

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение высшего образования

**«Дальневосточный федеральный университет»**

(ДВФУ)

**ИНСТИТУТ МАТЕМАТИКИ И КОМПЬЮТЕРНЫХ ТЕХНОЛОГИЙ**

**(ШКОЛА)**

**Департамент программной инженерии и искусственного интеллекта**

БАРБАЯНОВ МАКСИМ АНДРЕЕВИЧ

РАЗРАБОТКА БАЗЫ ДАННЫХ «ИЗДАТЕЛЬСТВО»

**КУРСОВАЯ РАБОТА**

по дисциплине «Технология разработки баз данных»

по образовательной программе подготовки бакалавров по направлению

09.03.04 «Программная инженерия», профиль «Программная инженерия»

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | | | | | | |  |  | Студент группы Б9121-09.03.04 | | | | | | | |
|  |  |  | | | | Барбаянов М.А. | | | |
|  | | | | | | |  |  | (подпись) | | | |  | | | |
|  | | | | | | | | | | | | | | | | |
|  | | | | | | |  |  | Руководитель | | | | | | | |
|  | | | | | | |  |  | ст. преподаватель ДПИиИИ | | | | | | | |
|  | | | | | | |  |  | ученая степень, должность | | | | |  | Крестникова О.А. | |
|  | | | | | | |  |  | (подпись) | | | | |  | (ФИО) | |
|  | | | | | | | | | | | | | | | | |
| Регистрационный № | | | | | | |  |  | Защищен с оценкой | | | | | | | |
|  | | | | | | |  |  |  | | | | | | | |
|  | | | |  |  | |  |  | « |  | » |  |  | | | 2024 г. |
| (подпись) | | | |  | (ФИО) | |  |  |  | | | | | | | |
| « |  | » |  | | | 2023 г. |  |  |  | | | | | | | |

г. Владивосток

2023

Содержание

[ВВЕДЕНИЕ 3](#_Toc162557391)

[1. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ 6](#_Toc162557392)

[1.1. Инфологическое проектирование 7](#_Toc162557393)

[1.2. Концептуальная модель данных 8](#_Toc162557394)

[1.3. Даталогическое проектирование 9](#_Toc162557395)

[1.3.1. Выбор СУБД 9](#_Toc162557396)

[1.3.2. Сравнение технологии No-SQL и реляционных баз данных 10](#_Toc162557397)

[1.3.3. Обзор СУБД 12](#_Toc162557398)

[1.3.4. Выбор нотации для проектирования структуры базы данных 19](#_Toc162557406)

[1.3.5. Обзор средств для проектирования баз данных 20](#_Toc162557407)

[1.3.6. Выбор CASE-средства для проектирования БД 23](#_Toc162557411)

[2. ПРОЕКТИРОВАНИЕ ТАБЛИЦ БАЗЫ ДАННЫХ 25](#_Toc162557412)

[2.1. Типы полей 25](#_Toc162557413)

[2.2. Первичные, внешние и альтернативные ключи 29](#_Toc162557414)

[2.3. Ключи таблиц 30](#_Toc162557415)

[2.4. Определение правил ограничения ввода 34](#_Toc162557416)

[2.5. Организация ввода данных 37](#_Toc162557417)

[ЗАКЛЮЧЕНИЕ 56](#_Toc162557419)

[СПИСОК ЛИТЕРАТУРЫ 58](#_Toc162557420)

# ВВЕДЕНИЕ

Современный мир информации и знаний требует эффективного управления и хранения данных для обеспечения качественной работы в различных областях. Одним из таких ключевых секторов является издательское дело, где базы данных играют важную роль в управлении издательским процессом, от сбора и обработки информации до распространения и продажи книг, журналов и других публикаций. В данной курсовой работе мы рассмотрим базу данных "Издательство", которая представляет собой систему управления данными, предназначенную для организации и управления информацией, связанной с издательской деятельностью. В рамках данной работы мы проведем анализ структуры и функциональности базы данных "Издательство", исследуем ее основные компоненты и возможности, а также рассмотрим примеры использования и практическое применение в сфере издательского бизнеса. Курсовой проект выполнен с целью практического освоения основных приемов и правил методологии информационного моделирования IDEF1X и структурированного языка запросов SQL. В качестве предметной области разрабатываемой базы данных (БД) выбрана система «Издание». Предложенный в настоящей курсовой работе проект направлен на достижение указанных целей.

Концептуальная модель данных представлена в виде IDEF1X-диаграммы данных, показывающая сущности предметной области и выявляющая обусловленные правила издательств. Диаграмма сопровождается глоссарием, содержащим формальные определения имен всех сущностей и хранимых элементов данных. Практическое значение: разработанная БД позволяет автоматизировать контроль над изданием или контроль любого издательства. Актуальность курсовой работы объясняется тем, что в настоящее время, специалистам в IT-сфере необходимо эффективно распределять и систематизировать большой поток информации. Для этого специалист должен уметь разрабатывать и внедрять распределенные системы обработки данных. База данных разработана в программной среде MYSQL, с использованием языка программирования SQL. Результатом курсовой работы должна стать готовая БД, удовлетворяющая всем требованиям технического задания.

Целью данной курсовой работы является рассмотрение базы данных "Издательство" как инструмента для эффективного управления информацией в издательской сфере, выявление ее преимуществ и возможностей, а также оценка ее вклада в повышение эффективности работы издательских предприятий. Объектом исследования является база данных "Издательство", которая содержит информацию о книгах, авторах, издательствах, жанрах, продажах и других связанных с издательской деятельностью данных. Исследование этой базы данных предполагает анализ структуры и содержания данных, выявление основных характеристик и взаимосвязей между различными сущностями, а также разработку методов для оптимизации управления издательскими процессами, повышения эффективности маркетинговых стратегий и улучшения качества выпускаемой продукции.

Предметом исследования является изучение процессов управления издательской деятельностью с использованием базы данных "Издательство". Это включает в себя анализ структуры данных, выявление основных характеристик и закономерностей в работе издательства, исследование спроса на различные категории книг, анализ эффективности маркетинговых стратегий и определение факторов, влияющих на успешность продаж. Кроме того, предметом исследования может быть разработка и оптимизация методов управления данными, повышение качества и актуальности выпускаемой литературы, а также адаптация издательской стратегии к изменяющимся потребностям рынка.

Целью данной курсовой работы является изучение и анализ базы данных "Издательство" с целью оптимизации управленческих процессов и повышения эффективности издательской деятельности. Для достижения поставленной цели необходимо решить следующие задачи:

1. Изучить структуру базы данных "Издательство": провести анализ ее таблиц, ключей, и связей между ними.

2. Проанализировать основные характеристики и параметры данных: выявить объем информации, типы данных, наличие и обработку пропущенных значений.

3. Исследовать динамику и характеристики спроса на книги: определить наиболее популярные категории, авторов, и жанры литературы.

4. Провести анализ эффективности маркетинговых стратегий: оценить влияние различных маркетинговых кампаний на продажи книг.

5. Определить факторы, влияющие на успешность продаж: проанализировать связь между характеристиками книг (например, жанр, автор, обложка) и их коммерческим успехом.

6. Разработать рекомендации по оптимизации управления данными: предложить методы улучшения качества данных, оптимизации процессов ведения базы данных, и улучшения взаимодействия с конечными пользователями.

Путем решения указанных задач планируется достичь цели курсовой работы – оптимизации управленческих процессов и повышения эффективности издательской деятельности на основе анализа базы данных "Издательство".

# 1. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

Анализ предметной области базы данных "Издательство" является важным этапом для понимания ее структуры, содержания и целей использования. Предметная область издательства включает в себя широкий спектр аспектов, связанных с производством, распространением и продажей книг и другой литературы. Ниже представлены основные аспекты анализа предметной области:

1. Производство книг: включает в себя все этапы создания книги, начиная от разработки концепции и написания текста до верстки, дизайна обложки и печати.

2. Маркетинг и продажи: анализируется спрос на книги, эффективность маркетинговых кампаний, стратегии ценообразования и распространения книг.

3. Управление ассортиментом: важно определить ассортимент книг, который предлагается читателям, а также понять, какие категории книг являются наиболее популярными.

4. Анализ конкурентов: изучаются другие издательства и их стратегии, чтобы определить сильные и слабые стороны, а также возможности и угрозы для собственного издательства.

5. Управление данными: проводится анализ качества данных, структуры и связей в базе данных "Издательство", а также определяются потребности в дополнительной обработке и оптимизации данных.

Анализ предметной области позволяет глубже понять основные аспекты работы издательства, выявить потенциальные проблемы и найти пути их решения, а также определить возможности для улучшения бизнес-процессов и повышения конкурентоспособности издательства.

## 1.1. Инфологическое проектирование

Инфологическое проектирование базы данных "Издательство" направлено на определение сущностей и их атрибутов, а также на установление связей между сущностями для правильной организации данных. Ниже представлены основные шаги и результаты инфологического проектирования:

1. Определение сущностей: Идентификация основных объектов, которые будут представлены в базе данных. Это могут быть, например, книги, авторы, издательства, заказы и клиенты.

2. Атрибуты сущностей: Определение характеристик каждой сущности, которые будут храниться в базе данных. Например, у книги могут быть атрибуты: название, автор, жанр, год издания и ISBN.

3. Определение связей: Установление связей между сущностями для представления их взаимосвязей и зависимостей. Например, книга может быть написана одним или несколькими авторами, издана определенным издательством и быть частью заказа от конкретного клиента.

4. Нормализация данных: Проверка и оптимизация структуры данных для предотвращения избыточности и повышения эффективности хранения и обработки информации. Это включает в себя разделение данных на отдельные таблицы и устранение аномалий связей.

