

ФЕДЕРАЛЬНОЕ АГЕНТСТВО СВЯЗИ

Ордена Трудового Красного Знамени федеральное государственное
образовательное бюджетное учреждение высшего образования «Московский
технический университет связи и информатики»

Кафедра «Интеллектуальные системы в управлении и автоматизации»

ПУГАЧЕВА МАРИЯ АЛЕКСЕЕВНА

Разработка базы данных для ИС «Книжный магазин»

Курсовая работа

Студентки 3 курса очного отделения

группа БВТ1702

Научный руководитель:

Доцент к.т.н. Воронов В.И.

Оценка _____

СОДЕРЖАНИЕ

Введение.....	3
Глава 1. Системный анализ предметной области книжного магазина.....	5
1.1 Анализ объекта автоматизации – ИС «Книжный магазин».....	5
1.2 Обзор информационных технологий по направлению разработки	7
1.2.1 Обзор распространенных СУБД	8
1.2.2. Обзор основных средств реализации пользовательских приложений	11
1.3 Обзор продуктов-аналогов	12
1.4 Требования к разрабатываемой базе данных ИС Книжного магазина	16
1.5 Выводы	17
Глава 2. Проектирование БД для ИС «Книжный магазин»	19
2.1 Разработка инфологической модели.....	19
2.2 Обоснование выбора даталогической модели данных	21
2.3 Даталогическое проектирование.....	25
2.4. Нормализация схемы базы данных «Книжный магазин».....	29
2.5 Выводы.....	30
Глава 3. Программная реализация БД «Книжный магазин»	32
3.1 Анализ и выбор СУБД.....	32
3.2 Физическое проектирование БД «Книжный магазин»	32
3.3 Разработка представлений для БД «Книжный магазин»	34
3.4. Разработка триггеров для БД «Книжный магазин».....	36
3.5 Реализация ограничений для БД «Книжный магазин»	36
3.6 Безопасность и контроль.....	38
3.7. Выводы.....	38
Заключение	40
Список литературы	41
Приложения	42
Приложение 1. Программный код реализации схемы отношений	42
Приложение 2. Программный код заполнения БД.....	46
Приложение 3. Программный код представлений	49
Приложение 4. Программный код триггеров.....	50
Приложение 5. Программный код создания пользователей и задания их привилегий.....	51

Введение

Целью курсовой работы является закрепление и развитие теоретических знаний, практических умений и навыков, полученных в ходе изучения дисциплины, связанной с созданием баз данных и информационных систем, результатом чего должна стать разработка базы данных «Книжного магазина» для автоматизации работы предприятия.

Задачи курсового проекта:

- Закрепление теоретического материала по дисциплине;
- Приобретение навыков анализа источников и литературы, используемой в процессе выполнения курсовой работы;
- Приобретение навыка системного анализа конкретной предметной области, для последующей реализации информационной системы (ИС);
- Приобретение навыка сравнительного обзора и анализа конкурентных продуктов и существующих технологий для детального решения поставленной задачи;
- Закрепление и развитие навыка составления технического задания для описания требований, предъявляемых к разработке данной базы данных;
- Приобретение и развитие навыков по проектированию и разработке ИС.

Разрабатываемая автоматизированная система управления «Книжный магазин» является актуальной в связи с постоянным спросом на физические и электронные носители текстовой информации как научной/образовательной, так и художественной литературы. Основной проблемой этого бизнеса является большой ассортимент и необходимость его систематизации. Кроме того, автоматизация должна комплексно решать вопрос увеличения клиентского потока и повышения лояльности покупателей. В настоящее время, помимо реальных магазинов, требующих фактического присутствия покупателя, стали популярны интернет-магазины, для реализации и последующей работы которых необходимы базы данных.

Описание структуры курсового проекта:

В данной работе рассматривается предметная область книжного магазина. Ее целью является разработать БД. Разработка будет проводиться

поэтапно. Проведется системный анализ предметной области, на основе чего будут сформированы требования к системе.

На основе системного анализа строится инфологическая модель, параллельно решается вопрос о наиболее оптимальном использовании программных средств для реализации БД, а также проводится анализ уже существующих решений для выявления достоинств и недостатков, которые необходимо учесть в разрабатываемой БД.

Инфологическая модель преобразуется в реляционную, после чего можно приступать к реализации базы данных, основываясь на выявленной информации. Разработка базы данных завершается программной реализацией, предполагающей дальнейшее тестирование и ввод в эксплуатацию.

Глава 1. Системный анализ предметной области книжного магазина

1.1 Анализ объекта автоматизации – ИС «Книжный магазин»

Книжный магазин является предприятием розничной торговли, реализующим издательскую книжную продукцию, а также сопутствующие товары, как правило, канцелярские. Книжные магазины в большинстве случаев представляют собой филиалы с торговыми залами для сбыта товара, и складские помещения, осуществляющие прием/хранение ассортимента.

В современных реалиях, конкурентно-экономическая ситуация призывает участников этого сегмента рынка к расширению каналов сбыта, побуждая дублировать физические филиалы в виртуальную среду, например, открывать интернет-магазины для практически бесконтактной реализации товара.

Таким образом, очевидна необходимость в создании информационной системы, для поддержания и осуществления работы предприятий данного сегмента: учет огромнейшего количество наименований, актуальность хранимой информации, удобство работы, как для сотрудников с целью изменения и/или добавления новой информации, так и для покупателя, потребности которого должны быть учтены и удовлетворены.

Со стороны внутреннего устройства организации должны быть учтены организационно-функциональная структура и полный ассортимент складов и торговых залов, с возможностью внесения изменений.

Со стороны покупателя:

- ознакомительный доступ к ассортименту;
- возможность покупки, посредством взаимодействия с ИС через сети Интернет;
- возможность создания пользовательской учетной записи для участия в программе лояльности магазина.

Исходя из вышеизложенного, ИС содержит в себе следующие блоки:

- структурная организация предприятия;
- информация об ассортименте;
- информация о покупателях.

Структурно-функциональная схема предприятия отражена на рисунке 1 и представляет из себя иерархию обязанностей.

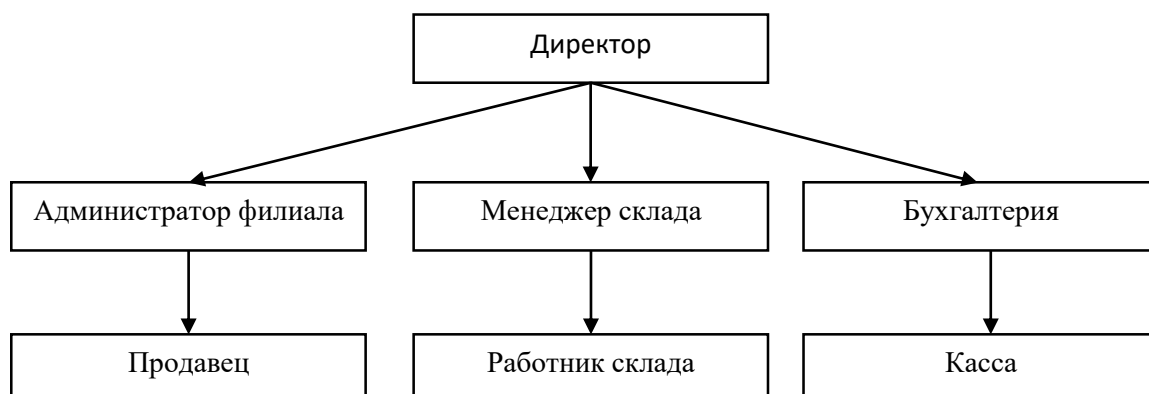


Рисунок 1 – структурно-функциональная схема предприятия

Директор магазинов осуществляет общее руководство деятельностью предприятия, производя контроль качества бизнес-процессов и организацию взаимодействия между отделами.

Бухгалтерия выполняет учет ресурсов: их распределение между складами и филиалами, планирование закупок, построение плана продаж, оформление финансово-юридической составляющей компании, в том числе кассовые операции филиалов и интернет-магазина.

Менеджеры складов осуществляют прием и передачу заказов на сборку работникам склада, а также внутреннюю обработку заявок на поставки в филиалы для поддержания ассортимента и прием поставок на склад.

Работники складов осуществляют сборку и передачу заказов и заявок, полученных от менеджера.

Администраторы филиалов выполняют анализ работы магазина: контроль работы продавца в отделе продаж, текущее состояние ассортимента, принимают поставки, создают заявки на пополнение.

Покупатель, приходя в магазин, взаимодействует с продавцом, который, в свою очередь, осуществляет продажу. Операции с денежными средствами и отбивку чека осуществляются на кассе также продавцом, в случаях физической покупки. При покупке товара, покупатель получает чек с указанием товара и его стоимости.

В магазине введется учет всех работников, товаров, их наличия на складах и в магазинах, заказов, поставок и клиентов.

Для сотрудника магазина хранится следующая информация: ФИО, Дата рождения, Должность, Телефон, Адрес работы, Оклад.

При оформлении заказа/регистрации на сайте интернет-магазина клиентом, фиксируются и хранятся: ФИО, Дата рождения, Телефон, Электронная почта, Логин, Пароль, Номер карты лояльности, Статус скидки.

При учете информации о торговом зале/складе в ИС хранятся данные о: Адресе филиала/склада, Руководитель, Сотрудники, Перечень и количество хранимых наименований.

Данные о заказе: ФИО покупателя, Телефон, Электронная почта, Наименования товара, Количество товара, Карта программы лояльности, Дата заказа, Адрес доставки, Дата доставки, Стоимость доставки, Стоимость заказа, Статус доставки.

Данные в чеке по результатам заказа/покупки в торговом зале: Код чека, Дата и время покупки, Продавец, Наименования товара, Количество, Стоимость каждой позиции, Подоходный налог, Итоговая стоимость.

Данные о книгах: ISBN, Штрих-код, Название, Автор, Категория(жанр) книги, Год, Цена, Общее количество в сети.

Данные о поставках: Название поставщика, Ответственное за прием лицо, Наименования товара, Количество, Стоимость поставки, Дата поставки.

Данные о заявках в сети: Заявитель, Магазин для поставки; Лицо, обработавшее заявку; Наименования товара, Количество, Дата заявки, Дата поставки.

Необходимо предусмотреть следующие ограничения:

- Возраст сотрудников не младше 18;
- Возраст клиентов от 14;
- Сумма количества наименований в филиале/складе не превышает общее количество книг в сети.

1.2 Обзор информационных технологий по направлению разработки

Современные информационные системы представляют собой комплекс программных средств, чей функционал условно подразделяется на два основных блока: front-end и back-end, т.е. внешний слой комплекса, посредством которого конечный пользователь взаимодействует с системой, например, интерфейс, и внутренний, осуществляющий обработку запросов и реализующий логику приложения.

Как правило, к back-end разработке относится вся аппаратно-программная часть сервиса:

- Организация и работа с базами данных посредством СУБД;

- Обеспечение корректной работы всех функций сайта/приложения;
- Разработка логики и алгоритмов работы приложения;
- Интеграции с внешними сервисами;
- Тестирование и отладка
- И др.

Ниже будут рассмотрены системы управления базами данных (СУБД), посредством которых в любой ИС осуществляется доступ к хранилищам данных, а также способ их организации.

1.2.1 Обзор распространенных СУБД

1.2.1.1. Microsoft Access

Microsoft Office Access — реляционная система управления базами данных (СУБД) корпорации Microsoft. Входит в состав пакета Microsoft Office. Имеет широкий спектр функций, включая связанные запросы, связь с внешними таблицами и базами данных. Благодаря встроенному языку VBA, в самом Access можно писать приложения, работающие с базами данных.

Основные компоненты MS Access:

- построитель таблиц;
- построитель экранных форм;
- построитель SQL-запросов (язык SQL в MS Access не соответствует стандарту ANSI);
- построитель отчетов, выводимых на печать.

Компоненты могут вызывать скрипты на языке VBA, поэтому MS Access позволяет разрабатывать приложения и БД практически «с нуля» или написать оболочку для внешней БД.

Microsoft Jet Database Engine, которая используется в качестве движка базы данных MS Access, является файл-серверной СУБД, и потому применима лишь к приложениям, работающим с небольшими объемами данных и при небольшом числе пользователей, одновременно работающих с этими данными. Непосредственно в Access отсутствует ряд механизмов, необходимых в многопользовательских базах данных, таких, например, как триггеры.

Встроенные средства взаимодействия MS Access со внешними СУБД с использованием интерфейса ODBC снимают ограничения, присущие Microsoft Jet Database Engine. Инструменты MS Access, которые позволяют реализовать такое взаимодействие, называются «связанные таблицы» (связь с

таблицей СУБД) и «запросы к серверу» (запрос на диалекте SQL, который «понимает» СУБД)[1].

Подводя итог: Microsoft Access является хорошей СУБД для настольного (домашнего) использования, так как предоставляет интуитивно понятный интерфейс для взаимодействия с базой данных. Однако едва ли данный продукт пригоден для профессионального корпоративного использования, поскольку в Access отсутствует поддержка возможности работы с базой данных множества пользователей одновременно. К тому же, затруднено взаимодействие сторонних приложений с БД ввиду того, что Access не поддерживает принятый ANSI стандарт SQL.

1.2.1.2. Oracle Database

Oracle Database или Oracle RDBMS — объектно-реляционная система управления базами данных компании Oracle.

