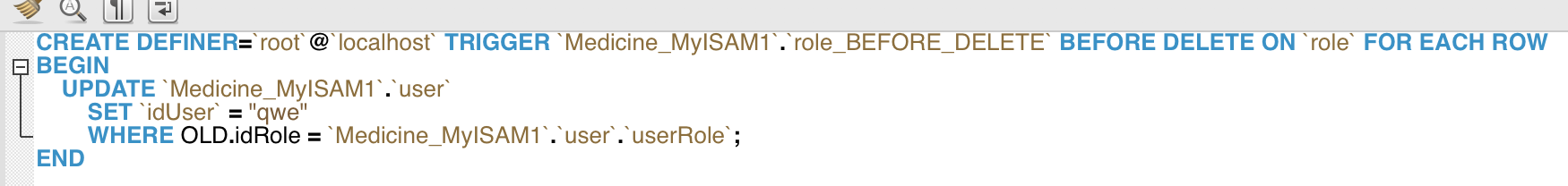


Рисунок Б.12 – Триггер, который срабатывает при удалении из таблицы «role», и не допускает удаления в случае если она присутствует в таблице «user»

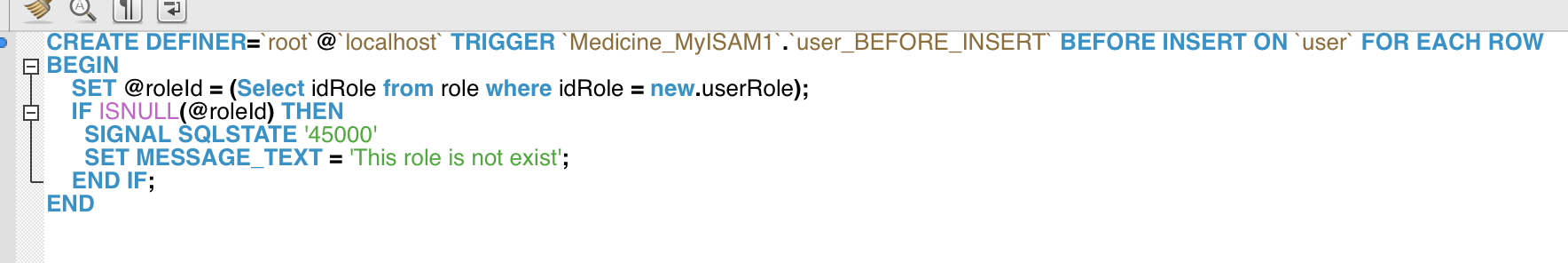


Рисунок Б.13 – Триггер, который срабатывает при добавлении в таблицу «», и не допускает добавления в случае, если указанной роли не существует

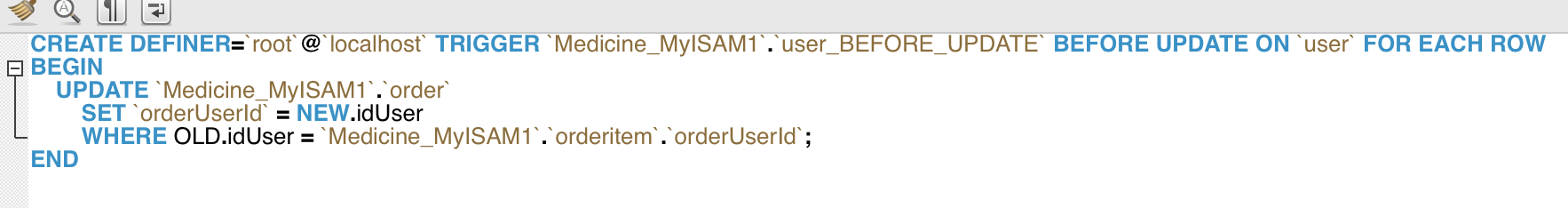


Рисунок Б.14 – Триггер, который срабатывает при обновлении таблицы «user», и обновляет внешний ключ «orderUserId»