5. Создание диаграммы сущность-связь (ER-диаграмма): Визуализация структуры базы данных с помощью диаграммы, которая отображает сущности, атрибуты и связи между ними. Это позволяет легче понять структуру базы данных и обсудить ее с заинтересованными сторонами.

6. Проверка и уточнение: Проверка инфологической модели на соответствие требованиям и корректировка при необходимости. Этот этап позволяет обнаружить и исправить ошибки или упущения в проектировании базы данных.

Инфологическое проектирование является важным этапом разработки базы данных, поскольку от его качества зависит дальнейшая эффективность и удобство использования базы данных в реальной работе издательства.

## 1.2. Концептуальная модель данных

Концептуальная модель данных представляет собой абстрактное представление структуры и связей данных в информационной системе без привязки к конкретным техническим аспектам или средствам хранения. В контексте базы данных "Издательство" концептуальная модель данных помогает описать основные сущности и их взаимосвязи, учитывая требования бизнеса и потребности пользователей. Концептуальная модель представляет собой описание основных сущностей (таблиц) и связей между ними без учета принятой модели БД и синтаксиса целевой БД.На рисунке 1.1 приведен блок физической информационной модели «Издание». Для ее построения использовалась программа Erwin 4.0.

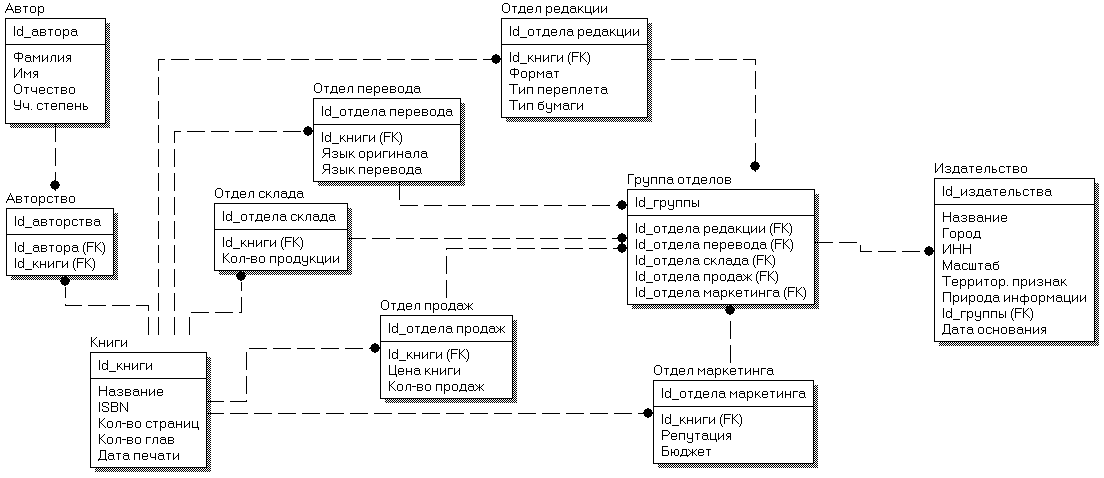


Рис. 1.1. Концептуальная модель «Издание»

## 1.3. Даталогическое проектирование

### 1.3.1. Выбор СУБД

При выборе СУБД (системы управления базами данных) для проекта "Издательство" следует учитывать несколько ключевых критериев, включая:

1. Тип данных и характеристики проекта: определите, какие типы данных будут храниться в базе данных (текст, числа, изображения и т.д.) и какие операции с данными будут часто выполняться (чтение, запись, обновление, удаление).

2. Масштаб проекта: оцените ожидаемый объем данных, частоту запросов и количество пользователей базы данных. Это поможет определить требования к производительности и масштабируемости СУБД.

3. Функциональные требования: убедитесь, что выбранная СУБД поддерживает необходимый набор функций и возможностей, таких как транзакции, индексирование, безопасность данных и резервное копирование.

4. Совместимость и интеграция: Проверьте, как хорошо выбранная СУБД интегрируется с другими технологиями, используемыми в проекте, такими как языки программирования, веб-фреймворки и инструменты администрирования.

5. Стоимость и лицензирование: оцените стоимость использования СУБД, включая лицензионные и поддерживающие услуги, и убедитесь, что они соответствуют бюджету проекта.

6. Сообщество и поддержка: Исследуйте активность сообщества пользователей выбранной СУБД, доступность документации, обучающих материалов и уровень поддержки со стороны разработчиков.

7. Производительность и надежность: Проведите тестирование производительности выбранной СУБД и оцените её способность обрабатывать нагрузку в реальном времени, а также её устойчивость к отказам и восстановление после сбоев.

На основе этих критериев можно выбрать наиболее подходящую СУБД для проекта "Издательство", будь то реляционная СУБД (например, PostgreSQL, MySQL, SQLite), NoSQL СУБД (например, MongoDB, Cassandra) или гибридная СУБД (например, MariaDB, Oracle Database).

### 1.3.2. Сравнение технологии No-SQL и реляционных баз данных

При конструировании баз данных [2] в современном мире перед разработчиками встает выбор структуры базы данных, выбор между реляционными и объектными базами данных (No-SQL) в зависимости от поставленных перед ними задачами. Сравнивая объектно-ориентированный и реляционный подходы к БД, можно отметить следующие особенности. В реляционных БД реальные объекты представляются как структуры, состоящие из набора элементарных типов данных. Такое представление имеет понятную интерпретацию – строка в плоской таблице. Реляционные базы данных работают по логической схеме, обеспечивая четыре ключевых условия: атомарность, согласованность, изолированность, надежность. Прежде всего, реляционные СУБД позволяют компаниям сохранить ссылочную целостность данных, предлагает широкий выбор инструментов для обработки и анализа данных, а также доступ к их хранению и резервному копированию, чем не может похвастаться No-SQL [3]. Реляционные СУБД [4] также предоставляют бизнес-пользователям отчеты, с которыми можно работать при помощи привычного языка SQL. Но довольно часто реляционная модель и ее способ описания предметной области в виде набора плоских таблиц не отражают внутренней структуры для многих предметных областей, являются искусственными и становятся совершенно непонятными при увеличении количества таблиц. Синонимом No-SQL стали огромные объемы данных, линейная масштабируемость, кластеры, отказоустойчивость, не реляционность. В No-SQL базах в отличие от реляционных структура данных не регламентирована (или слабо типизирована, если проводить аналогии с языками программирования) – в отдельной строке или документе можно добавить произвольное поле без предварительного декларативного изменения структуры всей таблицы. Таким образом, если появляется необходимость поменять модель данных, то единственное достаточное действие – отразить изменение в коде приложения.

В отличие от реляционной модели, которая сохраняет логическую бизнес-сущность приложения в различные физические таблицы в целях нормализации, NoSQL хранилища оперируют с этими сущностями как с целостными объектами. NoSQL имеют как преимущества, так и недостатки перед реляционными СУБД. Преимущества NoSQL перед реляционными СУБД:

* Широкий выбор типов хранилищ.
* Хорошая горизонтальная масштабируемость.
* Простота администрирования.
* Отсутствие жестко заданной схемы данных.
* Простой API для манипуляции данных.

Недостатки NoSQL перед реляционными СУБД:

* Отсутствие единого стандартного языка запросов.
* Большинство NoSQL баз данных не гарантирую выполнение
* требований ACID [5], что может повлечь потерю данных при отказе оборудования.
* Отсутствие поддержки целостности данных.
* Простой API [6] для манипуляции данных затрудняет выполнение
* сложных выборок.

Перед выпускной квалифицированной работой стоит задача в проектировании прототипа базы данных для ведения учета процесса ввода в

эксплуатацию оборудования. В такой базе данных будут хранится однородные данные, объем которых не будет превышать 1 Тбайт, поэтому для построения необходимой базы данных будет использоваться технология реляционных баз данных. Ниже представлен обзор современных реляционных СУБД, на основании которого будет произведен выбор СУБД для решения поставленных задач.

### 1.3.3. Обзор СУБД

Для организации информационной базы со сравнительно однородным содержимым веб-приложений можно использовать реляционные базы данных. Одна из важных черт баз данных – независимость данных от особенностей прикладных программ, которые их используют, а также возможность создания этих программ в такой форме, что изменение особенностей хранения, логической структуры или значений данных не требует изменения программ их обработки. Другой важной чертой баз данных является возможность изменения физических особенностей хранения данных без изменения их логической структуры. Информацией, хранящейся в базе данных (далее БД), может быть всё что угодно: каталог продукции, информация о клиентах, контент веб-сайта и др. Для обеспечения доступа к информации, хранящейся в базе данных, а также для управления ею, применяют систему управления базами данных (СУБД). СУБД – это комплекс языковых и программных средств, предназначенный для создания, ведения и совместного использования БД многими пользователями. Обычно СУБД различают по используемой модели данных. Так, СУБД, базирующиеся на использовании реляционной модели данных, называют реляционными СУБД. Системы управления базами данных помогают отсортировать информацию, а также связать базы данных между собой, при этом предоставив отчет об изменениях и зарегистрированных событиях. Существует большой выбор СУБД для реализации требований пользователя. Выбор зависит от поставленных перед ней задачах. При выборе СУБД стоит обратить внимание на ее масштабируемость, на ее интегрируемость с другими продуктами, если это требуется, есть ли возможность у выбранной СУБД расти вместе с предприятием, т. к. переход на другую СУБД может стать проблемой, не маловажным фактором при выборе СУБД является оценка предоставляемых ею функций и ее стоимости.