В состав дистрибутива СУБД входят продукты, позволяющие автоматизировать рутинный процессы администрирования: это ПО «Scheduler» и подсистема задач (jobs). Также компания предоставляет продукт Oracle Net Services, позволяющий удалённым клиентам или приложениям получить доступ к БД с использованием сетевых сессий при помощи различных протоколов. Администраторы базы данных могут осуществлять контроль многих параметров экземпляра БД через конфигурационный файл.

База данных Oracle осуществляет контроль множественного одновременного доступа к данным при помощи блокировок (locks). Также в базе данных используются «завдвижки» (latches) – низкоуровневые механизмы сериализации, позволяющие защитить разделяемые структуры данных.

Некоторые ключевые возможности Oracle Database:

- Real Application Cluster (RAC) обеспечивает работу одного экземпляра базы данных на нескольких узлах grid, позволяя управлять нагрузкой и гибко масштабировать систему в случае необходимости.
- Automatic Storage Management (ASM) позволяет автоматически распределять данные между имеющимися ресурсами систем хранения данных, что повышает отказоустойчивость системы и снижает общую стоимость владения (TCO).
- Большие базы данных. Максимальный размер экземпляра базы данных Oracle может достигать 8 экзбайт.
- Самоуправление. Специальные механизмы Oracle Database позволяют самостоятельно перераспределять нагрузку на систему,

оптимизировать и корректировать SQL-запросы, выявлять и прогнозировать ошибки. [2]

Oracle Database является примером профессиональной СУБД, поддерживающей большое разнообразие платформ, средств администрирования и управления, и предоставляющей собственную среду разработки – Oracle SQL Developer – хотя и не привязанной к ней жёстко. С другой стороны, внедрение целой экосистемы технологий, разработанных компанией Oracle, может быть целесообразно только для крупных компаний; для средних и малых же издержки на обучение персонала и поддержание необходимого оборудования могут оказаться слишком высоки.

1.2.1.3. MySQL

MySQL — свободная реляционная система управления базами данных. Разработку и поддержку MySQL осуществляет корпорация Oracle, получившая права на торговую марку вместе с поглощённой Sun Microsystems, которая ранее приобрела шведскую компанию MySQL AB. Продукт распространяется как под GNU General Public License, так и под собственной коммерческой лицензией. Помимо этого, разработчики создают функциональность по заказу лицензионных пользователей.

MySQL является решением для малых и средних приложений. Обычно MySQL используется в качестве сервера, к которому обращаются локальные или удалённые клиенты, однако в дистрибутив входит библиотека внутреннего сервера, позволяющая включать MySQL в автономные программы.

Гибкость СУБД MySQL обеспечивается поддержкой большого количества типов таблиц: пользователи могут выбрать как таблицы типа MyISAM, поддерживающие полнотекстовый поиск, так и таблицы InnoDB, поддерживающие транзакции на уровне отдельных записей. Благодаря открытой архитектуре и GPL-лицензированию, в СУБД MySQL постоянно появляются новые типы таблиц.[3]

MySQL имеет API для языков Delphi, C, C++, Эйфель, Java, Lisp, Perl, PHP, Python, Ruby, Smalltalk, Компонентный Паскаль и Tcl, библиотеки для языков платформы .NET, а также обеспечивает поддержку для ODBC посредством ODBC-драйвера MyODBC.

Хотя MySQL начинался как низкобюджетная альтернатива более мощным проприетарным базам данных, в итоге он эволюционировал до поддержки больших объёмов данных и распределённой инфраструктуры. Вокруг данной СУБД сформирована целая экосистема открытых инструментов разработки и администрирования.

Подводя черту: СУБД MySQL подходит для целей данного проекта. С одной стороны, она достаточно развита для поддержки множественного доступа пользователей и работы по сети; с другой стороны, она достаточно проста и не требует внедрения в информационную систему большого числа сторонних технологий. MySQL воплощает золотую середину между функциональностью и легковесностью, и поэтому хорошо подходит для использования в ИС среднего масштаба.

1.2.2. Обзор основных средств реализации пользовательских приложений

1.2.2.1. Язык программирования C#

C# — объектно-ориентированный язык программирования. Разработан в 1998—2001 годах группой инженеров под руководством Андерса Хейлсберга в компании Microsoft как язык разработки приложений для платформы Microsoft .NET Framework и впоследствии был стандартизирован как ECMA-334 и ISO/IEC 23270.

C# относится к семье языков с C-подобным синтаксисом, из них его синтаксис наиболее близок к C++ и Java. Язык имеет статическую типизацию, поддерживает полиморфизм, перегрузку операторов (в том числе операторов явного и неявного приведения типа), делегаты, атрибуты, события, свойства, обобщённые типы и методы, итераторы, анонимные функции с поддержкой замыканий, LINQ, исключения, комментарии в формате XML.[4]

Язык C# имеет развитые инструменты как для разработки настольных приложений для ПК (среда разработки Visual Studio), так и для создания веб-интерфейсов (технология ASP.NET). Основной идеолог, разработчик и распространитель языка — компания Microsoft. Как следствие, данный язык полностью заточен на работу с продуктами Microsoft — полноценно поддерживается только на ОС семейства Windows, качественная разработка возможна только средствами Microsoft Visual Studio. Поскольку при выборе СУБД мы обошли Microsoft SQL Server стороной, то для разработки приложений нет смысла использовать стек технологий Microsoft (отдельно от которого язык C# практически невозможно использовать). Тем более, что было принято решение не идти на высокие финансовые траты, а лицензии на ПО компании Microsoft могут составить львиную долю бюджета проекта.

1.2.2.2. Язык программирования Python

Python — высокоуровневый язык программирования общего назначения, ориентированный на повышение производительности разработчика и читаемости кода. Синтаксис ядра Python минималистичен, в то

же время стандартная библиотека включает большой объём полезных функций.

Python поддерживает несколько парадигм программирования, в том числе структурное, объектно-ориентированное, функциональное, императивное и аспектно-ориентированное. Основные архитектурные черты — динамическая типизация, автоматическое управление памятью, полная интроспекция, механизм обработки исключений, поддержка многопоточных вычислений и удобные высокоуровневые структуры данных. Код в Python организовывается в функции и классы, которые могут объединяться в модули (они в свою очередь могут быть объединены в пакеты). Python — активно развивающийся язык программирования, новые версии (с добавлением/изменением языковых свойств) выходят примерно раз в два с половиной года.[5]

Одно из популярных применений языка — разработка на нём web-приложений. Существует целая экосистема библиотек, фреймворков и инструментов разработки для создания web-приложений любого уровня. Для создания простых и легковесных сайтов часто используется библиотека Bottle; одним из стандартных фреймворков для создания web-приложений среднего масштаба является Django, интегрирующий в себе веб-сервер и обёртку над СУБД. Также существует множество других библиотек и фреймворков различной тяжёловесности: Flask, Pyramid, Tornado. Среди них можно выделить Morepath — микрофреймворк, позволяющий создавать приложения, определяемые моделью данных.

Также Python имеет официальные драйверы под множество популярных СУБД, включая: MySQL, MS SQL, Postgres SQL, Oracle, SQLite, MongoDB и многие другие.

Помимо прочего, язык имеет развитые инструменты для создания настольных приложений: к примеру, популярностью в этой сфере пользуется обёртка PyQt над известной кроссплатформенной библиотекой Qt, написанной на C++. При этом процесс разработки остаётся прежним, так как большинство инструментов разработки Qt применимы и к PyQt — например, графический редактор форм Qt Designer.

1.3 Обзор продуктов-аналогов

Основной спецификой и сложностью при создании книжного магазина, а также его последующей автоматизации выступает необходимость систематизации огромнейшего ассортимента, обязательного для удовлетворения потребностей широкой целевой аудитории. В связи с этим, полных аналогов хорошо автоматизированной системы на рынке немного.

Ниже будут представлены CRM-системы, нацеленные на реализацию интернет-магазина.

1.3.1 CRM система РосБизнесСофт

Российская CRM система для малых и средних предприятий. Продукт предназначен для управления отношениями с клиентами, поставщиками, автоматизации производства, сервисного обслуживания, маркетинга и другого. Среди клиентов: OZON.RU, Karcher, Мое дело, IMMERGAS и другие.[6]

Основные возможности ПО:

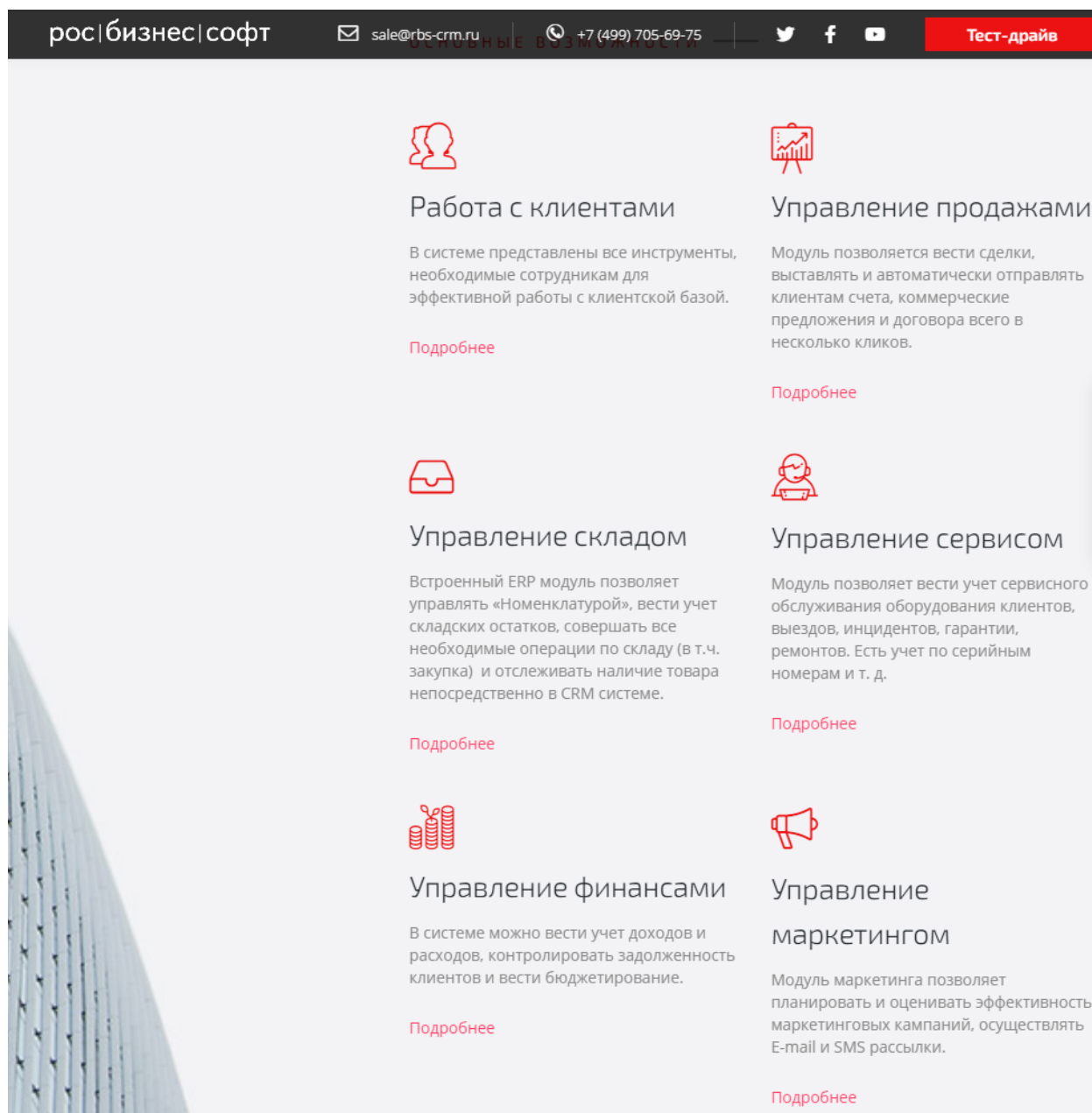


Рисунок 2 – основные возможности ПО автоматизации РосБизнесСофт

Таким образом,

представленная CRM платформа покрывает весь базовый функционал практически любого интернет-магазина, что удобно для быстрого запуска. Дополнительные возможности могут быть настроены с применением интегрированных сервисов, допустимых для платформы.

1.3.2 Система управления интернет-магазинами Magento

Magento — система управления интернет-магазинами. По данным статистического сайта alexa.com, Magento — самая популярная система управления интернет-магазинами в мире на февраль 2013 г. В июне 2011 г. компания Magento Inc. была приобретена компанией eBay Inc.

Magento — одна из самых популярных открытых систем для организации электронной коммерции в сети: на базе этой платформы создано более 100 000 интернет-магазинов, сторонними разработчиками создано более 2 тысяч расширений, сообщество проекта насчитывает около 375 000 участников.

Одна из самых сильных сторон Magento – обширный встроенный функционал, который клиент получает сразу после установки магазина прямо «из коробки». Основные поддерживаемые функции[7]:

- Множество валют
- Многоязычность
- Множество сайтов на одной системы
- Скидки/купоны
- Отчеты
- Отмеченные товары
- Ограничение доступа

К минусам системы можно отнести сложность её персонализации под нужды конкретного предприятия.