Рисунок Б.15 – Триггер, который срабатывает при удалении записи из таблицы «user», и не допускает удаление в случае если пользователь указан хотя бы в одной из записей таблицы «order»

# ПРИЛОЖЕНИЕ В Процедуры и функции для реализации бизнес-функций БД

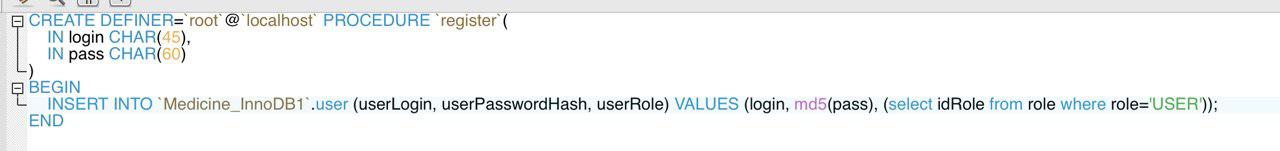


Рисунок В.1 – Процедура регистрации нового пользователя

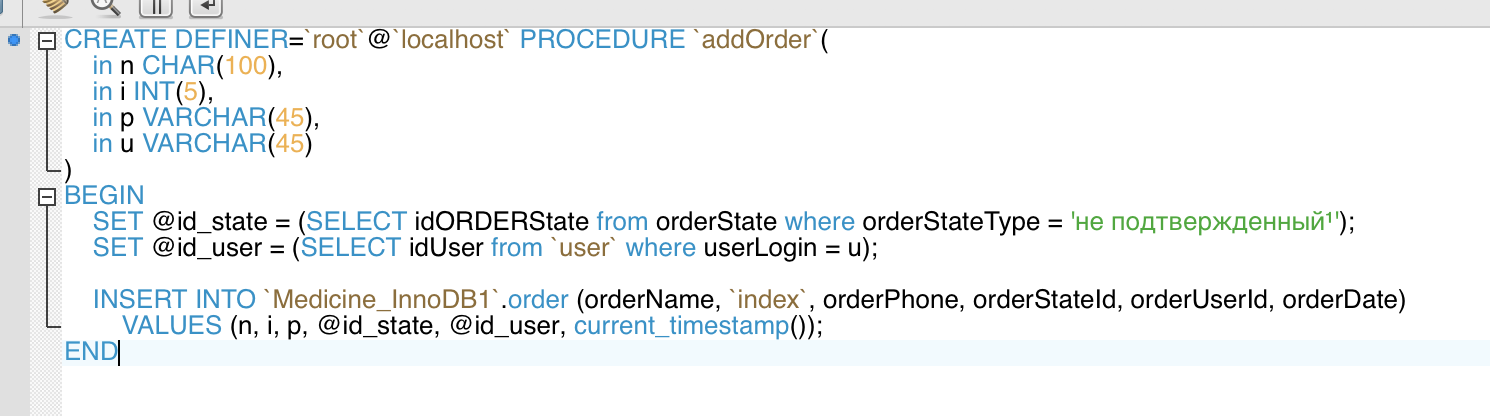


Рисунок В.2 – Процедура добавление заказа

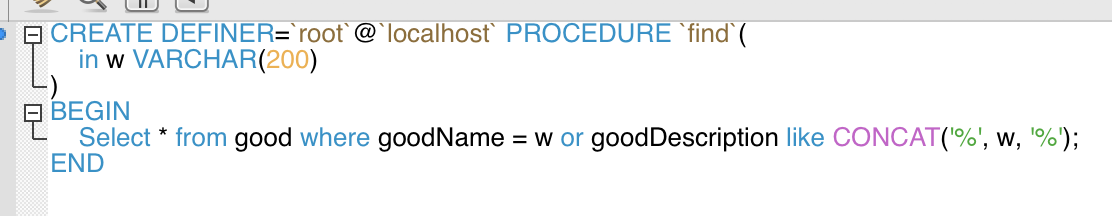


Рисунок В.3 – Процедура поиска товара по имени и описанию

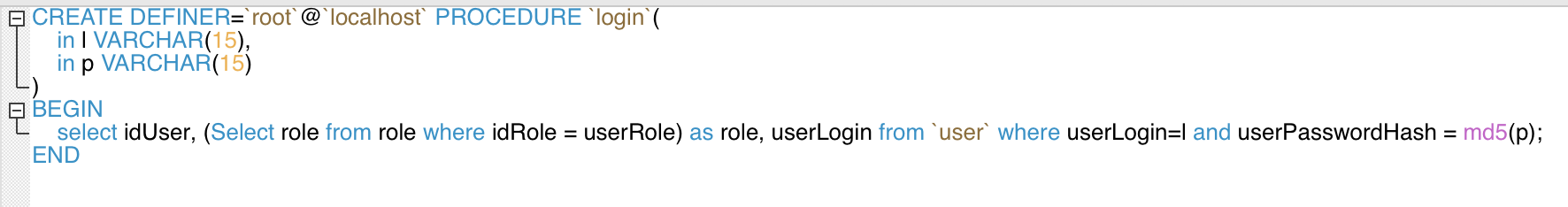


Рисунок В.4 – Процедура входа в учетную запись

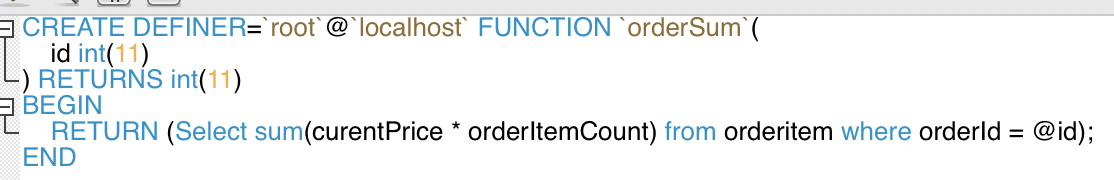


Рисунок В.5 – Функция подсчета суммы заказа

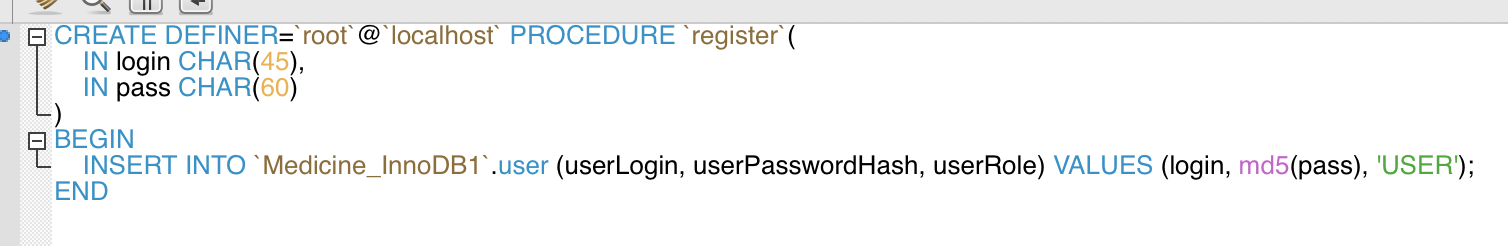


Рисунок В.6 – Оптимизированная процедура регистрации нового пользователя

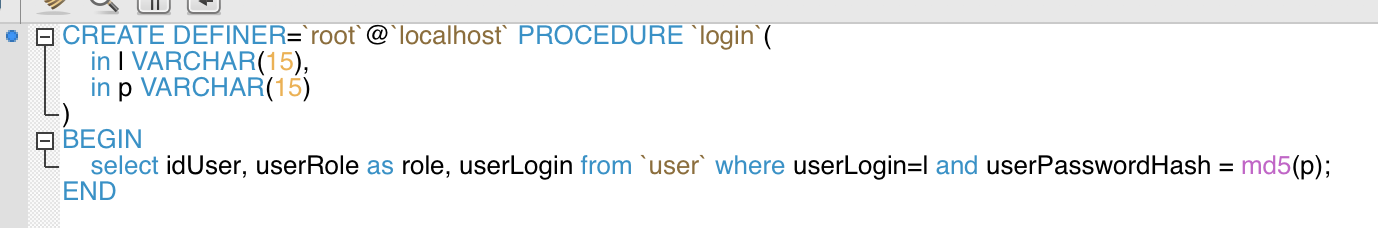


Рисунок В.7 – Оптимизированная процедура входа в учетную запись

# ПРИЛОЖЕНИЕ Г Индексирование полей таблиц

ALTER TABLE Medicine\_InnoDB2.good