### Oracle 12c

Oracle Database 12c [7] – это мощный программный комплекс, который позволяет создавать приложения любой степени сложности. К тому же это первая СУБД, разработанная специально для сред (публичных или частных)

делегированных вычислений. Ядром этого комплекса стала база данных, которая способна хранить практически неограниченные объемы информации за счет масштабирования. И даже резкое увеличение одновременно работающих с информацией пользователей при достаточных аппаратных ресурсах не приведет к ощутимому снижению производительности.

Oracle 12с – предназначена для облачных сред и может быть размещена на одном или нескольких серверах, это позволяет управлять базами данных, которые содержат миллиарды записей. Некоторые из функций новейшей версии Oracle включают в себя grid framework и использования как физических, так и логических структур. Разделение структур позволяет физическому управлению данными не влиять на доступ к логическим структурам. Кроме того, безопасность в этой версии доведена до высочайшего уровня, потому что каждая транзакция изолирована от других. Достоинством данной СУБД является ее надежность, позволяет полностью использовать все преимущества облачных вычислений, в том числе совместное использование ресурсов, гибкость в управлении. Есть различные редакции СУБД Oracle 12, каждая из которых имеет набор функциональных возможностей для удовлетворения различных потребностей в бизнес-приложениях. Компанией Oracle предлагается бесплатная редакция последнего продукта, имеющую ряд ограничений (11 Гб пользовательских данных, 1 Гб оперативной памяти, 1 ядро процессора, 5 пользователей) – Oracle Database Express Edition. Идеально подходит для крупных организаций, которые работают с огромными базами данных и разнообразными функциями. Недостатком является стоимость продукта, не имеющего значительных ограничений.

### MySQL

MySQL [8] – одна из самых популярных СУБД для веб-приложений. Существуют специальные платные версии, предназначенные для коммерческого использования. В бесплатной версии наибольший упор делается на скорость и надежность, а не на полноту функционала, который может стать и достоинством, и недостатком – в зависимости от области внедрения. Эта СУБД позволяет выбирать различные движки для системы хранения, которые позволяют менять функционал инструмента и выполнять обработку данных, хранящихся в различных типах таблиц. 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. Более того, СУБД MySQL поставляется со специальным типом таблиц EXAMPLE, демонстрирующим принципы создания новых типов таблиц. Благодаря открытой архитектуре и GPL-лицензированию, в СУБД MySQL постоянно появляются новые типы таблиц. Она также имеет простой в использовании интерфейс, и пакетные команды, которые позволяют удобно обрабатывать огромные объемы данных.

MySQL является многопользовательским, многопотоковым сервером, применяющий стандартный язык запросов для баз данных SQL. Достоинством данной СУБД является ее доступность, предлагает много функций, даже в бесплатной версии. MySQL отличается хорошей скоростью работы, надежностью, гибкостью. Работа с ней, как правило, не вызывает больших трудностей. Поддержка сервера MySQL автоматически включается в поставку PHP. Может работать с другими базами данных, включая DB2 и Oracle. Не имеет встроенных решений для ряда простых задач, например такой как, создание инкрементных резервных копий. Отсутствует встроенная поддержка XML или OLAP [9], для бесплатной версии эта поддержка приобретается отдельно (на сегодняшний день стоимость равна 80$). Идеально подходит для: организаций, которым требуется надежный инструмент управления базами данных, но бесплатный.

### Microsoft SQL сервер

Ещё одной из популярных СУБД является программный продукт Microsoft SQL-сервер [10]. Это система управления базами данных, движок которой работает на облачных серверах, а также локальных серверах, причем можно комбинировать типы применяемых серверов одновременно. На сегодняшний день продукт адаптирован для операционной системы Linux. Одной из уникальных особенностей последних версий является temporal data support (временная поддержка данных), которая позволяет отслеживать изменения данных с течением времени. Последняя версия Microsoft SQL-сервер поддерживает dynamic data masking (динамическую маскировку данных), которая гарантирует, что только авторизованные пользователи будут видеть конфиденциальные данные. Продукт очень прост в использовании. Движок предоставляет возможность регулировать и отслеживать уровни производительности, которые помогают снизить использование ресурсов. Можно получить доступ к визуализации на мобильных устройствах. Недостатком Microsoft SQL сервер является ее стоимость, которая оказывается неприемлемой для большей части организаций. Даже при тщательной настройке производительности корпорация SQL Server способна занять все доступные ресурсы. Есть проблема с использованием службы интеграции для импорта файлов. Идеально подходит для: крупных организаций, которые уже используют ряд продуктов Microsoft.

### PostgreSQL

PostgreSQL [11] является одним из нескольких бесплатных популярных вариантов СУБД, часто используется для ведения баз данных веб-сайтов. Это

была одна из первых разработанных систем управления базами данных, поэтому в настоящее время она хорошо развита, и позволяет пользователям управлять как структурированными, так и неструктурированными данными. Может быть использован на большинстве основных платформ, включая Linux. Прекрасно справляется с задачами импорта информации из других типов баз данных с помощью собственного инструментария. Движок БД может быть размещен в ряде сред, в том числе виртуальных, физических и облачных. Самая свежая версия, PostgreSQL 9.5, предлагает обработку больших объемов данных и увеличение числа одновременно работающих пользователей. Безопасность была улучшена благодаря поддержке DBMS\_SESSION. PostgreSQL является масштабируемым и способен обрабатывать терабайты данных. Поддерживает формат json. Существует множество предопределенных функций и имеет ряд готовых интерфейсов. К недостатку можно отнести не полную документацию. Скорость работы может падать во время проведения пакетных операций или выполнения запросов чтения. Идеально подходит для организаций с ограниченным бюджетом, но квалифицированными специалистами, когда требуется возможность выбрать свой интерфейс и использовать json.

### Firebird 2.5

Firebird [12] – кроссплатформенная реляционная система управления базами данных (РСУБД), работающая на macOS, Linux, Microsoft Windows и разнообразных Unix платформах. СУБД Firebird распространяется бесплатно с возможностью использования как для приложений с открытым кодом, так и для коммерческих целей. Удобством данной платформы можно считать наличие только одного файла БД, что упрощает процедуру переноса базы данных на другую аппаратную платформу. База данных легко разворачивается на сервере. СУБД обладает возможностью выполнения резервного копирования не требует остановки работы сервера. Технология копирования предполагает сохранение базы в момент создания копии. Процесс не мешает работе сервера. К минусам относятся отсутствие кэша (не отслеживается история запросов) и замедление работы с увеличением внутренней фрагментации базы.

### Линтер

"Линтер" – российская СУБД, реализующая стандарт SQL:2003 (за исключением не скалярных типов данных и объектно-ориентированных возможностей) и поддерживающая большинство операционных систем, в том

числе семейство Windows, различные версии UNIX, ОС реального времени (включая QNX). К особенностям можно отнести защиту данных: 2 класс защиты данных от несанкционированного доступа и 2 уровень контроля отсутствия не декларированных возможностей. Мандатный контроль доступа к данным на уровне таблиц, столбцов записей и отдельных полей записей. Управление доступом к рабочим станциям и устройствам хранения информации. Контроль доступа к СУБД по расписанию. Управление протоколированием операций над БД (аудит). 18 марта 2016 года по решению Экспертного совета по российскому программному обеспечению при Минкомсвязи России СУБД Линтер включена в единый реестр российских программ для электронных вычислительных машин и баз данных (реестр российского ПО). Имеет утилиты конвертации, работающие через ODBC и ADO.NET [13]. Конвертор из DBF-формата. Конвертор модели данных (из ERwin в Линтер). При высокой динамике изменений наблюдается падение эффективности. Идеально подходит для: отечественных организаций, которые работают с конфиденциальными и персональными данными.

### РЕД База Данных

"РЕД База Данных" – российская СУБД, работает на всех основных платформах и ОС (Windows, Linux, BSD Unix, IBM AIX, HP-UX, Sun Solaris и т. д.). Система модульная. Имеет открытый исходный код. Данная СУБД имеет возможность «горячего» резервного копирования и инкрементного резервного копирования. Сертифицирована ФСТЭК России. Соответствует отечественным требованиям по защите информации. В ней имеются модули сопряжения практически для всех используемых сред разработки [14] (драйверы ODBC, JDBC, C/C++, C#, Java, Delphi, PHP, Python, Perl, VB, и т. д.), результатов тестов этих модулей и гарантия стабильной работы. Возможность работы во «встроенном» в ПО (embedded) локальном режиме в виде библиотеки DLL без отдельной установки и настройки СУБД, в т. ч. поддержка встраивания в виртуальную машину Java. Преимуществом СУБД РЕД является ее высокое быстродействие, сравнимое с лидерами рынка, соответствие отечественным требованиям по защите информации. Идеально подходит для: отечественных организаций (включая оборонные), которые работают с конфиденциальными и персональными данными.

Выбор СУБД будет осуществлен на основе сравнения особо популярных СУБД, поддерживающих язык PHP: MySQL, Oracle, Firebird 2.5 (Таблица 1.1).

Таблица 1.1. Сравнение СУБД

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Наименование СУБД** | **Стоимость** | **Переносимость** | **Поддержка XML** | **Возможность логирования** |
| MySQL | Условно бесплатная,  дополнительные возможности покупаются отдельно | Легкая | Есть в платном модуле (80$) | Есть |
| Oracle | Платная | Есть трудности | Есть | Есть |
| Firebird 2.5 | Бесплатная, есть возможность бесплатного использования для коммерческих целей | Легкая | Есть | Нет |

При рассмотрении характеристик нескольких СУБД выбрана Firebird 2.5. Выбор на эту СУБД пал за бесплатную возможность ее использования в коммерческих проектах. Она легко переносима с одного сервера на другой, имеет возможность резервного копирования без отключения пользователей. На основе Firebird можно построить распределенную базу данных с графической оболочкой на языке PHP. Также Firebird поддерживает xml, что обеспечит автоматическое наполнение базы частью уже имеющихся данных.