1.3.3 1С:Розница 8. Книжный магазин

Отраслевое решение "1С:Книжный магазин" предназначено для автоматизации оперативного и управленческого учета, анализа и планирования операций в розничной торговле книгами, печатной продукцией, периодическими изданиями и канцелярскими принадлежностями, как в формате одиночных магазинов, так и розничной торговой сети. Отраслевое решение создано на основе программного продукта "1С:Розница 8".

Основные функциональные возможности отраслевого решения:

- Оформление прихода товаров от контрагента на склады магазина;

- Оформление реализации товаров и услуг контрагенту, в том числе в двухфазовом (ордерном) режиме;
 - Оформление перемещения товаров между магазинами, внутренними складами магазинов, магазинами и складами предприятия, в том числе в двухфазовом (ордерном) режиме;
 - Оформление возвратов товаров от покупателей (реализованы механизмы автоматического создания необходимых документов при возврате "Не День в День" в режиме РМК);
 - Оформление документов инвентаризации товаров ("Инвентаризация товаров", "Списание товаров", "Оприходование товаров");
 - Оформление приходных и расходных кассовых ордеров непосредственно в магазинах;
 - Оформление документов перемещения денежных средств между магазинами, внутренними кассами магазинов, магазинами и кассами предприятия;
 - Оформление документов передачи товаров в службу доставки;
 - Оформление чеков продажи, и по окончании смены сводного отчета по контрольно-кассовой машине, с учетом возвращенных товаров в смену;
 - Возможность использования процентных скидок по дисконтным картам (накопительные скидки), скидки с разделением по магазинам, скидки контрагентам, скидки на сумму чека, скидки по времени действия, по количеству товара, по виду оплаты, а также скидки на морально устаревшие издания;
 - Поддержка торгового оборудования: фискальные регистраторы, терминалы сбора данных, сканеры штрихкодов, весовое оборудование, дисплеи покупателя, платежные терминалы, ридеры магнитных карт.
- [8]

Ниже приведен пример внедрения продукта:

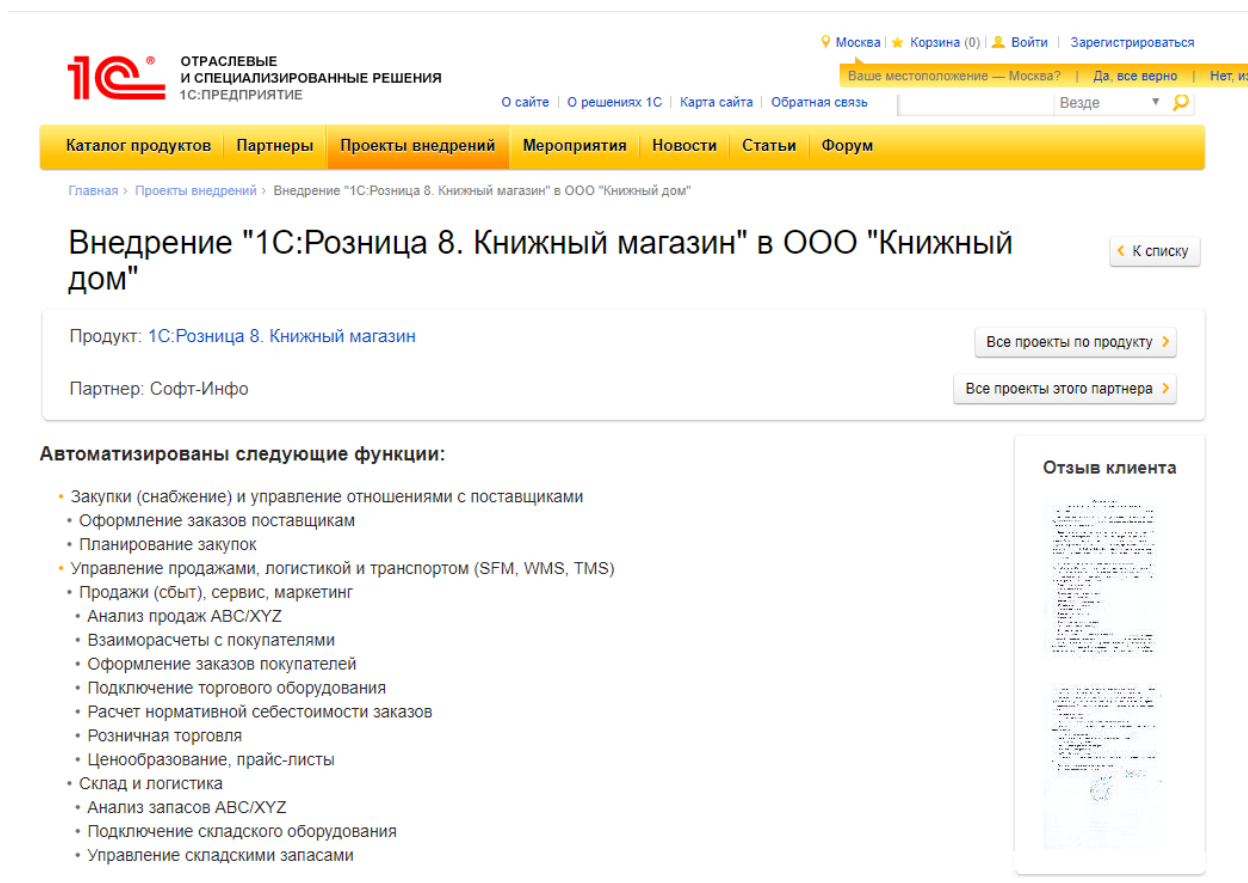


Рисунок 3 – пример внедрения системы 1С:Розница

В результате мы имеем многофункциональную русскоязычную платформу, покрывающую все базовые требования магазина отрасли. Не поддерживает системы лояльности. Любой продукты 1С можно скорректировать под индивидуальные нужды заказчика, но это потребует постоянного штата разработчиков, для поддержки и обновления системы.

1.4 Требования к разрабатываемой базе данных ИС Книжного магазина

На основе системного анализа предметной области и аналогов к базе данных были сформированы следующие требования:

С базой данных могут работать следующие группы пользователей:

- Администрация (Директор, Бухгалтерия)
- Менеджер (Администратор филиала, менеджер склада)
- Клиент

Администрация при работе с базой данных может выполнять задачи по изменению/добавлению/удалению информации о: Сотрудниках; Клиентах; Книгах; Магазинов; Заявках; Заказах; Поставках. А также просматривать любую информацию.

Пользователи группы **«менеджер»** могут выполнять следующие задачи: Добавление информации о заявках; Изменение информации о: перечне и количестве товаров в магазине/филиале и Статусе заказа клиента; Просмотр информации о: Чеках; Заявках; Поставках; Книгах.

При работе с базой данных **клиент** может: Вносить изменения в своих данных; Просматривать всю информацию о книгах; Добавлять заказ; Просматривать информацию о заказе.

Для данной базы требуется предусмотреть следующие ограничения:

- Обязательное заполнения всех данных в таблицах сотрудники, магазины, заявки, поставки, заказы;
- При заказе/регистрации обязательное заполнение ФИО, электронной почты и телефона клиента;
- Возраст сотрудников от 18.

Для автоматизации обработки информации в базе данных следует ввести следующие процессы:

- Подсчет количества книг в сети, на основе информации по филиалам/складам;
- Изменение статуса скидки в зависимости от суммы заказов за весь период;
- Изменение количества товара в результате продажи.

1.5 Выводы

В первой главе проведен системный анализ предметной области книжного магазина, в ходе которого перечислены должности работников, выявлены основные объекты предметной области с их основными характеристиками, а также сформулированы ограничения, накладываемые на информацию, содержащуюся в системе.

В ходе обзора информационных технологий, подходящих для реализации данной информационной системы, рассмотрены основные имеющиеся в данный момент на рынке СУБД, а именно: Microsoft Access, Microsoft SQL Server, Oracle Database, MySQL. Приведён обзор некоторых популярных средств для разработки пользовательских приложений – языки программирования C#, Python.

Рассмотрены продукты-аналоги разрабатываемой автоматизированной системы управления («РосБизнесСофт», «Magento», «1с: УНФ»).

В заключении первой главы сформулированы требования, предъявляемые к базе данных: выделен список групп пользователей системы,

перечислены выполняемые каждой группой задачи. Также разработан список действий, которые необходимо автоматизировать.

Таким образом, в главе в полном объёме осуществлена подготовка к этапу проектирования базы данных.

Глава 2. Проектирование БД для ИС «Книжный магазин»

2.1 Разработка инфологической модели

Инфологическая модель – частично формализованное описание предметной области, которое имеет удобное представление, как для специалистов баз данных, так и для пользователей.

Целью инфологического проектирования является создание структурированной информационной модели предметной области, для которой будет разрабатываться база данных.

При проектировании на инфологическом уровне создается информационно-логическая модель, которая должна отвечать следующим требованиям:

- обеспечение наиболее естественных для человека способов сбора и представления той информации, которую предполагается хранить в создаваемой базе данных;
- корректность схемы БД - адекватное отображение смоделированной предметной области;
- простота и удобство использования на следующих этапах проектирования, то есть может легко отображаться на выбранной модели базы данных;
- должна быть описана языком, понятным проектировщика БД, программистам, администратору и пользователям.

Суть инфологического моделирования состоит в выделении сущностей (информационных объектов предметной области), которые подлежат хранению в базе данных, а также в определении характеристик объектов и взаимосвязей между ними.

Для информационной системы «Книжный магазин» на основе системного анализа предметной области выделены следующие сущности:

1. Сотрудники: сущность содержит информацию о сотрудниках;
2. Клиенты: сущность содержит информацию о клиентах магазина;
3. Помещения: сущность содержит информацию о торговых залах и складах;
4. Заказы: сущность содержит информацию обо всех заказах, оформленных покупателями;
5. Чеки: сущность содержит информацию о покупках, совершенных в торговых залах;
6. Книги: сущность содержит информацию об ассортименте магазина.
7. Поставки: сущность хранит информацию о поставках товара;

8. Заявки: сущность хранит информацию о запросах торговых залов на пополнение ассортимента со склада.

Для ИС «Книжный магазин» на основе системного анализа предметной области выделены следующие связи между сущностями:

1. В заказе присутствует множество книг, каждая книга может быть во множестве заказов – связь «многие-ко-многим» между сущностями Книги и Заказ;
2. В каждом помещении может находиться множество книг, книга может находиться во многих помещениях – связь «многие-ко-многим» между сущностями Книги и Помещения;
3. Множество книг может быть в заявке, каждая книга может быть в разных заявках – связь «многие-ко-многим» между сущностями Книги и Заявки;
4. Множество книг может быть в поставке, каждая книга может быть в разных поставках – связь «многие-ко-многим» между сущностями Книги и Поставки;
5. Клиент может совершить множество заказов, но в каждый заказ совершен только одним клиентом – связь «один-ко-многим» между сущностями Клиенты и Заказы;
6. Каждому заказу соответствует один чек – связь «один-к-одному» между сущностями Заказы и Чеки;
7. Каждый сотрудник может обрабатывать множество заказов, но каждый заказ обработан одним сотрудником – связь «один-ко-многим» между сущностями Сотрудники и Заказы;
8. Каждый сотрудник может составить множество заявок, но заявка составляется только одним сотрудником – связь «один-ко-многим» между сущностями Сотрудники и Заявки;
9. Каждый сотрудник может принять множество поставок, но поставка принимается только одним сотрудником – связь «один-ко-многим» между сущностями Сотрудники и Поставки;
10. Каждый сотрудник работает только в одном помещении, но в одном помещении может работать множество сотрудников – связь «один-ко-многим» между сущностями Помещения и Сотрудники;
11. Множество заявок может быть сформировано для одного помещения – связь «один-ко-многим» между сущностями Помещения и Заявки;
12. Множество поставок может быть назначено на одно помещение – связь «один-ко-многим» между сущностями Помещения и Поставки;

Для построения инфологической модели будет использован инструмент Visual Paradigm.

Visual Paradigm Suite — это набор средств для моделирования информационных систем и бизнес-процессов, генерации кода на базе построенных моделей, проектирования БД и решения многих других задач.

VP for UML – это система управления требованиями, поддерживающая полный цикл разработки программного продукта – анализ, дизайн архитектуры, разработка программного кода, тестирование и размещение продукта на стороне заказчика. VP также обеспечивает поддержку версии и одновременной работы команды пользователей над одним проектом.