ADD UNIQUE INDEX goodName\_UNIQUE (`goodName` ASC),

ADD INDEX name (`goodName` ASC);

ALTER TABLE Medicine\_MyISAM2.good

ADD INDEX name (`goodName` ASC);

ALTER TABLE Medicine\_InnoDB2.user

ADD INDEX name (`userLogin` ASC);

ALTER TABLE Medicine\_MyISAM2.user

ADD INDEX name (`userLogin` ASC);

CREATE TABLE Medicine\_InnoDB2.user (

idUser INT NOT NULL,

userLogin VARCHAR(15) NOT NULL,

userPasswordHash CHAR(32) NOT NULL,

userRole VARCHAR(15) NOT NULL,

PRIMARY KEY (`idUser`),

UNIQUE INDEX userLogin\_UNIQUE (`userLogin` ASC));

ALTER TABLE Medicine\_InnoDB2.orderitem

ADD INDEX orderId\_idx (`orderId` ASC);

ALTER TABLE Medicine\_InnoDB2.orderitem

ADD CONSTRAINT orderId

FOREIGN KEY (`orderId`)

REFERENCES Medicine\_InnoDB2.order (`idOrder`)

ON DELETE RESTRICT

ON UPDATE CASCADE;

ALTER TABLE Medicine\_InnoDB2.orderitem

ADD INDEX goodId\_idx (`goodId` ASC);

ALTER TABLE Medicine\_InnoDB2.orderitem

ADD CONSTRAINT goodId

FOREIGN KEY (`goodId`)

REFERENCES Medicine\_InnoDB2.good (`idGood`)

ON DELETE RESTRICT

ON UPDATE CASCADE;