### 1.3.4. Выбор нотации для проектирования структуры базы данных

В настоящее время существуют два основных методологических подхода для описания и анализа предметной области и информационных систем:

* Структурный подход, в основу которого положен принцип алгоритмической декомпозиции: структура системы описывается в терминах иерархии ее функций и передачи информации между отдельными функциональными элементами (модулями). Поддерживается методологией системного анализа и проектирования Structure Analysis and Design Technique (SADT) – IDEF0, DFD [15].
* Объектно-ориентированный подход использует объектную декомпозицию: структура системы определяется множеством объектов и связей между ними, а поведение системы описывается в терминах обмена сообщениями между объектами. Поддерживается методологией Rational Unified Process (RUP) [16] и языком моделирования Unified Modeling Language (UML). Структурные методы в первую очередь ориентированы на процессы, поэтому для их описания, а также для документирования потоков данных являются более подходящими по сравнению с объектно-ориентированным UML. Например, диаграмма IDEF3 позволяет четко показать процесс, вследствие которого состояние предмета меняется, а в UML можно отобразить лишь операции и условия перехода из одного состояния предмета в другое. Главный недостаток структурного подхода заключается в следующем: процессы и данные существуют отдельно друг от друга (как в модели деятельности организации, так и в модели программной системы), причем проектирование ведется от процессов к данным. Таким образом, помимо функциональной декомпозиции, существует также структура данных, находящаяся на втором плане. В объектно-ориентированном подходе основная категория объектной модели – класс, он объединяет в себе на элементарном уровне как данные, так и операции, которые над ними выполняются (методы).

Для типизирования предметной области, создания концептуальной и логической модели мною выбран объектно-ориентированный подход с нотацией UML. Диаграммы UML сравнительно просты для формирования и чтения после ознакомления с его синтаксисом, что является преимуществом перед структурным подходом. UML расширяет и позволяет вводить собственные текстовые и графические стереотипы, что способствует расширению словаря UML для создания новых элементов моделирования, получаемых из существующих, но имеющих определенные свойства, которые подходят для конкретной проблемы предметной области. UML имеет широкое распространение и динамично развивается, имеется большое количество источников для его изучения.

В настоящее время наиболее часто используемыми средствами концептуального проектирования объектно-ориентированных систем являются UML-редакторы [17], например, Rational Rose, StarUML, Enterprise Architect, yED. В данной работе рассматриваются и сравниваются программные продукты, не только поддерживающие нотацию UML, но и позволяющие на основе построенных диаграмм выполнять генерацию кода.

### 1.3.5. Обзор средств для проектирования баз данных

Основными средствами, охватывающими обширную область поддержки многочисленных технологий автоматизированного проектирования автоматизированных систем (АС), являются CASE-средства. Они обеспечивают наиболее трудоемкие этапы разработки ИС, высокое качество принимаемых технических решений и подготовку проектной документации. CASE-средства основаны на методах визуального представления информации, что предполагает построение структурных или иных диаграмм в реальном масштабе времени, использование многообразной цветовой палитры, сквозную проверку синтаксических правил. Графические средства моделирования предметной области позволяют разработчикам в наглядном виде изучать существующую АС, перестраивать ее в соответствии с поставленными целями и имеющимися ограничениями. В разряд CASE-средств попадают как относительно дешевые системы для персональных компьютеров с весьма ограниченными возможностями, так и дорогостоящие системы для неоднородных вычислительных платформ и операционных сред. Так, современный рынок программных средств насчитывает около 300 различных CASE-средств, наиболее мощные из которых, так или иначе, используются практически всеми ведущими западными фирмами. Обычно к CASE-средствам относят любое программное средство, автоматизирующее ту или иную совокупность процессов жизненного цикла программного обеспечения. На рынке программного обеспечения (ПО) представлен широкий ряд программных продуктов, с помощью которых возможно автоматизировано проектировать корпоративные информационные системы, что значительно облегчает труд разработчика.

### Rational Rose

Rational Rose представляет собой CASE-средство проектирования и разработки информационных систем и программного обеспечения для управления предприятиями. Его можно применять для анализа и моделирования бизнес-процессов. Первая версия этого продукта была выпущена компанией Rational Software [18]. В дальнейшем Rational Rose был куплен IBM. Принципиальное отличие Rational Rose от других средств заключается в объектно-ориентированном подходе. Графические модели, создаваемые с помощью этого средства, основаны на объектно-ориентированных принципах и языке UML (Unified Modeling Language). Инструменты моделирования Rational Rose позволяют разработчикам создавать целостную архитектуру процессов предприятия, сохраняя все взаимосвязи и управляющие воздействия между различными уровнями иерархии. Данное CASE-средство не является кроссплатформенным продуктом, поддерживается только ОС Windows. Продукт является платным, стоимость лицензии от 70 000 рублей. Rational Rose остается довольно популярным CASE-пакетом моделирования бизнес-процессов. По данному программному пакету написано достаточно много литературы и учебных материалов. Но не смотря на все это Rational Rose сложен для изучения в связи с его комплексностью и многогранностью. Имеются следующие недостатки:

* недостаточно функциональная графика (нельзя менять толщину линий, надписи не центрируются, текст иногда обрезается);
* нет возможности отобразить потоки данных между объектами и процессами;
* case-средство Rational Rose требователен к ресурсам компьютера;
* распространяется только по лицензии.

### StarUML

StarUML [19] – это пакет с открытым программным кодом, написанный на Delphi и работающий под управлением ОС семейства Windows. StarUML поддерживает UML 2.0 (плюс его профайлы), имеет кодогенерацию на распространенных языках программирования. Интерфейс пакета показался не удобным. Не поддерживает возможность генерации sql кода на основе диаграмм UML.

### Enterprise Architect

Enterprise Architect (EA) – CASE-инструмент для проектирования и конструирования программного обеспечения. EA поддерживает спецификацию UML2.0+, описывающую визуальный язык, которым могут быть определены модели проекта. Используя EA, можно выполнять форвард и реверс-инжиниринг ActionScript, C++, C#, Delphi, Java, Python, PHP, VB.NET and Visual Basic классов, синхронизировать код и элементы моделей, проектировать и генерировать элементы баз данных. Из моделей может быть быстро создана документация в стандартном rtf-формате и импортирована в MS Word для финального редактирования, так же доступна генерация HTML-документов. EA поддерживает все модели/диаграммы UML 2.0. С его помощью можно моделировать бизнес-процессы, веб-сайты, пользовательские интерфейсы, сети, конфигурации аппаратного обеспечения, сообщения, оценивать размер трудозатрат проектных работ в часах, фиксировать и трассировать требования, ресурсы, тест-планы, дефекты и запросы на изменения. Данное CASE-средство имеет огромные возможности для проектирования программным продуктов, а также интуитивно понятный и дружелюбный интерфейс, что уменьшает затраченное время на освоение Enterprise Architect.

### 1.3.6. Выбор CASE-средства для проектирования БД

В таблице 1.2 приведены некоторые характеристики case-средств, на основе которых было выбрано одно из них для проектирования БД.

Таблица 1.2. Сравнение CASE-средств

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Наименование case средства** | **Стоимость** | **Оценка функционала** | **Дружелюбность интерфейса** | **Возможность генерации БД** |
| Rational Rose | Платная, лицензия от 70000 руб. | Огромный | сложен для изучения | есть |
| StarUML | Бесплатная | большой, не под держивает реин жиринг | интерфейс  дружелюбен после привыкания | нет |
| Enterprise Architect | Платная лицензия от 4000 руб., есть пробная  версия | огромный, есть  кодогенерация, воз можность  реинжирования | не требует привыкания, интуитивно понятен, дружелюбный | есть |

Для проектирования логической структуры базы данных выбрано CASE средство Enterprise Architect. Его интуитивно понятный и дружелюбный интерфейс позволил затратить незначительное время для его изучения, поэтому на стадии изучения нотации UML была возможность сразу приступить к реализации диаграмм, что сократило время на создание UML диаграмм и построение логической структуры базы данных. Поддержка СУБД Firebird позволяет сгенерировать структуру базы данных в SQL коде. Дальнейшее изучение данного продукта, имеющего огромный пакет инструментов, даст возможность для проектирования графического интерфейса базы данных с возможностью генерации его кода на PHP языке.

# 2. ПРОЕКТИРОВАНИЕ ТАБЛИЦ БАЗЫ ДАННЫХ

## 2.1. Типы полей

Поля таблицы предназначены для хранения в них данных [20]. Это могут быть числа, текстовая информация, даты, графические файлы и т. д. Для определения типа данных, размещаемых в поле, используются тип поля, его ширина и количество знаков после запятой.

Для заданной логической модели «Издание» были созданы таблицы со следующим описанием полей.