Исходя из приведенных выше сущностей и связей между ними построена инфологическая модель предметной области, представленная на рисунке 2:

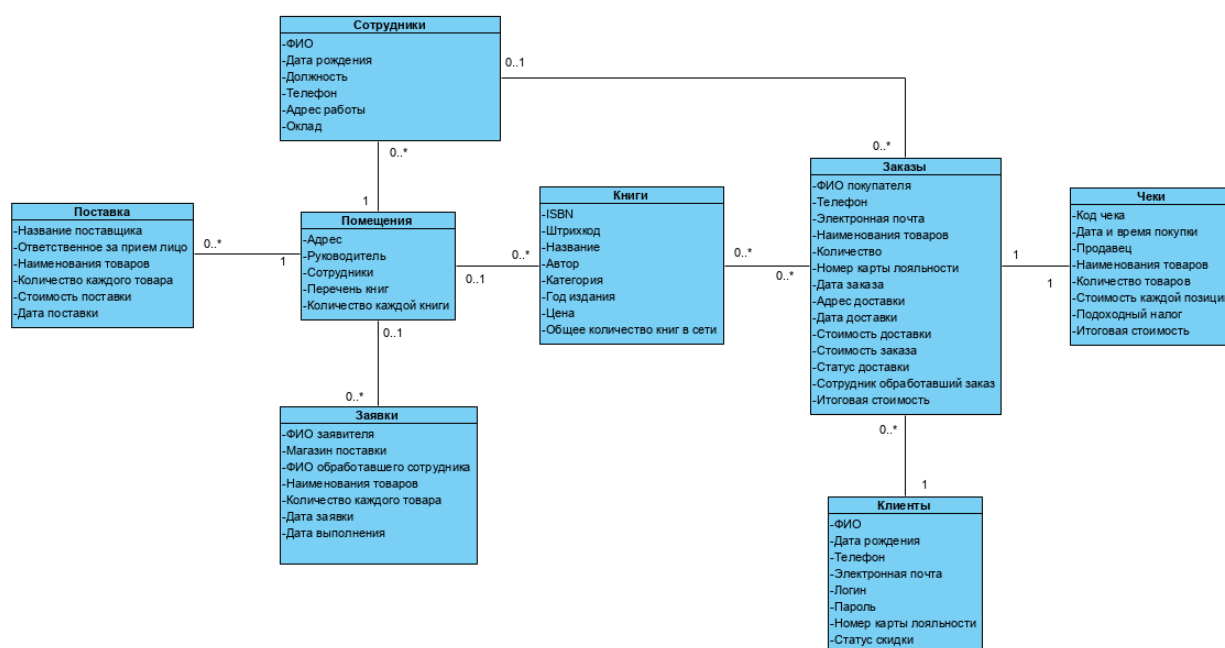


Рисунок 4 – инфологическая модель предметной области

2.2 Обоснование выбора даталогической модели данных

Под даталогической моделью понимается модель, отражающая логические взаимосвязи между элементами данных безотносительно их содержания и физической организации. Даталогическая модель строится на основе инфологической модели конкретной предметной области, с учетом ее особенностей.

Существуют различные виды даталогической модели данных:

- Иерархическая

- Сетевая
- Объектно-ориентированная
- Реляционная

Рассмотрим каждый вид даталогической модели данных из представленных выше.

Иерархическая модель данных

Иерархическая модель позволяет строить иерархию элементов. У каждого элемента может быть несколько “наследников” и существует один “родитель”. Для каждого уровня связи вводится интерпретация, зависящая от предметной области и описывающая взаимоотношение между “родителями” и “наследниками”. Каждый элемент представляется с помощью записи. Структура данных, обычно используемая для представления этой записи об элементе, содержит некоторые атрибуты, характеристики каждого элемента.

Иерархическая модель представляет собой связные неориентированный граф древовидной структуры, объединяющий сегменты. Иерархическая БД состоит из упорядоченного набора деревьев.

У иерархических СУБД есть достоинства и недостатки. К достоинствам относится возможность реализовать фантастически быстрый поиск нужных значений, когда условия запроса соответствуют иерархии в схеме базы данных. С другой стороны, если запрос не соответствует имеющейся иерархии, то и его программирование, и его исполнение, потребуют значительных усилий.

Другим недостатком иерархической модели является сложность внесения в нее изменений. Если, по каким-то причинам изменились условия задачи, и модель предметной области перестала быть иерархической, то приведение схемы базы данных в соответствие предметной области становится нетривиальной задачей.

Иерархическая модель очень хорошо подходит для устоявшихся предметных областей с четкими зависимостями “родитель-потомок”, то есть к моделям, где есть четкая субординация между понятиями. Там же, где эти условия выполнены, проявляются достоинства иерархической модели - очень быстрая скорость поиска.

Сетевая модель

Стандарт сетевой модели впервые был определен в 1975 году организацией CODASYL (Conference of Data System Languages), которая определила базовые понятия модели и формальный язык описания.

Сетевая модель данных – логическая модель, являющаяся решением иерархического подхода. Разница между иерархической моделью данных заключается в том, что в иерархических структурах данных запись-потомок должна иметь в точности одного предка, а в сетевой структуре данных у потомка может быть любое число предков.

Сетевую модель можно представить, как граф узлами, которого является запись, а ребрами – связи между ними. Сегменты данных в сетевых БД могут иметь множественные связи с сегментами старшего уровня. При этом направление и характер связи в сетевых БД не являются столь очевидными, как в случае иерархических БД. Поэтому имена и направление связей должны идентифицироваться при описании БД.

Недостатком сетевой модели данных являются высокая сложность и жесткость схемы БД, построенной на её основе. Поскольку логика процедуры выборки данных зависит от физической организации этих данных, то эта модель не является полностью независимой от приложения. Другими словами, если необходимо изменить структуру данных, то нужно изменить и приложение.

Объектно-ориентированная модель

Объектно-ориентированная модель данных является расширением положений объектно-ориентированного программирования (в то время как реляционная модель возникла на основе теории множеств, именно как модель данных). Группой управления Объектно-ориентированных БД разработан стандарт ODMG-93 (Object DataBase Management Group). Этот стандарт полностью еще не реализован.

Структура объектно-ориентированной БД графически представима в виде дерева, узлами которого являются объекты. Видимая структура объекта определяется свойствами его класса. Класс включает в себя объекты, при этом структура и поведение объектов одного класса одинаковы. Каждый объект, а именно экземпляр класса считается потомком объекта, в котором он определен как свойство. Свойства объектов - или стандартный тип, например, string, или конструируемый пользователем тип class. Поведение объектов задается с помощью методов. Метод – это некая операция, которую можно применить к объекту.

Достоинствами объектно-ориентированной модели в сравнении с реляционной является возможность отображения информации о сложных взаимосвязях объектов, отсутствие ограниченности в структурах хранения данных. В объектно-ориентированной БД может храниться не только структура, но и поведенческие аспекты данных. С использованием объектно-

ориентированного подхода могут создаваться и БД с большими объемами семантической информации, такие как мультимедийные и специализированные для конкретных предметных областей (географические, проектные и др.).

К недостаткам данного подхода можно отнести высокую понятийную сложность, неудобство обработки данных и низкую скорость выполнения запросов.

Реляционная модель

В реляционной модели достигается гораздо более высокий уровень абстракции данных, чем в иерархической или сетевой. Реляционная модель предоставляет средства описания данных на основе только их естественной структуры, т.е. без потребности введения какой-либо дополнительной структуры для целей машинного представления. Другими словами, представление данных не зависит от способа их физической организации. Это обеспечивается за счет использования математической теории отношений (само название "реляционная" происходит от английского relation - "отношение").

Структура информации дает основание предполагать, что наиболее подходящей для датологического проектирования будет реляционная модель данных, т.к. она способна обеспечить целостность данных при вставке, удалении и изменении записей, а так же дает возможность организации всех видов связей 1:1, 1:M и M:M (при этом связи M:M раскрываются). К недостаткам традиционных реляционных моделей данных можно отнести является избыточность по полям (из-за создания связей), а также факт того, что в качестве основного и, часто, единственного механизма, обеспечивающего быстрый поиск и выборку отдельных строк таблицы (или в связанных через внешние ключи таблицах), обычно используются различные модификации индексов, основанных на В-деревьях. Такое решение оказывается эффективным только при обработке небольших групп записей и высокой интенсивности модификации данных в базах данных.

Основным достоинством реляционных баз данных является совместимость с самым популярным языком запросов SQL.

С помощью единственного запроса на этом языке можно соединить несколько таблиц во временную таблицу и вырезать из нее требуемые строки и столбцы (селекция и проекция). Так как табличная структура реляционной базы данных интуитивно понятна пользователям, то и язык SQL является простым и легким для изучения. Реляционная модель имеет солидный теоретический фундамент, на котором были основаны эволюция и реализация

реляционных баз данных. На волне популярности, вызванной успехом реляционной модели, SQL стал основным языком для реляционных баз данных.

Но выявлены и недостатки рассмотренной модели баз данных:

- Так как все поля одной таблицы должны содержать постоянное число полей заранее определенных типов, приходится создавать дополнительные таблицы, учитывающие индивидуальные особенности элементов, при помощи внешних ключей. Такой подход сильно усложняет создание сколько-нибудь сложных взаимосвязей в базе данных;
- Высокая трудоемкость манипулирования информацией и изменения связей.

2.3 Даталогическое проектирование

Для логического проектирования была выбрана реляционная модель данных, т.к. она наиболее полно соответствует требованиям, предъявленным к разрабатываемой информационной системе:

- Отсутствие дублируемой информации;
- Поддержание целостности данных при вставке, удалении или изменении записей;
- Возможность организации всех видов связи между отношениями 1:1, 1:M и M:N.

В реляционной базе данных даталогическое проектирование приводит к разработке схемы базы данных – совокупности отношений, которые адекватно моделируют абстрактные объекты предметной области и семантические связи между ними. Схема должна быть корректной, т.е. такой схемой, в которой отсутствуют нежелательные зависимости между атрибутами. При этом можно использовать процесс проектирования с помощью декомпозиции, т.е. последовательно нормализовывать схему отношений, тем самым накладывая ограничения и избавляясь от нежелательных зависимостей между атрибутами.

Основой анализа корректности схемы являются функциональные зависимости между атрибутами БД. Функциональные зависимости определяют устойчивые отношения между атрибутами отношений. Некоторые могут быть нежелательными.

Переход от инфологической модели к реляционной

Для перехода от инфологической модели к реляционной воспользуемся существующим алгоритмом:

1. Каждой сущности ставится в соответствие отношение;
2. Каждому атрибуту сущности ставится в соответствие соответствующий атрибут;
3. Первичный ключ сущности становится РК соответствующего отношения, при этом атрибуты, входящие в РК, обязательны для заполнения (NOT NULL);
4. В каждое отношение, соответствующее подчиненной сущности, добавляется набор атрибутов основной сущности, являющийся в ней первичным ключом. В отношении, соответствующее подчиненной сущности эти атрибуты становятся FK (внешним ключом).
5. При реализации связи многие-ко-многим, допустимой в инфологической модели, производится ее преобразование к связям один-ко-многим, например, через промежуточное отношение. Промежуточное отношение будет иметь первичный ключ, состоящий из первичных ключей связываемых отношений.

Пользуясь приведенным выше алгоритмом, опишем каждую сущность инфологической модели:

1. Книги – books:

- a. ISBN книги – ISBN varchar(13) NOT NULL PRIMARY KEY
- b. Название – title varchar(45)
- c. Автор – author varchar(45)
- d. Категория – category varchar(45)
- e. Год издания – publish_year year(4)
- f. Цена – price number(7,2) DECIMAL(7,2)
- g. Количество книг в сети – book_count int

2. Заказы – orders:

- a. Идентификатор заказа – id int NOT NULL PK
- b. Идентификатор клиента – client_id int NOT NULL FK
- c. Идентификатор книги – product_id int NOT NULL FK
- d. Количество каждой позиции – product_count int NOT NULL
- e. Общее количество товаров – total_count int
- f. Дата заказа – order_date date
- g. Адрес доставки – delivery_adress varchar(50)
- h. Дата доставки – delivery_date date
- i. Стоимость доставки – delivery_cost number(6,2)
- j. Стоимость заказа – order_cost number(7,2)
- k. Статус доставки – delivery_status bool
- l. Идентификатор сотрудника – employee_id int NOT NULL FK
- m. Итоговая стоимость – total_price number(7,2)

3. Клиенты – clients:

- a. Идентификатор клиента – id int NOT NULL PK
- b. ФИО клиента – full_name varchar(40)
- c. Дата рождения клиента – birth_date date
- d. Электронная почта – email varchar(25)
- e. Логин – login varchar(15)
- f. Пароль – password varchar(25)
- g. Номер карты лояльности – loyalty_card varchar(15)
- h. Статус скидки – discount_status int

4. Чеки – cheque:

- a. Идентификатор чека – id int NOT NULL PK
- b. Дата и время покупки – purchase_date date
- c. Идентификатор заказа – order_id int NOT NULL FK
- d. Подоходный налог – tax int NOT NULL

5. Помещения – spaces:

- a. Идентификатор помещения – space_id int NOT NULL PK
- b. Адрес помещения – space_adress varchar(100)
- c. Тип помещения – type ENUM("store", "warehouse")
- d. Перечень книг – book_id int NOT NULL FK

6. Сотрудники – employees:

- a. Идентификатор сотрудника – id int NOT NULL PK
- b. ФИО сотрудника – full_name varchar(100)
- c. Дата рождения – birth_date date
- d. Должность – position varchar(45)
- e. Телефон – phone_number varchar(11)
- f. Идентификатор помещения – space_id int NOT NULL FK
- g. Оклад – salary number(8,2)

7. Заявки – requests:

- a. Идентификатор заявки – request_id int NOT NULL PK
- b. ФИО заявителя – applicant_id int NOT NULL FK
- c. Магазин поставки – space_id int NOT NULL FK
- d. ФИО обработавшего сотрудника – employee_id int NOT NULL FK
- e. Идентификатор книг – book_id int NOT NULL FK
- f. Количество каждого товара – book_count int NOT NULL
- g. Дата заявки – request_date date
- h. Дата выполнения – complete_date date

8. Поставки – supplies:

- a. Идентификатор поставки – supply_id int NOT NULL PK

- b. Название поставщика – supplier varchar(20)
- c. Идентификатор сотрудника – employee_id int NOT NULL FK
- d. Идентификатор книги – book_id int NOT NULL FK
- e. Количество каждого товара – book_count int
- f. Стоимость поставки – supply_cost number (8,2)
- g. Дата поставки – supply_date date

При переходе от инфологической модели к даталогической были раскрыты связи типа «многие-ко-многим», в результате чего появились следующие таблицы:

1. Книги в заказах – books_has_orders:
 - a. id INT UNSIGNED NOT NULL AUTO_INCREMENT,
 - b. books_ISBN VARCHAR(13) NOT NULL,
 - c. orders_id INT UNSIGNED NOT NULL,
 - d. book_count INT,
 - e. position_cost DECIMAL(7,2),
2. Сотрудники, создавшие заявку – employees_has_requests:
 - a. Идентификатор записи – id INT NOT NULL,
 - b. Код сотрудника – employees_id INT NOT NULL,
 - c. Код заявки – requests_id INT NOT NULL,
3. Книги, содержащиеся в помещении – spaces_has_books:
 - a. Идентификатор записи – `id` INT NOT NULL,
 - b. Код помещения – `spaces_id` INT NOT NULL,
 - c. Код книги – `books_ISBN` VARCHAR(13) NOT NULL,
 - d. Количество в помещении – `count_in_space` INT,

Схема полученной реляционной модели данных представлена на рисунке 5.