ALTER TABLE Medicine\_InnoDB2.order

ADD INDEX orderUserId\_idx (`orderUserId` ASC);

ALTER TABLE Medicine\_InnoDB2.order

ADD CONSTRAINT orderUserId

FOREIGN KEY (`orderUserId`)

REFERENCES Medicine\_InnoDB2.user (`idUser`)

ON DELETE RESTRICT

ON UPDATE CASCADE;

CREATE FULLTEXT INDEX FullTextIndexGood ON Medicine\_InnoDB2.good(goodDescription);

CREATE INDEX OrderNameIndex ON Medicine\_InnoDB2.order(orderName);

CREATE TABLE Medicine\_MyISAM2.user (

idUser INT NOT NULL,

userLogin VARCHAR(15) NOT NULL,

userPasswordHash VARCHAR(32) NOT NULL,

userRole VARCHAR(15) NOT NULL,

PRIMARY KEY (`idUser`),

UNIQUE INDEX userLogin\_UNIQUE (`userLogin` ASC)) ENGINE MyISAM;

ALTER TABLE Medicine\_MyISAM2.user

CHANGE COLUMN idUser idUser INT(11) NOT NULL AUTO\_INCREMENT ;

ALTER TABLE Medicine\_MyISAM2.order

CHANGE COLUMN idOrder idOrder INT(11) NOT NULL AUTO\_INCREMENT ;

ALTER TABLE Medicine\_MyISAM2.good

ADD UNIQUE INDEX goodName\_UNIQUE (`goodName` ASC);

ALTER TABLE Medicine\_InnoDB1.category

ADD UNIQUE INDEX categoryName\_UNIQUE (`categoryName` ASC);