Описание полей для таблицы Avtor представлено в таблице 2.1:

Таблица 2.1. Описание полей таблицы Avtor

|  |  |  |  |
| --- | --- | --- | --- |
| *Имя* | *Тип данных* | *Длина поля* | *Определение* |
| Id\_avtor | Int Autoinc | 11 | Личный номер автора |
| S\_name | Varchar | 30 | Фамилия автора |
| Name | Varchar | 30 | Имя автора |
| T\_name | Varchar | 30 | Отчество автора |
| Academ\_degree | Varchar | 30 | Ученая степень |

Описание полей для таблицы Authorship представлено в таблице 2.2:

Таблица 2.2. Описание полей таблицы Authorship

|  |  |  |  |
| --- | --- | --- | --- |
| *Имя* | *Тип данных* | *Длина поля* | *Определение* |
| Id\_avtor | Int | 11 | Личный номер автора |
| Id\_book | Int | 11 | Личный номер книги |

Описание полей для таблицы Book представлено в таблице 2.3:

Таблица 2.3. Описание полей таблицы Book

|  |  |  |  |
| --- | --- | --- | --- |
| *Имя* | *Тип данных* | *Длина поля* | *Определение* |
| Id\_book | Int Autoinc | 11 | Личный номер книги |
| Name\_book | Varchar |  | Название книги |
| ISBN | Bigint | 13 | Международный стандартный книжный номер |
| Number\_pages | Smallint | 4 | Количество страниц |
| Number\_chapters | Tinyint | 3 | Количество глав |
| Print\_date | Date | 8 | Дата пописи |

Описание полей для таблицы Edition\_department представлено в таблице 2.4:

Таблица 2.4. Описание полей таблицы Edition\_department

|  |  |  |  |
| --- | --- | --- | --- |
| *Имя* | *Тип данных* | *Длина поля* | *Определение* |
| Id\_edition | Int Autoinc | 11 | Личный номер редакции |
| Id\_book | Int | 11 | Личный номер книги |
| Format | Varchar | 20 | Формат печати |
| Type\_paper | Varchar | 30 | Тип бумаги |
| Type\_binding | Varchar | 30 | Тип переплета |

Описание полей для таблицы Stock\_department представлено в таблице 2.5:

Таблица 2.5. Описание полей таблицы Stock\_department

|  |  |  |  |
| --- | --- | --- | --- |
| *Имя* | *Тип данных* | *Длина поля* | *Определение* |
| Id\_stock | Int Autoinc | 11 | Личный номер склада |
| Id\_book | Int | 11 | Личный номер книги |
| Number\_products | Int | 11 | Количество книг |

Описание полей для таблицы Sale\_department представлено в таблице 2.6:

Таблица 2.6. Описание полей таблицы Sale\_department

|  |  |  |  |
| --- | --- | --- | --- |
| *Имя* | *Тип данных* | *Длина поля* | *Определение* |
| Id\_sale | Int Autoinc | 11 | Личный номер отдела продаж |
| Id\_book | Int | 11 | Личный номер книги |
| Price | Smallint | 4 | Цена |
| Number\_sales | Mediumint | 9 | Количество продаж |

Описание полей для таблицы Marketing\_department представлено в таблице 2.7:

Таблица 2.7. Описание полей таблицы Marketing\_department

|  |  |  |  |
| --- | --- | --- | --- |
| *Имя* | *Тип данных* | *Длина поля* | *Определение* |
| Id\_marketing | Int Autoinc | 11 | Личный номер книги |
| Id\_book | Int | 11 | Личный номер книги |
| Budget | Mediumint | 9 | Бюджет |
| Reputation | Float | 5,1 | Репутация |

Описание полей для таблицы Translation\_department представлено в таблице 2.8:

Таблица 2.8. Описание полей таблицы Translation\_department

|  |  |  |  |
| --- | --- | --- | --- |
| *Имя* | *Тип данных* | *Длина поля* | *Определение* |
| Id\_translation | Int Autoinc | 11 | Личный номер книги |
| Id\_book | Int | 11 | Личный номер книги |
| Original\_language | Varchar | 40 | Язык оригинала |
| Translated\_language | Varchar | 40 | Язык перевода |

Описание полей для таблицы Group\_departments представлено в таблице 2.9:

Таблица 2.9. Описание полей таблицы Group\_departments

|  |  |  |  |
| --- | --- | --- | --- |
| *Имя* | *Тип данных* | *Длина поля* | *Определение* |
| Id\_group | Int | 11 | Личный номер группы отделов |
| Id\_edition | Int | 11 | Личный номер редакции |
| Id\_stock | Int | 11 | Личный номер склада |
| Id\_sale | Int | 11 | Личный номер отдела продаж |
| Id\_marketing | Int | 11 | Личный номер книги |
| Id\_translation | Int | 11 | Личный номер книги |

Описание полей для таблицы Publish\_house представлено в таблице 2.10:

Таблица 2.10. Описание полей таблицы Publish\_house

|  |  |  |  |
| --- | --- | --- | --- |
| *Имя* | *Тип данных* | *Длина поля* | *Определение* |
| Id\_izd | Int Autoinc | 11 | Личный номер издательства |
| Name\_ph | Varchar | 30 | Название |
| City | Varchar | 30 | Город |
| Scale | Varchar | 30 | Масштаб |
| Geo\_indication | Varchar | 30 | Территориальный признак |
| Nature\_info | Varchar | 40 | Природа информации |
| INN | Bigint | 11 | Идентификационный номер налогоплательщика |
| Id\_group | Int | 11 | Личный номер группы отделов |
| Founding\_date | Date | 4 | Дата основания |

## 2.2. Первичные, внешние и альтернативные ключи

При определении первичных и внешних ключей, также, как и при создании таблиц, необходимо учитывать связи типа «предок-потомок» между таблицами. Так как внешний ключ – это атрибут первичного ключа таблицы предка, то сначала надо определить все первичные ключи предков, затем внешние и первичные ключи потомков, и так далее по иерархии. Ключи любого типа определяются при помощи индексов различного типа. Для создания первичных, внешних и альтернативных ключей была использована электронная документация "MyQSL" [2[1](#_Список_использованных_источников)].

В соответствии с правилом определения типа ключа, для таблиц БД «Издание» были определены следующие первичные и внешние ключи.

## 2.3. Ключи таблиц

Таблица Avtor содержит 1 ключ:

* первичный ключ id\_avtor, представляющий собой уникальный номер автора.



Рис. 2.1. Ключ таблицы Avtor

Таблица Book содержит 2 ключа:

* первичный ключ id\_book, представляющий собой уникальный номер книги;
* уникальный ключ ISBN, представляет собой Международный стандартный книжный номер.

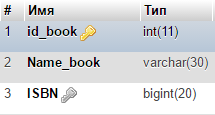


Рис. 2.2. Ключи таблицы Book

Таблица Authorship содержит 3 ключа:

* первичный ключ id\_authorship, представляющий собой уникальный номер авторства;
* внешний ключ id\_book, создает связь между таблицами;
* внешний ключ id\_avtor, создает связь между таблицами.

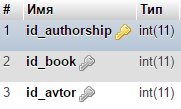


Рис. 2.3. Ключи таблицы Authorship

Таблица Edition\_department содержит 2 ключа:

* первичный ключ id\_edition, представляющий собой уникальный номер отдела редакции;
* внешний ключ id\_book, создает связь между таблицами.

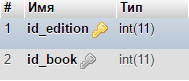


Рис. 2.4. Ключи таблицы equipment

Таблица Stock\_department содержит 2 ключа:

* первичный ключ id\_stock, представляющий собой уникальный номер отдела склада;
* внешний ключ id\_book, создает связь между таблицами.

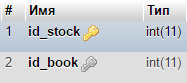


Рис. 2.5. Ключи таблицы Stock\_department

Таблица Sale\_department содержит 2 ключа:

* первичный ключ id\_sale, представляющий собой уникальный номер отдела продаж;
* внешний ключи id\_book, создает связь между таблицами.

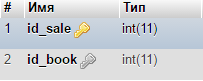


Рис. 2.6. Ключи таблицы Sale\_department

Таблица Marketing\_department содержит 3 ключа:

* первичный ключ id\_marketing, представляющий собой уникальный номер отдела маркетинга;
* внешний ключ id\_book, создает связь между таблицами.

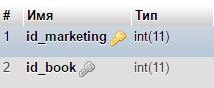


Рис. 2.7. Ключи таблицы Marketing\_department

Таблица Translation\_department содержит 2 ключа:

* первичный ключ id\_translation, представляющий собой уникальный номер отдела перевода;
* внешний ключ id\_book, создает связь между таблицами.

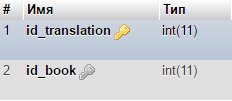


Рис. 2.8. Ключи таблицы Translation\_department

Таблица Group\_departments содержит 6 ключей:

* первичный ключ id\_group, представляющий собой уникальный номер группы отделов;
* внешний ключ id\_edition, создает связь между таблицами;
* внешний ключ id\_stock, создает связь между таблицами;
* внешний ключ id\_sale, создает связь между таблицами;
* внешний ключ id\_marketing, создает связь между таблицами;
* внешний ключ id\_translation, создает связь между таблицами.



Рис. 2.9. Ключи таблицы Group\_departments

Таблица Publish\_house содержит 2 ключа:

* первичный ключ id\_izd, представляющий собой уникальный номер издательства;
* внешний ключ id\_group, создает связь между таблицами.

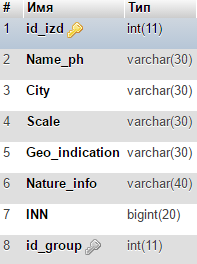


Рис. 2.10. Ключи таблицы Publish\_house

## 2.4. Определение правил ограничения ввода

В соответствии с описанием модели ПО, установить правила, ограничивающие ввод значений. Проверить работу установленных правил. Для создания ограничений была использована книга "MyQSL" [22]. Для ограничения полей таблицы Avtor было выбрано следующие поле: Academ\_degree. Его ограничение представлено в таблице 2.11.

Таблица 2.11. Ограничения поля таблицы Avtor

|  |  |  |
| --- | --- | --- |
| *Имя поля* | *Ограничение поля* | *Пояснение* |
| Academ\_degree | new.Academ\_degree NOT IN ('Бакалавр','Магистр','Доктор философии','Кандидат наук','Доктор наук') | Ученая степень может быть только 5 видов: бакалавр, магистр, Доктор философии, кандидат наук, Доктор наук |

Для ограничения полей таблицы Book были выбраны следующие поля: ISBN, Number\_pages. Их ограничения представлены в таблице 2.12.