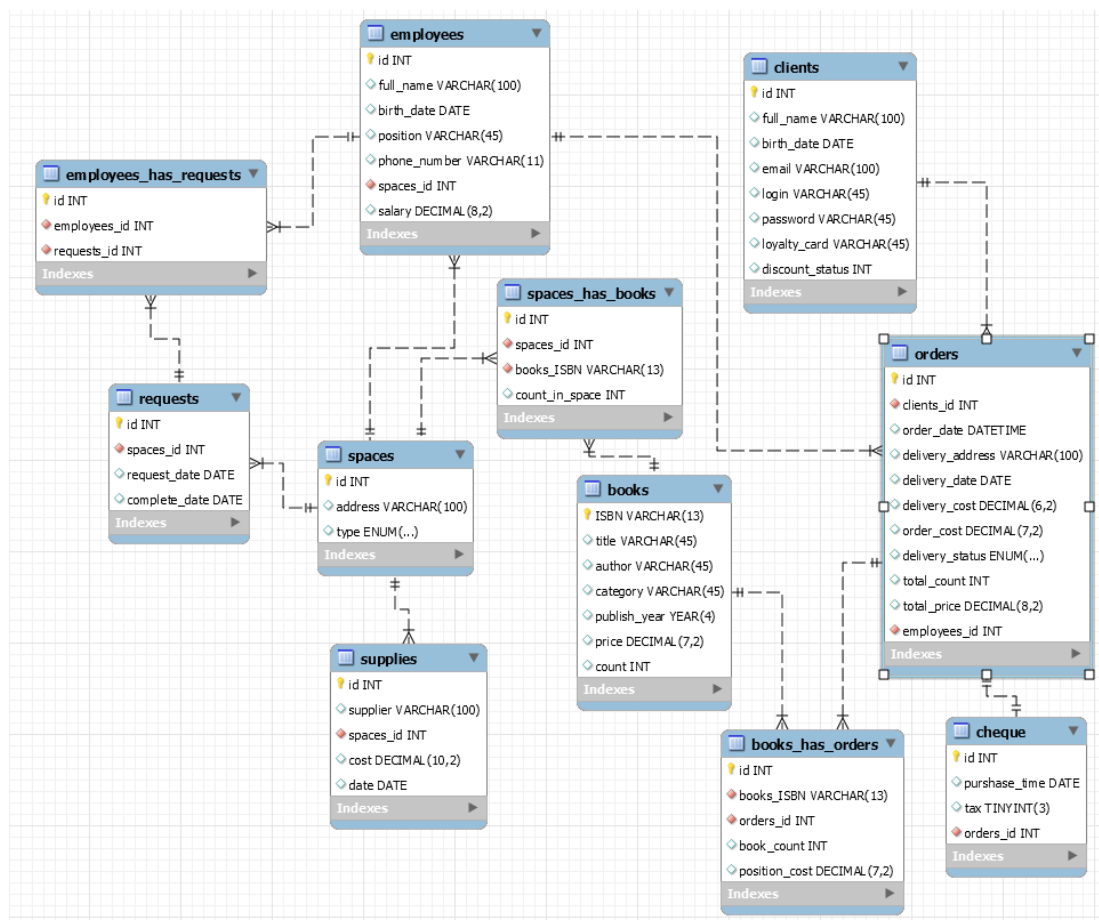


Рисунок 5 – даталогическая модель ИС Книжного магазина

2.4. Нормализация схемы базы данных «Книжный магазин»

Нормализация предназначена для приведения структуры БД к виду, обеспечивающему минимальную логическую избыточность, и не имеет целью уменьшение или увеличение производительности работы или же уменьшение, или увеличение физического объёма базы данных. Конечной целью нормализации является уменьшение потенциальной противоречивости, хранимой в базе данных информации. Общее назначение процесса нормализации заключается в следующем:

- Исключение некоторых типов избыточности;
- Устранение некоторых аномалий обновления;
- Разработка проекта базы данных, который является достаточно «качественным» представлением реального мира, интуитивно понятен и может служить хорошей основой для последующего расширения;

- Упрощение процедуры применения необходимых ограничений целостности.

Устранение избыточности производится, как правило, за счёт декомпозиции отношений таким образом, чтобы в каждом отношении хранились только первичные факты (то есть факты, не выводимые из других хранимых фактов).

Нормализация – процесс приведения отношений базы данных к виду, отвечающему нормальным формам.

Нормальная форма — свойство отношения в реляционной модели данных, характеризующее его с точки зрения избыточности, потенциально приводящей к логически ошибочным результатам выборки или изменения данных. Нормальная форма определяется как совокупность требований, которым должно удовлетворять отношение.

- Первая нормальная форма: отношение находится в 1НФ, если все его атрибуты являются простыми, все используемые домены должны содержать только скалярные значения. Не должно быть повторений строк в таблице.
- Вторая нормальная форма: Отношение находится во 2НФ, если оно находится в 1НФ и каждый не ключевой атрибут неприводимо зависит от первичного ключа.
- Третья нормальная форма: Отношение находится в 3НФ, когда находится во 2НФ и каждый не ключевой атрибут нетранзитивно зависит от первичного ключа.
- Нормальная форма Бойса-Кодда: Отношение находится в НФБК, когда каждая нетривиальная и неприводимая слева функциональная зависимость обладает потенциальным ключом в качестве детерминанта.
- Четвертая нормальная форма: Отношение находится в 4НФ, если оно находится в НФБК и все нетривиальные многозначные зависимости фактически являются функциональными зависимостями от ее потенциальных ключей.

Согласно определениям нормальных форм, представленная выше даталогическая модель приведена к нормальной форме Бойса-Кодда, поэтому нет необходимости в дальнейшей нормализации.

2.5 Выводы

Во второй главе была описана разработка инфологической модели базы данных ИС «Книжный магазин», содержащей 8 сущностей и 12 связей, четыре из которых реализуют связь «многие-ко-многим».

Было дано краткое описание даталогических моделей – иерархической, сетевой, объектно-ориентированной и реляционной, определены достоинства и недостатки каждой модели. На основе анализа в качестве был произведен выбор в сторону реляционной модели.

По инфологической модели была построена даталогическая модель данных, дан список атрибутов и отношений, проведена нормализация до нормальной формы Бойса-Кодда. Таким образом завершено проектирование базы данных, получена вся информация, необходимая для реализации базы данных проектируемой информационной системы в одной из реляционных СУБД.

Глава 3. Программная реализация БД «Книжный магазин»

3.1 Анализ и выбор СУБД

MySQL — свободная реляционная система управления базами данных. Разработку и поддержку MySQL осуществляет корпорация Oracle, получившая права на торговую марку вместе с поглощённой Sun Microsystems, которая ранее приобрела шведскую компанию MySQL AB. Продукт распространяется как под GNU General Public License, так и под собственной коммерческой лицензией. Помимо этого, разработчики создают функциональность по заказу лицензионных пользователей.

MySQL является решением для малых и средних приложений. Обычно MySQL используется в качестве сервера, к которому обращаются локальные или удалённые клиенты, однако в дистрибутив входит библиотека внутреннего сервера, позволяющая включать MySQL в автономные программы.

3.2 Физическое проектирование БД «Книжный магазин»

Для проектирования базы данных была выбрана СУБД MySQL. Разработка велась через ПО MySQL Workbench, поставляемой в общем пакете продукта MySQL. На основе даталогической модели произведена программная реализация. База данных содержит 11 таблиц, перечисленных на рисунке 6 в результате успешного создания базы данных:


















	Название	Движок	Авто-увеличение	Размер	Описание
 Таблицы	 books	InnoDB	0	16 384	
 Представления	 books_has_orders	InnoDB	1	16 384	
 Индексы	 cheque	InnoDB	1	16 384	
 Процедуры	 clients	InnoDB	1	16 384	
 Триггеры	 employees	InnoDB	1	16 384	
 События	 employees_has_requests	InnoDB	1	16 384	
	 orders	InnoDB	1	16 384	
	 requests	InnoDB	1	16 384	
	 spaces	InnoDB	1	16 384	
	 spaces_has_books	InnoDB	1	16 384	
	 supplies	InnoDB	1	16 384	

Рисунок 6 – результат создания таблиц базы данных

Пример атрибутов таблицы и ее содержимого будет представлен на рисунках 7 и 8 соответственно:

Название	#	Тип данных	Not NULL	Авто-увеличение	Ключ
ABC ISBN	1	varchar(13)	<input checked="" type="checkbox"/>	<input type="checkbox"/>	PRI
ABC title	2	varchar(45)	<input type="checkbox"/>	<input type="checkbox"/>	
ABC author	3	varchar(45)	<input type="checkbox"/>	<input type="checkbox"/>	
ABC categ...	4	varchar(45)	<input type="checkbox"/>	<input type="checkbox"/>	
🕒 publi...	5	year	<input type="checkbox"/>	<input type="checkbox"/>	
123 price	6	decimal(7,2) unsigned	<input type="checkbox"/>	<input type="checkbox"/>	
123 count	7	int unsigned	<input type="checkbox"/>	<input type="checkbox"/>	

Рисунок 7 – состав таблицы books

<div>books</div> <div>Введите SQL выражение чтобы отфильтровать результаты</div>									
Таблица		<div>ABC ISBN</div>	<div>ABC title</div>	<div>ABC author</div>	<div>ABC category</div>	<div>publish_year</div>	<div>123 price</div>	<div>123 count</div>	
	1	9785170738472	Полная история Средиземья	Джон Р.Р.Толкин	Фэнтези	2013	986	50	
Текст	2	9785170786961	Властелин колец	Джон Р.Р.Толкин	Фэнтези	2013	748,5	85	
	3	9785170905430	Чужб земли	Макс Фрай	Фэнтези	2018	402	150	
	4	9785170923212	Властелин Морморы	Макс Фрай	Фэнтези	2018	405	150	
	5	9785170924554	Неуловимый Хабба Хэн	Макс Фрай	Фэнтези	2018	450	200	
	6	9785170924561	Ворона на мосту	Макс Фрай	Фэнтези	2018	390	145	
	7	9785498075808	Психология лжи	Пол Экман	Психология	2010	500	325	
	8	9785769548789	Психология общения	В.А.Горянина	Психология	2004	358,5	77	
	9	9785928707088	Сонник	Г.Х.Миллер	Эзотерика	2004	160	25	
	10	9785941422306	Личный ребрендинг	Т.Мужицкая	Саморазвитие	2013	250	102	
	11	9785942787468	Молот ведьм	Яков Шпренгер	Эзотерика	2005	268	50	
	12	9785988571230	Рунические практики	Светлана Некрасова	Эзотерика	2008	215	60	

Рисунок 8 – содержимое таблицы books

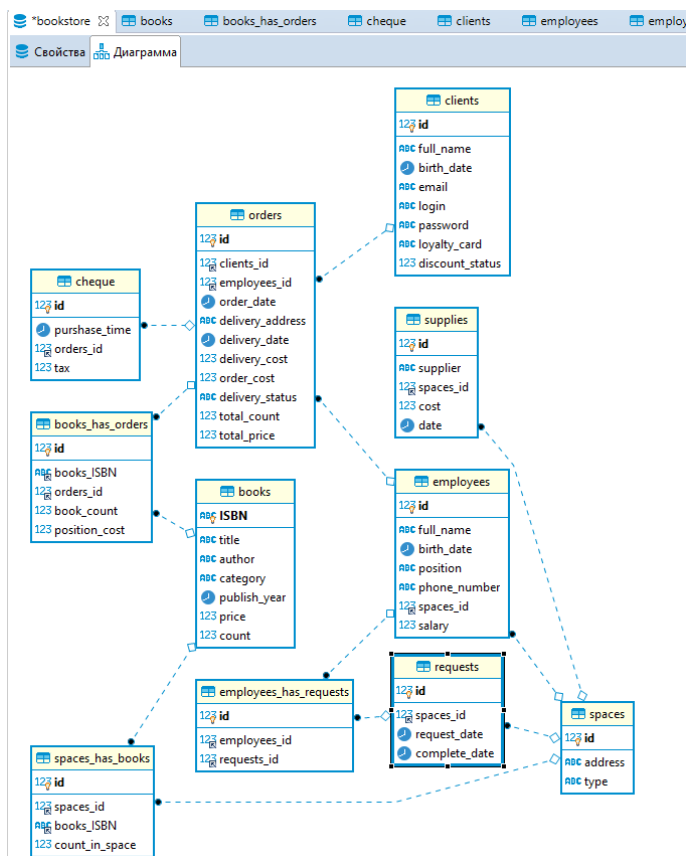


Рисунок 9 – схемы созданной ПО

3.3 Разработка представлений для БД «Книжный магазин»

Для вывода часто запрашиваемой информации были созданы следующие представления:

1. All_orders;
2. Not_enough_books;
3. Not_paid_orders;
4. Report_mounth.

Все представления были успешно созданы. Листинги представлений размещены в «приложении 3».

Представление all_orders содержит список всех клиентов, совершавших заказы и общую сумму их заказов за весь период.

	ABC full_name	123 total_sum
1	Марселина Хоуп	1 647
2	Иосиф Блумач	858,5
3	Николай Джаллибов	2 542
4	Элеонора Федотова	268
5	Егор Шастун	2 010

Рисунок 10 – результат выполнения представления all_orders

Представление not_enough_books содержит список магазинов, в которых количество книг, согласно конкретному артикулу, меньше 10 (см. рисунок 11). На тестовой проверке все книги находятся на складе, в результате чего во всех магазинах количество книг является недостаточным.

	123 spaces_id	ABC ISBN	ABC title	123 count_in_space
1	2	9785170905430	Чуб земли	0
2	3	9785170905430	Чуб земли	0
3	4	9785170905430	Чуб земли	0
4	5	9785170905430	Чуб земли	0
5	2	9785170924554	Неуловимый Хабба Хэн	0
6	3	9785170924554	Неуловимый Хабба Хэн	0
7	4	9785170924554	Неуловимый Хабба Хэн	0
8	5	9785170924554	Неуловимый Хабба Хэн	0
9	2	9785170923212	Властелин Морморы	0
10	3	9785170923212	Властелин Морморы	0
11	4	9785170923212	Властелин Морморы	0
12	5	9785170923212	Властелин Морморы	0
13	2	9785170924561	Ворона на мосту	0
14	3	9785170924561	Ворона на мосту	0
15	4	9785170924561	Ворона на мосту	0
16	5	9785170924561	Ворона на мосту	0
17	2	9785170786961	Властелин колец	0
18	3	9785170786961	Властелин колец	0
19	4	9785170786961	Властелин колец	0
20	5	9785170786961	Властелин колец	0
21	2	9785170738472	Полная история Средиземья	0
22	3	9785170738472	Полная история Средиземья	0
23	4	9785170738472	Полная история Средиземья	0
24	5	9785170738472	Полная история Средиземья	0
25	2	9785941422306	Личный ребрендинг	0
26	3	9785941422306	Личный ребрендинг	0
27	4	9785941422306	Личный ребрендинг	0
28	5	9785941422306	Личный ребрендинг	0
29	2	9785769548789	Психология общения	0
30	3	9785769548789	Психология общения	0
31	4	9785769548789	Психология общения	0
32	5	9785769548789	Психология общения	0
33	2	9785498075808	Психология лжи	0
34	3	9785498075808	Психология лжи	0
35	4	9785498075808	Психология лжи	0
36	5	9785498075808	Психология лжи	0
37	2	9785988571230	Рунические практики	0
38	3	9785988571230	Рунические практики	0
39	4	9785988571230	Рунические практики	0
40	5	9785988571230	Рунические практики	0

Рисунок 11 – результат выполнения представления not_enough_books

Представление not_paid_orders отражает номера тех заказов, которые являются неоплаченными в данный момент:

	123 id
1	4
2	5

Рисунок 12 – результат выполнения представления not_paid_orders

Представление report_month предоставляет отчет по продажам за текущий календарный месяц (артикул и количество проданных товаров):

	ABC books_ISBN	123 count
1	9785928707088	2
2	9785941422306	1

Рисунок 13 – результат выполнения представления report_month

3.4. Разработка триггеров для БД «Книжный магазин»

Для обработки вновь добавленной информации о количестве книг в системе и о статусе скидки клиентов после совершения покупки были разработаны следующие триггеры:

1. Decrease_count;
2. Update_total_count;
3. Discount_status.

Все триггеры были успешно созданы. Коды триггеров размещены в приложении 4.

Триггер Decrease_count выполняет уменьшение количество книг в помещениях после появления чеков, которые сигнализируют об оплате заказа.

Триггер Update_total_count при изменении количества книг в конкретном помещении пересчитывается общее количество книг в сети.

Триггер Discount_status выполняет следующую роль: при оплате покупки, пересчитывает общую сумму покупок пользователя, совершившего заказ и, в зависимости от размера этой суммы, устанавливает его скидку в системе.

3.5 Реализация ограничений для БД «Книжный магазин»

С базой данных могут работать 3 типа пользователей: администрация, менеджеры и клиенты.

Администрация при работе с базой данных может выполнять задачи по изменению/добавлению/удалению информации о: Сотрудниках; Клиентах; Книгах; Магазилах; Заявках; Заказах; Поставках. А также просматривать любую информацию.

Пользователи группы **«менеджер»** могут выполнять следующие задачи: Добавление информации о заявках; Изменение информации о: перечне и количестве товаров в магазине/филиале и Статусе заказа клиента; Просмотр информации о: Чеках; Заявках; Поставках; Книгах.

При работе с базой данных **клиент** может: Вносить изменения в своих данных; Просматривать всю информацию о книгах; Добавлять заказ; Просматривать информацию о заказе.

В рамках разработки интерфейса для взаимодействия с БД были реализованы представители групп менеджеры и клиенты, для демонстрации их возможностей.

Листинг создания пользователей с ролью и присваиванием им прав доступа отражен в приложении 5.

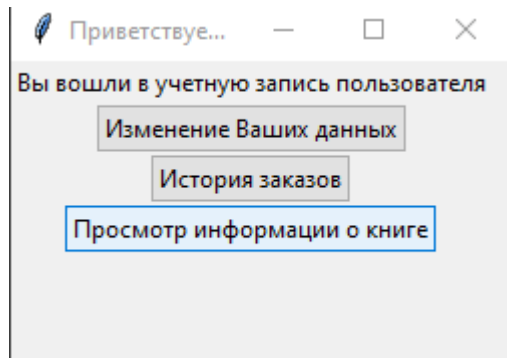


Рисунок 14 – пример окна клиента

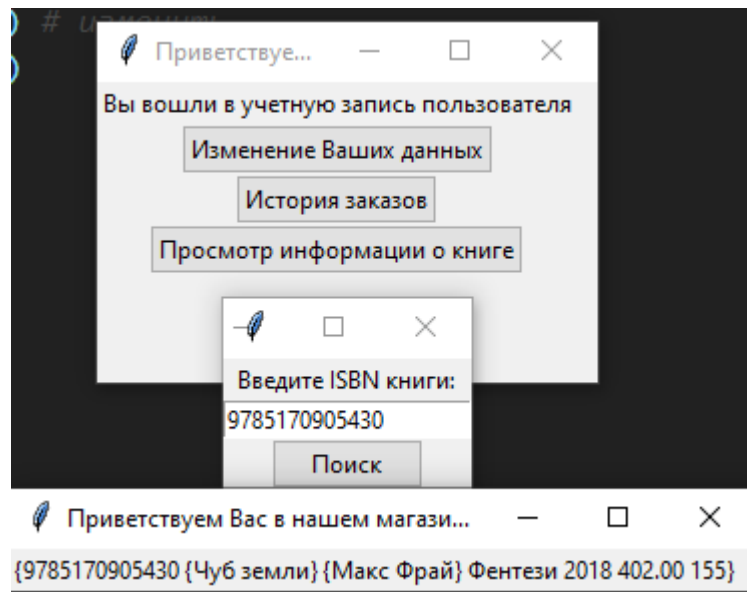


Рисунок 15 – пример запроса пользователя

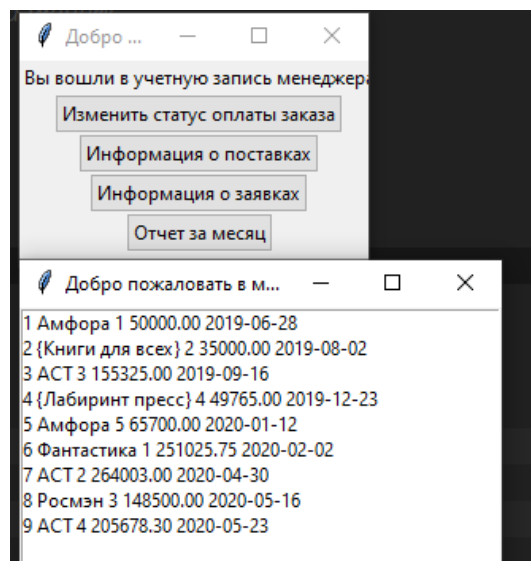


Рисунок 16 – возможности пользователя группы менеджер и пример запроса о поставках

3.6 Безопасность и контроль

Средства безопасности для управления доступом в MySQL подразделяются на 2 этапа:

- Верификация подсоединений;
- Верификация запросов.

Верификация подсоединения осуществляется следующим образом: при попытке соединения с сервером MySQL он либо устанавливает соединение, либо отказывает в нем - на основе данных о вашей личности и того, можете ли вы подтвердить их соответствующим паролем. Если нет, сервер полностью отказывает вам в доступе. В противном случае сервер устанавливает соединение, затем переходит ко второму этапу и ожидает запросов.

Личность задается двумя порциями информации:

- хостом, с которого вы подсоединяетесь
- вашим именем пользователя MySQL

Проверка личности осуществляется с помощью трех полей контекста таблицы user (Host, User и Password). Сервер устанавливает соединение только в том случае, если находит в таблице user запись, в которой имя хоста и имя пользователя совпадают с введенными вами, и вы указываете правильный пароль.

В свою очередь, верификация запроса происходит после установления соединения. Для каждого поступающего запроса сервер проверяет, имеется ли у вас достаточно привилегий для его выполнения, основываясь при этом на типе операции, которую вы хотите выполнить. Теперь в действие вступают поля привилегий в таблицах привилегий. Информация о привилегиях может находиться в любой из таблиц привилегий - user, db, host, tables_priv или columns_priv. Обработка таблиц привилегий осуществляется с помощью команд GRANT и REVOKE.

Системные привилегии — это права на выполнение общих задач, таких как SELECT ANY TABLE и UPDATE ANY TABLE. Объектные привилегии относятся к действиям с определенными элементами базы данных — таблицами, представлениями и последовательностями. Для предоставления привилегий другому пользователю можно использовать оператор GRANT.

3.7. Выводы

В данной главе произведено физическое проектирование базы данных Книжного магазина. Созданы триггеры, представления, формы и отчет. Определены ограничения и произведен анализ безопасности в СУБД MySQL.

Для реализации БД была выбрана СУБД MySQL, в которой было осуществлено физическое проектирование базы данных.

Заключение

Курсовая работа посвящена разработке базы данных книжного магазина. Решены следующие задачи:

- Произведен системный анализ предметной области;
- Построена инфологическая модель;
- Построена даталогическая модель;
- БД реализована в СУБД MySQL.

Результатом работы является реляционная база данных, содержащая элементы автоматизации и обработки данных. В ее состав входит 11 таблиц, 4 представления и 3 триггера.

Список литературы

1. Microsoft Access URL: https://ru.wikipedia.org/wiki/Microsoft_Access
2. Oracle Database 10g URL: <http://www.interface.ru/oracle/OracleDB10g.htm>
3. MySQL URL: <https://ru.bmstu.wiki/MySQL>
4. C Sharp URL: https://ru.wikipedia.org/wiki/C_Sharp
5. Python URL: <https://ru.wikipedia.org/wiki/Python>
6. CRMindex РосБизнесСофт URL:
<https://crmindex.ru/products/rosbiznesssoft>
7. Magento URL: <http://wp.wiki-wiki.ru/wp/index.php/Magento>
8. 1С:Розница 8. Книжный магазин URL:
<https://solutions.1c.ru/catalog/books-store/features>
9. . Ю. А. Шпак. Проектирование баз данных. Просто как дважды два.
Издательство «Эксмо». Москва, 2007 г.
10. Л.И.Воронова-Учебно-методическое пособие «подготовка и оформление
курсовых работ по дисциплине “Базы данных”-Москва,2016-26с.
11. Л.И.Воронова-Лабораторный практикум по дисциплине «базы данных»-
Москва,2014-56с.
- 12.8. Яргер, Р.Дж.; Риз, Дж.; Кинг, Т. MySQL и mSQL: Базы данных для
небольших предприятий и Интернета; СПб: Символ-Плюс, 2013. - 560 с
- 13.9. Дейв Энсор, Йен Стивенсон. Oracle. Проектирование баз данных.
- 14.10. В. Дунаев. Базы данных. Язык SQL для студентов. Издание «БХВ -
Петербург». Санкт - Петербург, 2006 г