Таблица 2.12. Ограничения полей таблицы Book

|  |  |  |
| --- | --- | --- |
| *Имя поля* | *Ограничение поля* | *Пояснение* |
| ISBN | CHAR\_LENGTH(CAST(new.ISBN AS CHAR)) <> 13 | Данное поле должно содержать ровно 13 цифр |
| Number\_pages | new.Number\_pages >2500 | Количество страниц не может быть больше 2500 штук |

Для ограничения полей таблицы Sale\_department было выбрано следующие поле: Price. Его ограничение представлено в таблице 2.13.

Таблица 2.13. Ограничения поля таблицы Sale\_department

|  |  |  |
| --- | --- | --- |
| *Имя поля* | *Ограничение поля* | *Пояснение* |
| Price | new.Price >20000 | Цена книги не может превышать 20000 |

Для ограничения полей таблицы Marketing\_department было выбрано следующие поле: Budget. Его ограничение представлено в таблице 2.14.

Таблица 2.14. Ограничения поля таблицы Marketing\_department

|  |  |  |
| --- | --- | --- |
| *Имя поля* | *Ограничение поля* | *Пояснение* |
| Budget | new.Budget >6000000 | Бюджет не может превышать 6000000 |

Для ограничения полей таблицы Edition\_department было выбраны следующие поля: Format, Type\_paper, Type\_binding. Их ограничения представлены в таблице 2.15.

Таблица 2.15. Ограничения полей таблицы Edition\_department

|  |  |  |
| --- | --- | --- |
| *Имя поля* | *Ограничение поля* | *Пояснение* |
| Format | new.Format NOT IN ('Сверхкрупный','Крупный','Средний','Малый','Сверхмалый') | Формат может быть только 5 видов:сверхкрупный, крупный, средний, малый, сверхмалый |
| Type | new.Type\_paper NOT IN ('Офсетная','Мелованная','Дизайнерская') | Тип бумаги может быть только 3 видов: офсетная, мелованная, дизайнерская |
| Type\_binding | new.Type\_binding NOT IN ('Мягкий переплет','Твердый переплет','Брошюровка скобой') | Тип переплета может быть только 3 видов: мягкий переплет, твердый переплет, брошюровка скобой |

Для ограничения полей таблицы Publish\_house были выбраны следующие поля: Scale, Geo\_indication, Nature\_info, INN. Их ограничения представлены в таблице 2.16.

Таблица 2.16. Ограничения полей таблицы Publish\_house

|  |  |  |
| --- | --- | --- |
| *Имя поля* | *Ограничение поля* | *Пояснение* |
| Scale | new.Scale NOT IN ('Крупный','Средний','Малый') | Масштаб может быть только 3 видов: крупный, средний, малый |
| Geo\_indication | new.Geo\_indication NOT IN ('Местный','Национальный','Транснациональный') | Территор. признак может быть только 3 видов: местный, национальный, транснациональный |
| Nature\_info | new.Nature\_info NOT IN ('Текстовые книги','Электронные издания','Картографическая продукция') | Природа информации может быть только 3 видов: текстовые книги, электронные издания, картографическая продукция |
| INN | CHAR\_LENGTH(CAST(new.INN AS CHAR)) <> 10 | Поле ИНН не может содержать больше 10 цифр |

# 

## 2.5. Организация ввода данных

Необходимо ввести данные в таблицы БД для будущей демонстрации работы SQL запросов:

* данные не должны носить фривольный характер;
* конкретные значения атрибутов должны быть максимально приближены к реальности;
* обязательно соблюдать ограничения ссылочной целостности. Это означает, что атрибуты, отмеченные в подчиненной таблице спецификатором FK (внешний ключ [23]), не могут принимать значения, которые не существуют среди значений соответствующего атрибута родительской таблицы. На диаграмме линия связи «предок-потомок» отмечается точкой со стороны подчиненной таблицы;
* подчиненная таблица должна содержать несколько записей, соответствующих одной записи родительской таблицы.
* для соблюдения правил ссылочной целостности рекомендуется такая же последовательность заполнения таблиц от предков к потомкам.

Введенные данные, удовлетворяющие ТЗ для таблицы Avtor на рисунке 2.11.



Рис. 2.11. Введенные данные в таблицу Avtor

Введенные данные, удовлетворяющие ТЗ для таблицы Book на рисунке 2.12.



Рис. 2.13. Введенные данные в таблицу Book

Введенные данные, удовлетворяющие ТЗ для таблицы Authorship на рисунке 2.14.



Рис. 2.14. Введенные данные в таблицу Authorship

Введенные данные, удовлетворяющие ТЗ для таблицы Edition\_department на рисунке 2.15.



Рис. 2.15. Введенные данные в таблицу Edition\_department

Введенные данные, удовлетворяющие ТЗ для таблицы Stock\_department на рисунке 2.16.

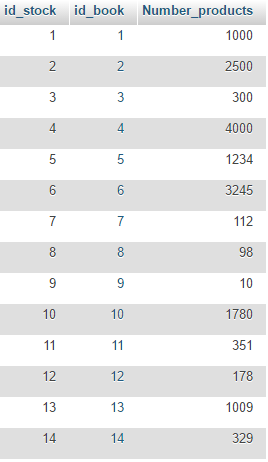


Рис. 2.16. Введенные данные в таблицу Stock\_department

Введенные данные, удовлетворяющие ТЗ для таблицы Sale\_department на рисунке 2.17.

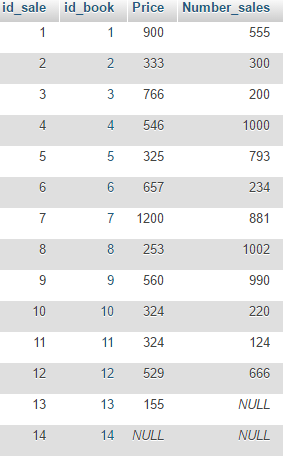


Рис. 2.17. Введенные данные в таблицу Sale\_department

Введенные данные, удовлетворяющие ТЗ для таблицы Marketing\_department на рисунке 2.18.

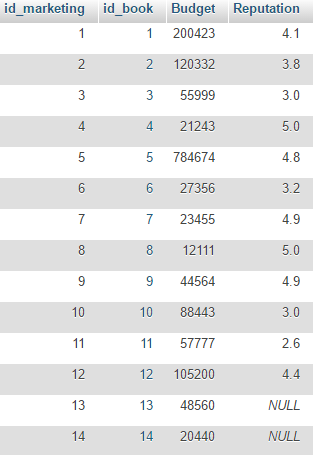


Рис. 2.18. Введенные данные в таблицу Marketing\_department

Введенные данные, удовлетворяющие ТЗ для таблицы Translation\_department на рисунке 2.19.



Рис. 2.19. Введенные данные в таблицу Translation\_department

Введенные данные, удовлетворяющие ТЗ для таблицы Group\_departments на рисунке 2.20.

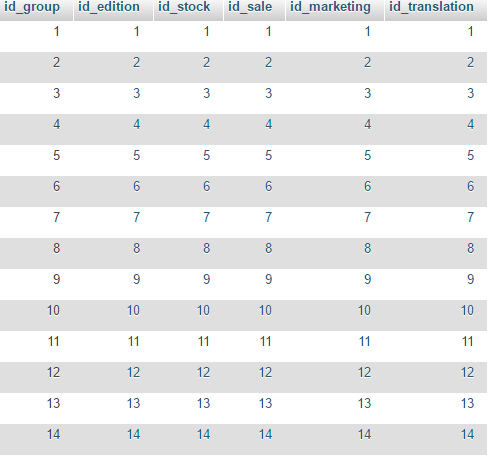


Рис. 2.20. Введенные данные в таблицу Group\_departments

Введенные данные, удовлетворяющие ТЗ для таблицы Publish\_house на рисунке 2.21.

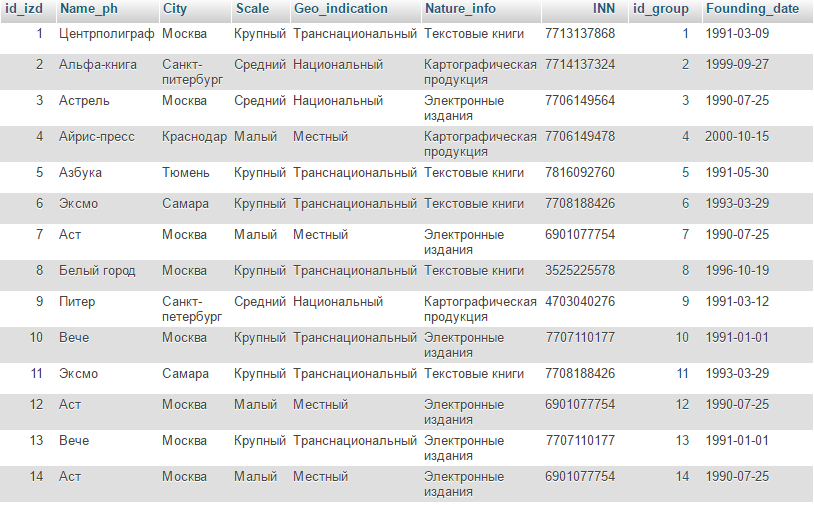
****

Рис. 2.21. Введенные данные в таблицу Publish\_house

# Запросы

При необходимости просмотреть, добавить, изменить или удалить данные из базы данных, удобно использовать SQL запросы. Запросы можно использовать для фильтрации данных, выполнения расчетов на основе данных и отображения сводных данных. Кроме того, запросы позволяют автоматизировать выполнение многих задач управления данными и просматривать изменения в данных перед их использованием.