Приложения

Приложение 1. Программный код реализации схемы отношений

```
-- MySQL Workbench Forward Engineering

SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0;
SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=0;
SET @OLD_SQL_MODE=@@SQL_MODE,
SQL_MODE='ONLY_FULL_GROUP_BY,STRICT_TRANS_TABLES,NO_ZERO_IN_DATE,NO_ZERO_DATE
,ERROR_FOR_DIVISION_BY_ZERO,NO_ENGINE_SUBSTITUTION';

--
-----
-- Schema BookStore
-----

DROP SCHEMA IF EXISTS `BookStore` ;

--
-----
-- Schema BookStore
-----

CREATE SCHEMA IF NOT EXISTS `BookStore` DEFAULT CHARACTER SET utf8 ;
USE `BookStore` ;

--
-----
-- Table `BookStore`.`books`
-----

DROP TABLE IF EXISTS `BookStore`.`books` ;

CREATE TABLE IF NOT EXISTS `BookStore`.`books` (
  `ISBN` VARCHAR(13) NOT NULL,
  `title` VARCHAR(45) NULL,
  `author` VARCHAR(45) NULL,
  `category` VARCHAR(45) NULL,
  `publish_year` YEAR(4) NULL,
  `price` DECIMAL(7,2) UNSIGNED NULL,
  `count` INT UNSIGNED NULL,
  PRIMARY KEY (`ISBN`),
  UNIQUE INDEX `ISBN_UNIQUE` (`ISBN` ASC) VISIBLE)
ENGINE = InnoDB;

--
-----
-- Table `BookStore`.`spaces`
-----

DROP TABLE IF EXISTS `BookStore`.`spaces` ;

CREATE TABLE IF NOT EXISTS `BookStore`.`spaces` (
  `id` INT UNSIGNED NOT NULL AUTO_INCREMENT,
  `address` VARCHAR(100) NULL,
  `type` ENUM("store", "warehouse") NULL,
  PRIMARY KEY (`id`))
ENGINE = InnoDB;

--
-----
-- Table `BookStore`.`employees`
-----

DROP TABLE IF EXISTS `BookStore`.`employees` ;

CREATE TABLE IF NOT EXISTS `BookStore`.`employees` (
  `id` INT UNSIGNED NOT NULL AUTO_INCREMENT,
  `full_name` VARCHAR(100) NULL,
  `birth_date` DATE NULL,
  `position` VARCHAR(45) NULL,
```

```

`phone_number` VARCHAR(11) NULL,
`spaces_id` INT UNSIGNED NOT NULL,
`salary` DECIMAL(8,2) NULL,
PRIMARY KEY (`id`),
INDEX `fk_employees_spaces1_idx` (`spaces_id` ASC) VISIBLE,
CONSTRAINT `fk_employees_spaces1`
  FOREIGN KEY (`spaces_id`)
    REFERENCES `BookStore`.`spaces` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;

-----
-- Table `BookStore`.`clients`
-----

DROP TABLE IF EXISTS `BookStore`.`clients` ;

CREATE TABLE IF NOT EXISTS `BookStore`.`clients` (
  `id` INT UNSIGNED NOT NULL AUTO_INCREMENT,
  `full_name` VARCHAR(100) NULL,
  `birth_date` DATE NULL,
  `email` VARCHAR(100) NULL,
  `login` VARCHAR(45) NULL,
  `password` VARCHAR(45) NULL,
  `loyalty_card` VARCHAR(45) NULL,
  `discount_status` INT UNSIGNED NULL,
  PRIMARY KEY (`id`))
ENGINE = InnoDB;

-----
-- Table `BookStore`.`orders`
-----

DROP TABLE IF EXISTS `BookStore`.`orders` ;

CREATE TABLE IF NOT EXISTS `BookStore`.`orders` (
  `id` INT UNSIGNED NOT NULL AUTO_INCREMENT,
  `clients_id` INT UNSIGNED NOT NULL,
  `employees_id` INT UNSIGNED NOT NULL,
  `order_date` DATETIME NULL DEFAULT NOW(),
  `delivery_address` VARCHAR(100) NULL,
  `delivery_date` DATE NULL,
  `delivery_cost` DECIMAL(6,2) NULL,
  `order_cost` DECIMAL(7,2) NULL,
  `delivery_status` ENUM("accepted", "paid", "in_delivery", "delivered")
  NULL,
  `total_count` INT NULL,
  `total_price` DECIMAL(8,2) NULL,
  PRIMARY KEY (`id`),
  INDEX `fk_orders_clients1_idx` (`clients_id` ASC) VISIBLE,
  INDEX `fk_orders_employees1_idx` (`employees_id` ASC) VISIBLE,
  CONSTRAINT `fk_orders_clients1`
    FOREIGN KEY (`clients_id`)
      REFERENCES `BookStore`.`clients` (`id`)
      ON DELETE NO ACTION
      ON UPDATE NO ACTION,
  CONSTRAINT `fk_orders_employees1`
    FOREIGN KEY (`employees_id`)
      REFERENCES `BookStore`.`employees` (`id`)
      ON DELETE NO ACTION
      ON UPDATE NO ACTION)
ENGINE = InnoDB;

```

```

-----
-- Table `BookStore`.`cheque`
-----
DROP TABLE IF EXISTS `BookStore`.`cheque` ;

CREATE TABLE IF NOT EXISTS `BookStore`.`cheque` (
  `id` INT UNSIGNED NOT NULL AUTO_INCREMENT,
  `purshase_time` DATE NULL,
  `orders_id` INT UNSIGNED NOT NULL,
  `tax` TINYINT(3) NULL DEFAULT 10,
  PRIMARY KEY (`id`),
  INDEX `fk_cheque_orders_idx` (`orders_id` ASC) VISIBLE,
  CONSTRAINT `fk_cheque_orders`
    FOREIGN KEY (`orders_id`)
      REFERENCES `BookStore`.`orders` (`id`)
      ON DELETE NO ACTION
      ON UPDATE NO ACTION)
ENGINE = InnoDB;

-----
-- Table `BookStore`.`requests`
-----
DROP TABLE IF EXISTS `BookStore`.`requests` ;

CREATE TABLE IF NOT EXISTS `BookStore`.`requests` (
  `id` INT UNSIGNED NOT NULL AUTO_INCREMENT,
  `spaces_id` INT UNSIGNED NOT NULL,
  `request_date` DATE NULL,
  `complete_date` DATE NULL,
  PRIMARY KEY (`id`),
  INDEX `fk_requests_spaces1_idx` (`spaces_id` ASC) VISIBLE,
  CONSTRAINT `fk_requests_spaces1`
    FOREIGN KEY (`spaces_id`)
      REFERENCES `BookStore`.`spaces` (`id`)
      ON DELETE NO ACTION
      ON UPDATE NO ACTION)
ENGINE = InnoDB;

-----
-- Table `BookStore`.`supplies`
-----
DROP TABLE IF EXISTS `BookStore`.`supplies` ;

CREATE TABLE IF NOT EXISTS `BookStore`.`supplies` (
  `id` INT UNSIGNED NOT NULL AUTO_INCREMENT,
  `supplier` VARCHAR(100) NULL,
  `spaces_id` INT UNSIGNED NOT NULL,
  `cost` DECIMAL(10,2) NULL,
  `date` DATE NULL,
  PRIMARY KEY (`id`),
  INDEX `fk_supplies_spaces1_idx` (`spaces_id` ASC) VISIBLE,
  CONSTRAINT `fk_supplies_spaces1`
    FOREIGN KEY (`spaces_id`)
      REFERENCES `BookStore`.`spaces` (`id`)
      ON DELETE NO ACTION
      ON UPDATE NO ACTION)
ENGINE = InnoDB;

```

```

-- -----
-- Table `BookStore`.`books_has_orders`
-- -----
DROP TABLE IF EXISTS `BookStore`.`books_has_orders` ;

CREATE TABLE IF NOT EXISTS `BookStore`.`books_has_orders` (
  `id` INT UNSIGNED NOT NULL AUTO_INCREMENT,
  `books_ISBN` VARCHAR(13) NOT NULL,
  `orders_id` INT UNSIGNED NOT NULL,
  `book_count` INT UNSIGNED NULL,
  `position_cost` DECIMAL(7,2) NULL,
  INDEX `fk_books_has_orders_orders1_idx` (`orders_id` ASC) VISIBLE,
  INDEX `fk_books_has_orders_books1_idx` (`books_ISBN` ASC) VISIBLE,
  PRIMARY KEY (`id`),
  CONSTRAINT `fk_books_has_orders_books1`
    FOREIGN KEY (`books_ISBN`)
      REFERENCES `BookStore`.`books` (`ISBN`)
      ON DELETE NO ACTION
      ON UPDATE NO ACTION,
  CONSTRAINT `fk_books_has_orders_orders1`
    FOREIGN KEY (`orders_id`)
      REFERENCES `BookStore`.`orders` (`id`)
      ON DELETE NO ACTION
      ON UPDATE NO ACTION)
ENGINE = InnoDB;

-- -----
-- Table `BookStore`.`employees_has_requests`
-- -----
DROP TABLE IF EXISTS `BookStore`.`employees_has_requests` ;

CREATE TABLE IF NOT EXISTS `BookStore`.`employees_has_requests` (
  `id` INT UNSIGNED NOT NULL AUTO_INCREMENT,
  `employees_id` INT UNSIGNED NOT NULL,
  `requests_id` INT UNSIGNED NOT NULL,
  INDEX `fk_employees_has_requests_requests1_idx` (`requests_id` ASC)
  VISIBLE,
  INDEX `fk_employees_has_requests_employees1_idx` (`employees_id` ASC)
  VISIBLE,
  PRIMARY KEY (`id`),
  CONSTRAINT `fk_employees_has_requests_employees1`
    FOREIGN KEY (`employees_id`)
      REFERENCES `BookStore`.`employees` (`id`)
      ON DELETE NO ACTION
      ON UPDATE NO ACTION,
  CONSTRAINT `fk_employees_has_requests_requests1`
    FOREIGN KEY (`requests_id`)
      REFERENCES `BookStore`.`requests` (`id`)
      ON DELETE NO ACTION
      ON UPDATE NO ACTION)
ENGINE = InnoDB;

-- -----
-- Table `BookStore`.`spaces_has_books`
-- -----
DROP TABLE IF EXISTS `BookStore`.`spaces_has_books` ;

CREATE TABLE IF NOT EXISTS `BookStore`.`spaces_has_books` (
  `id` INT UNSIGNED NOT NULL AUTO_INCREMENT,
  `spaces_id` INT UNSIGNED NOT NULL,
  `books_ISBN` VARCHAR(13) NOT NULL,

```

```

`count_in_space` INT NULL,
INDEX `fk_spaces_has_books_books1_idx` (`books_ISBN` ASC) VISIBLE,
INDEX `fk_spaces_has_books_spaces1_idx` (`spaces_id` ASC) VISIBLE,
PRIMARY KEY (`id`),
CONSTRAINT `fk_spaces_has_books_spaces1`
  FOREIGN KEY (`spaces_id`)
    REFERENCES `BookStore`.`spaces` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
CONSTRAINT `fk_spaces_has_books_books1`
  FOREIGN KEY (`books_ISBN`)
    REFERENCES `BookStore`.`books` (`ISBN`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;

SET SQL_MODE=@OLD SQL_MODE;
SET FOREIGN_KEY_CHECKS=@OLD FOREIGN_KEY_CHECKS;
SET UNIQUE_CHECKS=@OLD UNIQUE_CHECKS;

```

Приложение 2. Программный код заполнения БД

```

use bookstore;
insert into `books` values
('9785170905430', 'Чуб земли', 'Макс Фрай', 'Фентези', '2018', '402.00', '150'),
('9785170924554', 'Неуловимый Хабба Хэн', 'Макс Фрай', 'Фентези', '2018',
'450.00', '200'),
('9785170923212', 'Властелин Морморы', 'Макс Фрай', 'Фентези', '2018',
'405.00', '150'),
('9785170924561', 'Ворона на мосту', 'Макс Фрай', 'Фентези', '2018',
'390.00', '145'),
('9785170786961', 'Властелин колец', 'Джон Р.Р.Толкин', 'Фентези', '2013',
'748.50', '85'),
('9785170738472', 'Полная история Средиземья', 'Джон
Р.Р.Толкин', 'Фентези', '2013', '986.00', '50'),
('9785941422306', 'Личный ребрендинг', 'Т.Мужицкая', 'Саморазвитие', '2013',
'250.00', '102'),
('9785769548789', 'Психология общения', 'В.А.Горянина', 'Психология', '2004',
'358.50', '77'),
('9785498075808', 'Психология лжи', 'Пол Экман', 'Психология', '2010',
'500.00', '325'),
('9785988571230', 'Рунические практики', 'Светлана
Некрасова', 'Эзотерика', '2008', '215.00', '60'),
('9785942787468', 'Молот ведьм', 'Яков Шпренгер', 'Эзотерика', '2005',
'268.00', '50'),
('9785928707088', 'Сонник', 'Г.Х.Миллер', 'Эзотерика', '2004', '160.00', '25');

INSERT INTO `clients` VALUES ('1', 'Марселина Хоуп', '2005-03-
19', 'marcelina.hoppe@example.net', 'dubuque.mossie', 'e5a043773a67fb8c0fcdc5e29
5d687509e0c7d1c', '4934539883228389', '0'),
('2', 'Иосиф Блумач', '1985-03-
30', 'friesen.toni@example.com', 'cole.shaun', '56f9f99f963fc0f0dc67180188dcb505
09c6e9f2', '4024007111132', '15'),
('3', 'Элеонора Федотова', '1972-09-
10', 'elenora49@example.org', 'grayson56', 'f40e6b02bf051d5a026dae5422e6f2a324ca
e7cb', '4916041015092394', '5'),
('4', 'Егор Шастун', '1994-10-
01', 'rocky65@example.com', 'rebeka03', '762b2918ab17b939d32f6290d09326c31f3d36a
0', '5416317396757656', '0'),
('5', 'Николай Джаллибов', '1987-03-
08', 'nikolaus.jordon@example.com', 'ivah18', '8f155bb9178d199ed655b253e783d89ce
52c49ba', '5405493198341772', '25'),

```

```
(
'6','Анна-Мария Грант','1970-04-12',
'amalia.grant@example.net','annamarie.ankunding',
'dabafe900dc0b5226bc3c1263b9eb77bd959eaf5',
'377007879790721','10'),
('7','Ольга Байцова','1974-05-09',
'obie88@example.com','maurine.wehner',
'0de6e01842387411233f8785ea6dae0b1d8d80d0',
'340815543579308','0'),
('8','Елисей Ховенцов','1988-07-11',
'ehowe@example.com','lgreen',
'60eeac1807c35a33c2d1e47477af507c17359469',
'5166295782019827','10'),
('9','Инесса Ким','2006-10-13',
'bins.kurt@example.org','doug.senger',
'bceb0d1f4eca24c4f2c61d046478a62e1f31b44f',
'4716478251497228','15'),
('10','Григорий Вутц','2002-12-06',
'lueilwitz.jack@example.com','sonya.stanton',
'f4f04f216d6d7548e9943171372639aa76085255',
'5275783236827135','0'),
('11','Матвей Брейтенберг','2003-01-25',
'ruben27@example.org','vbreitenberg',
'43e6b63325711621136c2f653c276c2199240e74',
'4556679341670886','5'),
('12','Мария Шерман','1994-01-30',
'hil111.lucienne@example.com','sherman46',
'e359de3ce69323e1d0f958c67fe56245e6bb6bc3',
'4916606818295805','5'),
('13','Герман Фоменко','2000-07-31',
'herminia.kreiger@example.com','fkirlin',
'a1586c7d02475b62e0a375ba11b73360f08d6e27',
'5484951236587168','0'),
('14','Анна Земляк','1995-01-09',
'vhickle@example.org','shanna.zemlak',
'beaa629a8938d58136412083060d83b87c410076',
'5534776066385728','10'),
('15','Михаил Подобедов','2004-10-03',
'michele.mosciski@example.net','gbeer',
'adf8c93f8d73ad9b7234a6847b304f871265de57',
'342564896224129','0');
```

```
INSERT INTO `spaces` VALUES ('1','Хорошевское Шоссе, дом 62','warehouse'),
('2','Студенческая Ул., дом 23/A','store'),
('3','Вадковский Пер., дом 20','store'),
('4','Горького, дом 58','store'),
('5','Пограничная, дом 26','store');
```

```
INSERT INTO `employees` VALUES ('1','Екатерина Волкова','1988-11-05',
'бухгалтер','9549665964','2','27000.00'),
('2','Даниил Коваленко','2001-06-27','сотрудник склада',
'9656062935','1','22500.00'),
('3','Максим Шульц','1983-05-23','директор','9853452022','3',
'60000.00'),
('4','Эвелина Скворцова','1993-12-30','администратор',
'9691120466','2','35700.00'),
('5','Никита Алексеев','1991-11-01','продавец','9776487598',
'3','32500.00'),
('6','Галина Зорина','1970-04-21','бухгалтер','9766636193',
'5','40000.00'),
('7','Виктор Попов','2000-12-05','продавец','9672794202',
'4','22000.07'),
('8','Константин Южаков','1997-11-16','менеджер склада',
'9720764558','1','31800.00'),
('9','Марина Михайлова','1988-08-21','администратор',
'9544270713','4','42000.40'),
('10','Наталья Бычкова','1982-10-06','бухгалтер',
'9493873883','1','44500.00'),
('11','Кристина Ермилова','1988-06-30','администратор',
'9735451-42','3','35000.00'),
('12','Николай Чудин','2000-10-17','продавец','9778741593',
'3','23500.00'),
('13','Нелли Акопян','1972-10-29','продавец','9241849699',
'2','20450.00'),
('14','Михаил Виноградов','2004-07-04','администратор',
'9258467952','5','38600.00'),
('15','Светлана Новицкая','1972-12-07','продавец',
'9771136485','5','17850.00');
```

```
INSERT INTO `orders` VALUES ('1','1','7','2019-12-07 13:28:27',
'Горького, дом
```

```

28','2019-12-06','0.00','99999.99','delivered','4','1647.00'),
('2','2','8','2019-09-07 21:50:35','Молостовых, 5','2019-09-
10','300.00','2040.00','delivered','2','858.50'),
('3','5','13','2019-09-07 17:50:35','Студенческая Ул., дом 23/А','2019-09-
07','0.00','1972.00','delivered','2','1972.00'),
('4','3','8','2020-05-27 10:17:21','26-Бакинских Комиссаров,15','2020-06-
04','300.00','268.00','in_delivery','1','268.00'),
('5','4','8','2020-06-01 17:35:23','Новикова, 34','2020-06-
09','0.00','2010.00','accepted','5','2010.00'),
('6','5','8','2020-05-30 22:04:09','Станиславского, 9','2020-06-
10','300.00','570.00','paid','3','570.00');

INSERT INTO `cheque` VALUES ('1','2019-12-07 13:28:27','1','10'),
('2','2019-09-10 12:55:47','2','10'),
('3','2019-09-07 17:50:35','3','10'),
('4','2020-05-30 22:04:09','6','10');

INSERT INTO `requests` VALUES ('1','2','2019-07-09','2019-07-13'),
('2','3','2019-08-27','2019-08-31'),
('3','4','2019-10-01','2019-10-06'),
('4','5','2019-10-04','2019-10-09'),
('5','3','2020-01-10','2020-01-14'),
('6','5','2020-02-10','2020-02-15'),
('7','4','2020-03-14','2020-03-23'),
('8','2','2020-03-30','2020-04-05'),
('9','3','2020-05-29','2003-06-01'),
('10','5','2020-05-30','2020-06-04');

INSERT INTO `supplies` VALUES ('1','Амфора','1','50000.00','2019-06-28'),
('2','Книги для всех','2','35000.00','2019-08-02'),
('3','АСТ','3','155325.00','2019-09-16'),
('4','Лабиринт пресс','4','49765.00','2019-12-23'),
('5','Амфора','5','65700.00','2020-01-12'),
('6','Фантастика','1','251025.75','2020-02-02'),
('7','АСТ','2','264003.00','2020-04-30'),
('8','Росмэн','3','148500.00','2020-05-16'),
('9','АСТ','4','205678.30','2020-05-23');

INSERT INTO `spaces_has_books` VALUES ('1','1','9785170905430','150'),
('2','2','9785170905430','0'),
('3','3','9785170905430','0'),
('4','4','9785170905430','0'),
('5','5','9785170905430','0'),
('6','1','9785170924554','200'),
('7','2','9785170924554','0'),
('8','3','9785170924554','0'),
('9','4','9785170924554','0'),
('10','5','9785170924554','0'),
('11','1','9785170923212','150'),
('12','2','9785170923212','0'),
('13','3','9785170923212','0'),
('14','4','9785170923212','0'),
('15','5','9785170923212','0'),
('16','1','9785170924561','145'),
('17','2','9785170924561','0'),
('18','3','9785170924561','0'),
('19','4','9785170924561','0'),
('20','5','9785170924561','0'),
('21','1','9785170786961','85'),
('22','2','9785170786961','0'),
('23','3','9785170786961','0'),
('24','4','9785170786961','0'),
('25','5','9785170786961','0'),

```



```

('26','1','9785170738472','50'),
('27','2','9785170738472','0'),
('28','3','9785170738472','0'),
('29','4','9785170738472','0'),
('30','5','9785170738472','0'),
('32','1','9785941422306','102'),
('33','2','9785941422306','0'),
('34','3','9785941422306','0'),
('35','4','9785941422306','0'),
('36','5','9785941422306','0'),
('37','1','9785769548789','77'),
('38','2','9785769548789','0'),
('39','3','9785769548789','0'),
('40','4','9785769548789','0'),
('41','5','9785769548789','0'),
('42','1','9785498075808','325'),
('43','2','9785498075808','0'),
('44','3','9785498075808','0'),
('45','4','9785498075808','0'),
('46','5','9785498075808','0'),
('47','1','9785988571230','60'),
('48','2','9785988571230','0'),
('49','3','9785988571230','0'),
('50','4','9785988571230','0'),
('51','5','9785988571230','0');

INSERT INTO `employees_has_requests` VALUES ('1','4','1'),
('2','11','2'),
('3','9','3'),
('4','14','4'),
('5','4','5'),
('6','11','6'),
('7','9','7'),
('8','4','8'),
('9','11','9'),
('10','14','10');

INSERT INTO `books_has_orders` VALUES ('1','9785170905430','1','1','402.00'),
('2','9785170924554','1','1','450.00'),
('3','9785170923212','1','1','405.00'),
('4','9785170924561','1','1','390.00'),
('5','9785769548789','2','1','358.50'),
('6','9785498075808','2','1','500.00'),
('7','9785170738472','3','2','1972.00'),
('8','9785942787468','4','1','268.00'),
('9','9785170905430','5','5','2010.00'),
('10','9785928707088','6','2','320.00'),
('11','9785941422306','6','1','250.00');

```

Приложение 3. Программный код представлений

```

use bookstore;

create or replace view not_enough_books as
  select spaces_id,ISBN,title,count_in_space from spaces_has_books,books
  where count_in_space <10 and books_ISBN=ISBN;

create or replace view not_paid_orders as
  select orders.id from orders
  left join cheque on orders.id=orders_id
  where orders_id is null;

create or replace view report_month as

```

```

select books_ISBN,SUM(book_count) as count from books_has_orders
where orders_id in (select orders.id from orders,cheque
where orders.id=orders_id and orders_id is not null and
month(purshase_time)=month(now()))
group by books_ISBN;

create or replace view all_orders as
select clients.id,full name,sum(total price) as total sum from
clients,orders
where clients.id = clients_id
group by full_name;

create or replace view short_order_info as
select orders_id,books_ISBN,book_count,employees.spaces_id from orders
left join books_has_orders on orders.id = books_has_orders.orders_id
inner join employees on employees_id = employees.id;

```

Приложение 4. Программный код триггеров

```

use bookstore;

drop trigger decrease_count;
create trigger decrease_count before insert on cheque for each row begin
declare finished int default 0;
declare curr_count int;
declare curr_ISBN varchar(13);
declare curr_space int;
declare curr_order int;
declare row_cursor cursor for
select orders_id,books_ISBN, spaces_id, book_count
from short_order_info where short_order_info.orders_id =
new.orders_id;
DECLARE continue HANDLER FOR NOT FOUND SET finished = 1;
OPEN row_cursor;
while finished = 0 do
FETCH row_cursor into curr_order,curr_ISBN,curr_space,curr_count;
update spaces_has_books set count_in_space = count_in_space -
curr_count
where spaces_has_books.spaces_id = curr_space and
books_ISBN = curr_ISBN and
curr_order = new.orders_id;
end while;

end;

drop trigger update_total_count;
create trigger update_total_count before update on spaces_has_books for each
row
if new.count_in_space is not null then
update books set count = count - OLD.count_in_space +
new.count_in_space
where new.books_ISBN = ISBN;
end if;

drop trigger discount_status;
create trigger discount_status after insert on cheque for each row begin
declare client_sum decimal(10,2);
declare curr_id int;
select all_orders.total_sum, all_orders.id into client_sum, curr_id from
all_orders where id = (select clients_id from orders where orders.id =
new.orders_id);
if client_sum < 5000
then update clients set discount_status = 0 where clients.id =

```

```

curr_id;
    elseif client_sum = 5000 or client_sum > 5000 and client_sum < 10000
    then update clients set discount_status = 5 where clients.id =
curr_id;
    elseif client_sum = 5000 or client_sum > 10000 and client_sum < 20000
    then update clients set discount_status = 10 where clients.id =
curr_id;
    elseif client_sum = 10000 or client_sum > 20000 and client_sum <
50000
    then update clients set discount_status = 15 where clients.id =
curr_id;
    else update clients set discount_status = 25 where clients.id =
curr_id;
    end if;
end;

```

Приложение 5. Программный код создания пользователей и задания их привилегий

```

CREATE USER 'customer'@'localhost' IDENTIFIED BY '123';
CREATE USER 'manager'@'localhost' IDENTIFIED BY '123';

GRANT SELECT ON BookStore.orders TO 'customer'@'localhost';
GRANT SELECT ON BookStore.books_has_orders TO 'customer'@'localhost';
GRANT SELECT ON BookStore.cheque TO 'customer'@'localhost';
GRANT SELECT ON BookStore.books TO 'customer'@'localhost';

GRANT UPDATE ON BookStore.clients to 'customer'@'localhost';

GRANT INSERT ON BookStore.orders TO 'customer'@'localhost';

GRANT SELECT ON BookStore.orders TO 'manager'@'localhost';
GRANT SELECT ON BookStore.books_has_orders TO 'manager'@'localhost';
GRANT SELECT ON BookStore.cheque TO 'manager'@'localhost';
GRANT SELECT ON BookStore.books TO 'manager'@'localhost';
GRANT SELECT ON BookStore.supplies TO 'manager'@'localhost';

GRANT UPDATE ON BookStore.orders TO 'manager'@'localhost';

```