SQL запрос, представляет собой обращение к данным для получения информации и выполнения действий с данными. Запрос можно использовать для получения ответа на поставленный вопрос, выполнения расчетов, объединения данных из разных таблиц или же для добавления, изменения или удаления данных в таблице. Запросы, используемые для извлечения данных из таблицы или выполнения расчетов, называются запросами на выборку. Запросы, используемые для добавления, изменения или удаления данных, называются запросами на изменение. Осуществляем первый запрос: найти все книги, у которых количество страниц варьируется от 100 до 450 страниц. Результаты запроса показаны на рисунке 2.22.

SELECT \*

# выбираем все записи

FROM Book B

# из таблицы Book

WHERE B.Number\_pages BETWEEN 100 AND 450;

# которые удовлетворяют условию BETWEEN 100 AND 450

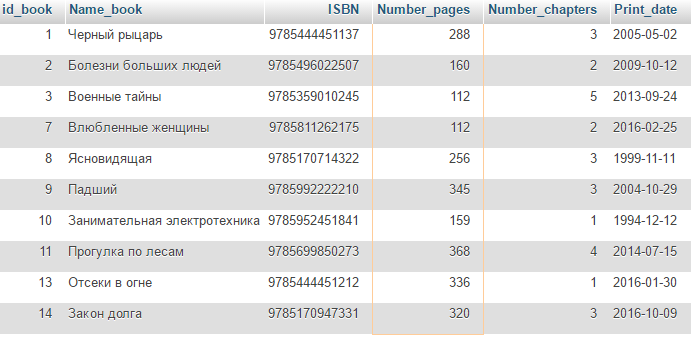


Рис. 2.22. Результат первого запроса

Осуществляем второй запрос: найти сумму бюджета всех книг, у которых оценка варьируется от 3 до 5. Результат показан на рисунке 2.23.

SELECT SUM(Budget)

# используем агрегирующею функцию Sum, которая сложит весь

# бюджет книг

FROM Marketing\_department M

# из таблицы Marketing\_department

WHERE M.Reputation BETWEEN 3 AND 5;

# используем условие, где поле M.Reputation удовлетворяет

# условию BETWEEN 3 AND 5



Рис. 2.23. Результат второго запроса

Осуществляем третий запрос: найдем все языки оригинала и сгруппируем их. Результат запроса показан на рисунке 2.24.

SELECT Original\_language, COUNT(\*)

# выберем поле Original\_language, и агрегирующею функцию

# COUNT(\*), которая будет считать количество записей

FROM Translation\_department

# из таблицы Translation\_department

GROUP BY Original\_language;

# и сгрупперуем результат запроса по полю Original\_language,

# c помощью конструкции GROUP BY

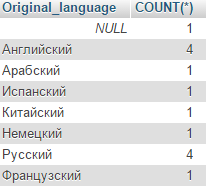


Рис. 2.24. Результат третьего запроса

Осуществляем четвертый запрос: найдем всех авторов, которые не написали ни дну книгу. Результат выполнения SQL запроса представлен на рисунке 2.25.

SELECT A.id\_avtor

# выбираем поле id\_avtor

FROM Avtor A

# из таблицы Avtor

LEFT JOIN Authorship ON A.id\_avtor = Authorship.id\_avtor

# используем оператор левого присоединения LEFT JOIN,

# который присоединит 2 таблицы по ключам

WHERE Authorship.id\_book IS NULL;

# так же используем условие,

# где поле Authorship.id\_book равно NULL



Рис. 2.25. Результат четвёртого запроса

Осуществляем пятый запрос: выведем всех авторов, которые написали книгу и присоединим таблицу с книгами, которые они написали. Результат запроса показан на рисунке 2.26.

SELECT \*

# выберем все записи

FROM Avtor

# из таблицы Avtor

INNER JOIN Authorship ON Avtor.id\_avtor = Authorship.id\_avtor

# присоединим таблицу Authorship,

# используя оператор INNER JOIN

INNER JOIN Book ON Authorship.id\_book = Book.id\_book

# присоединим таблицу Book

WHERE Authorship.id\_book is NOT NULL;

# используем условие, где поле Authorship.id\_book равно NULL

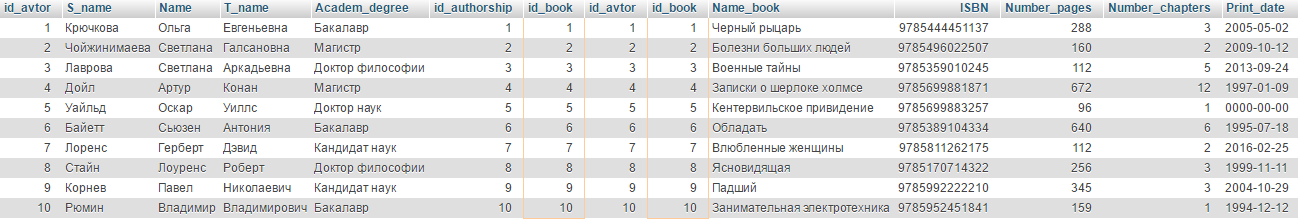


Рис. 2.26. Результат пятого запроса

Осуществляем шестой запрос: найдем всех авторов, которые написали больше, чем одну книгу. Результат запроса представлен на рисунке 2.27.

SELECT S\_name, Name, T\_name, COUNT(\*)

# выберем поля S\_name, Name, T\_name

# и агрегирующею функцию COUNT(\*)

FROM Avtor A

# из таблицы Avtor

INNER JOIN Authorship Au ON A.id\_avtor = Au.id\_avtor

# присоединим к таблице Avtor таблицу Authorship

GROUP BY A.id\_avtor HAVING COUNT(\*) > 1;

# сгруппируем поле A.id\_avtor и используем условие отбора

# при помощи конструкции HAVING



Рис. 2.27. Результат шестого запроса

Осуществляем седьмой запрос: найдем все книги, которые прошли через все отделы издательства. Результат запроса представлен на рисунке 2.28.

SELECT Name\_book, ISBN

# выберем поля Name\_book, ISBN

FROM Book B

# из таблицы Book

INNER JOIN Edition\_department E ON B.id\_book = E.id\_book

# присоединим к таблице Book таблицу Edition\_department

INNER JOIN Stock\_department ST ON B.id\_book = ST.id\_book

# присоединим к таблице Book таблицу Stock\_department

INNER JOIN Sale\_department S ON B.id\_book = S.id\_book

# присоединим к таблице Book таблицу Sale\_department

INNER JOIN Marketing\_department M ON B.id\_book = M.id\_book

# присоединим к таблице Book таблицу Marketing\_department

INNER JOIN Translation\_department T ON B.id\_book = T.id\_book

# присоединим к таблице Book таблицу Translation\_department

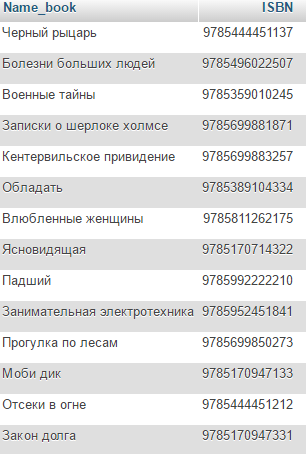


Рис. 2.28. Результат седьмого запроса

Осуществляем восьмой запрос, состоящий из двух: в первом запросе, найдем все книги, у которых количество страниц варьируется от 100 до 370 штук; во втором запросе найдем все книги, у которых количество глав варьируется от 1 до 4. Результаты двух запросов объединим. Результат запроса представлен на рисунке 2.29.

SELECT Name\_book, Number\_pages

FROM Book B

WHERE B.Number\_pages BETWEEN 100 AND 370

Union

SELECT Name\_book,Number\_chapters

FROM Book B

WHERE B.Number\_chapters BETWEEN 1 AND 4;



Рис. 2.29. Результат восьмого запроса

Осуществляем девятый запрос: найдем язык оригинала и количество книг, написанных на нем, сгруппируем и отсортируем по убыванию. Результат запроса представлен на рисунке 2.30.

SELECT Original\_language, COUNT(\*)

# выберем поле Original\_language

# и агрегирующею функцию COUNT(\*)

FROM Translation\_department

# из таблицы Translation\_department

GROUP BY Original\_language ORDER BY Original\_language DESC;

# группируем по полю Original\_language, при помощи GROUP BY

# и сортируем по полю Original\_language, при помощи ORDER BY

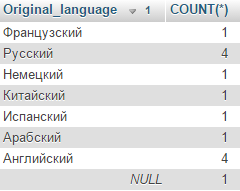


Рис. 2.30. Результат девятого запроса

Осуществляем десятый запрос: найдем всю информацию о книге, чей личный номер равен 6. Результат запроса представлен на рисунке 2.31.

SELECT S\_name, T\_name, Name\_book, ISBN,Number\_pages, Number\_chapters,Print\_date,Format,Type\_paper,Type\_binding,Price,Number\_sales,Budget,Reputation

#выберем поля S\_name,T\_name,Name\_book,ISBN,Number\_pages,

# Number\_chapters,Print\_date,Format,Type\_paper,Type\_binding,

# Price,Number\_sales,Budget,Reputation

FROM Book B

# из таблицы Book

INNER JOIN Authorship AU ON B.id\_book = AU.id\_book

# присоединим к таблице Book таблицу Authorship

INNER JOIN Avtor A ON AU.id\_avtor = A.id\_avtor

# присоединим к таблице Authorship таблицу Avtor

INNER JOIN Edition\_department E ON B.id\_book = E.id\_book

# присоединим к таблице Book таблицу Edition\_department

INNER JOIN Sale\_department S ON B.id\_book = S.id\_book

# присоединим к таблице Book таблицу Sale\_department

INNER JOIN Marketing\_department M ON B.id\_book = M.id\_book

# присоединим к таблице Book таблицу Marketing\_department

WHERE B.id\_book = 6;

# используем условие, где поле B.id\_book,

# удовлетворяет условию B.id\_book = 6



Рис. 2.31. Результат десятого запроса

Осуществляем одиннадцатый запрос: найдем издательства, у которых территориальный признак равен «Транснациональный», а название начинается на «А» и книги которые эти издательства издали. Результат запроса представлен на рисунке 2.32.

SELECT Name\_book,Name\_ph,City,Scale,Geo\_indication,Nature\_info,INN

# выберем поля Name\_book,Name\_ph,City,Scale,

# Geo\_indication,Nature\_info,INN

FROM Publish\_house P

# из таблицы Publish\_house

INNER JOIN Group\_departments G ON P.id\_group = G.id\_group

# присоединим к таблице Publish\_house

# таблицу Group\_departments

INNER JOIN Edition\_department E ON G.id\_edition = E.id\_edition

# присоединим к таблице Group\_departments

# таблицу Edition\_department

INNER JOIN Book B ON E.id\_book = B.id\_book

# присоединим к таблице Edition\_department таблицу Book

WHERE P.Geo\_indication = 'Транснациональный' AND P.Name\_ph LIKE 'а%';

# используем условие, где поле P.Geo\_indication,

# удовлетворяет условию

# P.Geo\_indication = 'Транснациональный'

# и удовлетворяет условию

# P.Name\_ph LIKE 'а%'



Рис. 2.32. Результат одиннадцатого запроса

Осуществляем двенадцатый запрос: найдем все данные о книги, номер, который найдем из подзапроса. Результат запроса представлен на рисунке 2.33.

SELECT S\_name,T\_name,Name\_book,ISBN,Number\_pages,Number\_chapters, Print\_date,Format,Type\_paper,Type\_binding,Price, Number\_sales,Budget,Reputation

# выберем поля S\_name,T\_name,Name\_book,ISBN,Number\_pages,

# Print\_date,Format,Type\_paper,Type\_binding,Price,

# Number\_sales,Budget,Reputation

FROM Book B

# из таблицы Book

LEFT JOIN Authorship AU ON B.id\_book = AU.id\_book

# присоединяем к таблице Book таблицу Authorship

LEFT JOIN Avtor A ON AU.id\_avtor = A.id\_avtor

# присоединяем к таблице Authorship таблицу Avtor

LEFT JOIN Edition\_department E ON B.id\_book = E.id\_book

# присоединяем к таблице Book таблицу Edition\_department

LEFT JOIN Sale\_department S ON B.id\_book = S.id\_book

# присоединяем к таблице Book таблицу Sale\_department

LEFT JOIN Marketing\_department M ON B.id\_book = M.id\_book

# присоединяем к таблице Book таблицу Marketing\_department

WHERE B.id\_book =

# используем условие, которое является подзапросом

(SELECT COUNT(\*)

# выберем все поля

FROM Book B

# из таблицы Book

WHERE B.Number\_pages BETWEEN 50 AND 800);

# используем условие, где поле B.Number\_pages,

# удовлетворяет условию BETWEEN 50 AND 800



Рис. 2.33. Результат двенадцатого запроса

# ЗАКЛЮЧЕНИЕ

В рамках курсовой работы была создана база данных "Издательство", которая представляет собой комплексную систему для хранения и управления информацией о книгах, авторах, издательствах и других сущностях, связанных с издательским делом. В процессе разработки базы данных были проведены следующие этапы:

1. Анализ предметной области, включающий изучение требований заказчика, определение основных сущностей и их атрибутов, а также выявление связей между ними.

2. Инфологическое проектирование, где на основе анализа предметной области были построены сущности и их атрибуты в виде информационной модели.

3. Концептуальное проектирование, где была разработана концептуальная модель данных с использованием диаграммы сущность-связь для более наглядного представления структуры данных.

4. Логическое проектирование, включающее создание схемы базы данных с использованием реляционной модели и языка SQL для определения таблиц, ключей и связей между ними.

5. Физическое проектирование, где были определены типы данных, индексы и другие аспекты, связанные с конкретной реализацией базы данных на выбранной СУБД.

Полученная база данных "Издательство" представляет собой гибкую и эффективную систему для хранения и управления информацией об изданных книгах, их авторах, издательствах, а также других сопутствующих данных. Она может быть использована издательскими компаниями для ведения каталогов книг, контроля над продажами и управления авторскими правами.

Таким образом, разработка базы данных "Издательство" представляет собой важный шаг в создании инструмента для эффективного управления издательскими процессами и обеспечения качественного обслуживания клиентов. В результате выполнения курсового проекта была построена концептуальная модель процесса «Издание» и описана структура базы данных. Были построены таблицы с детальным описанием всех сущностей, атрибутов и доменов, созданы ключи, ограничения, последовательности, запросы. По данной предметной области и ее модели возможна реализация этой модели в СУБД. Также в ходе данной курсовой работы была изучена программная среда phpMyAdmin. Были изучены и применены принципы работы SQL запросов.

# СПИСОК ЛИТЕРАТУРЫ

1. Пирогов В.Ю. Информационные системы и базы данных: организация и проектирование: учеб. пособие. – Спб.: БХВ-Петербург, 2009.;

2. Чистов Д.В. Информационные системы в экономике: учеб.пособие. М.: ИНФРА-М, 2009.;

3. Заботина Н.Н. Проектирование информационных систем. М.: ИНФРАМ, 2013.;

4. Емельянова Н.З., Попов И.И., Партыка Т.Л. Проектирование информационных систем: учеб.пособие. – М.: Форум, 2009.;

5. Романов, А.Н. Информационные системы в экономике (лекции, упражнения и задачи): Учеб. пособие/А.Н. Романов, Б.Е. Одинцов – М.:Вузовский учебник, 2006. – 300 с.;

6. Советов Б.Я., Яковлев С.А. Моделирование систем: Практикум. – М.: Высшая школа, 2005. – 224 с.;

7. Колесов Ю.Б., Сениченков Ю.Б Моделирование систем. Практикум по компьютерному моделированию. – Спб.:БХВ-Петербург, 2005. — 282 с.;

8. Чистов Д.В., Шуремов Е.Л., Лямова и др. Информационные системы управления предприятием. М.: Бухгалтерский учет, 2006.

9. Митин А.А., Горяйнов А.С. Разработка формальной модели для решения задачи составления расписания в Вузе. Информационные системы и технологии. Материалы II международной научно-технической интернет-конференции. – Орел: Госуниверситет-УНПК, 2013. – Режим доступа: http://irsit.ru/files/article/343.pdf.

10. Волков В.Н., Стычук А.А., Митин А.А. Разработка информационной модели представления данных о регламенте электронной услуги. Информационные системы и технологии. – Орел.: Госуниверситет -УНПК, 2014. – № 4(84). – С. 21-30

11. Виноградова Е.Ю., Шориков А.Ф. Принципы и задачи разработки программного обеспечения для системы управления планированием на предприятии // Информационные технологии в экономике: теория, модели и методы. Сб. научных трудов. Екатеринбург: Изд-во Уральского государственного экономического университета, 2006.

12. Шориков А.Ф., Виноградова Е.Ю. Проектирование информационного обеспечения для решения динамических задач управления газотранспортным предприятием // Известия Уральского государственного экономического университета. 2006. № 2 (14).

13. Виноградова Е.Ю., Шориков А.Ф. Методика проектирования информационного обеспечения для решения динамических задач управления предприятием // Современные проблемы прикладной информатики. Сборник научных трудов II научно-практической конференции. СПб.: СПбГИЭУ, 2006.

14. Виноградова Е.Ю., Шориков А.Ф. Моделирование и оптимизация комплексного управления технологическими процессами на предприятии // Современные проблемы прикладной информатики. Сборник научных трудов III научно-практической конференции. СПб.: СПбГИЭУ, 2007.

15. Вентцель Е.С. Исследование операций. М.: Советское радио, 1972.

16. Лотов А.В. Введение в экономико-математическое моделирование. М.: Наука, Главная редакция физико-математической литературы, 1984.

17. Баженова И.Ю. Проектирование приложений баз данных, Издательство: Бином. Лаборатория знаний, Интернет-университет информационных технологий, 2016 г., 328 с.

18. Вендров А.М. CASE-технологии. Современные подходы к проектированию информационных систем - М.: Финансы и статистика, 2014 г., 456 с.

19. Вигерс Карл. Основные требования к программному обеспечению, Пер, с англ. - М.:Издательско-торговый дом "Русская Редакция", 2014. -576 с.

20. Гашков С.Б., Применко Э.А., Черепнев М.А. Основы криптографической защиты информации, М, Издательство: Академия, 2015 г., 304 с.

21. Гвоздева Т. В., Баллод Б.А, Основы проектирования информационных систем, М, Издательство: Феникс, 2014 г., 512 с.

22. Гвоздева В.А. Информатика, автоматизированные информационные технологии и системы: Учебник / В.А. Гвоздева. - М.: ИД Форум, НИЦ Инфра-М, 2013. - 544 c.

23. Голицына О.Л., Попов И.И., Максимов Н.В., Партыка Т.Л. Современные информационные технологии, М, Издательство Инфра-М, 2016, 608 с.