**СОДЕРЖАНИЕ**

ВВЕДЕНИЕ 4

1 АНАЛИТИЧЕСКАЯ ЧАСТЬ 5

1.1 Анализ предметной области 5

1.2 Постановка задачи 6

2 РАЗРАБОТКА ЛОГИЧЕСКОЙ МОДЕЛИ БАЗЫ ДАННЫХ 9

2.1 Построение диаграммы «сущность-связь» в нотации П.Чена 10

2.2 Построение модели основанной на ключах 11

2.3 Построение полной атрибутивной модели 12

3 РАЗРАБОТКА ФИЗИЧЕСКОЙ МОДЕЛИ БАЗЫ ДАННЫХ 13

3.1 Выбор аппаратной и программной платформы для реализации БД 14

3.2 Реализация базы данных 15

3.3 Тестирование базы данных 16

3.4 Разграничение прав доступа 17

3.5 Расчет информационных параметров базы данных 18

4 РАЗРАБОТКА КЛИЕНТСКОГО ПРИЛОЖЕНИЯ 19

4.1 Обоснование выбора языка программирования 20

4.2 Разработка интерфейса пользователя 21

4.3 Алгоритм работы каждого из модулей 22

4.4 Тестирование работы приложения 23

ЗАКЛЮЧЕНИЕ 24

СПИСОК ЛИТЕРАТУРЫ И ИНФОРМАЦИОННЫХ РЕСУРСОВ 25

ПРИЛОЖЕНИЕ А SQL скрипт создания базы данных 26

ПРИЛОЖЕНИЕ Б Код клиентского приложения 27

**ВВЕДЕНИЕ**

В современном мире вся информация, хранимая и использующаяся в процессе работы, должна быть жестко структурирована и упорядочена. Это необходимо для того, чтобы упростить поиск, работу и сохранение данных.

В ходе выполнения курсового проекта разрабатывается физическая модель базы данных для учета и хранения постояльцев мини-отеля в рамках технического задания, утвержденного \_\_ \_\_\_\_\_\_\_ 2018 г. Севастопольским государственным университетом.

Целями курсового проектирования является закрепление теоретических знаний, полученных в результате изучения курсов «Технологии баз данных» и «Управление данными», формирование практических навыков в области информационного моделирования и проектирования баз данных, разработка физической модели базы данных.

Для достижения цели на разных этапах курсового проектирования должны быть решены следующие задачи:

* выбор варианта задания детализация постановки задачи и анализ предметной области;
* разработка логической модели базы данных;
* разработка и тестирование физической модели БД;
* разработка клиентского приложения.

Данная пояснительная записка состоит из: содержания, введения, а также других разделов. Раздел «Аналитическая часть», в котором описывается предметная область. Раздел «Разработка логической модели базы данных», в котором отражены логические связи между сущностями. Раздел «Разработка физической модели базы данных» описывает физические свойства базы данных (типы данных, размер полей, индексы). Раздел «Разработка клиентского приложения» в котором находятся сведения о выбранной платформе реализации, описание взаимодействие модулей, а также описание работы пользователя с ней.

**1 АНАЛИТИЧЕСКАЯ ЧАСТЬ**

**1.1 Анализ предметной области**

В базе данных для выбранной предметной области «Мини-отель» необходимо как можно более точно производить учет занятых и свободных номеров, находящихся постояльцев в отеле, а также дополнительных услуг для каждого клиента.

Типичный пример работы отеля: постоялец заселяется в номер, заказывает дополнительные услуги, выселяется из номера.

**1.2 Постановка задачи**

Основные объекты предметной области:

* номер;
* постоялец;
* услуга.

Связи между объектами:

* постоялец заселяется в один из заранее созданных номеров на некоторое время;
* услуга предоставляется клиенту, заселившемуся на некоторое время.

Пользователем данной базы данных является управляющей мини-отелем. Он может добавлять, удалять, изменять информацию о номерах и услугах, предоставляемых постояльцам, добавлять, изменять информацию о постояльцах, изменять даты заезда и менять информацию о том, заселены ли они в отель.

**2 РАЗРАБОТКА ЛОГИЧЕСКОЙ МОДЕЛИ БАЗЫ ДАННЫХ**

**2.1 Построение диаграммы «сущность-связь» в нотации П.Чена**

Разработка «сущность-связь» основывается на переходе от сложной сетевой структуры к древовидной и построения диаграммы.

Для начала была сформирована сложная сетевая структура, основанная на основных объектах предметной области: Рисунок 1.



Рисунок 1 – Сложная сетевая структура

На основе сложной сетевой структуры была синтезирована простая сетевая структура путем добавления промежуточных таблиц для того что бы связь многие ко многим трансформировать в связь один ко многим.



Рисунок 2 – Простая сетевая структура

На основе простой сетевой структуры была построена древовидная структура, отображающая взаимосвязь ближайших объектов базы, а также показывающая будущие внешние ключи.



Рисунок 3 – Древовидная структура

Далее на основе прошлых таблиц была синтезирована ER диаграмма, отображающая взаимосвязь объектов, а также основные элементы взаимодействующих объектов.

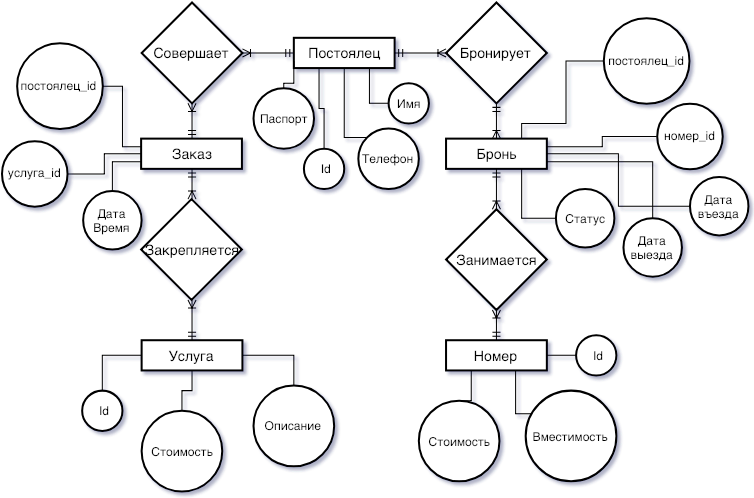


Рисунок 4 – ER диаграмма в нотации П.Чена

Объект «Постоялец» (id – уникальный номер, паспорт – информация о паспортных данных гостя, имя – ФИО постояльца, телефон – мобильный номер телефона постояльца) связывается с объектом «Номер» (id – номер в отеле, стоимость данного номера и его вместимость) через объект «Бронь» (уникальный номер постояльца, номер в отеле, дата въезда и выезда, статус брони(постоялец в отеле или нет)), объект «Услуга» (уникальный номер, стоимость услуги, описание услуги) соединяется с постояльцем через объект «Заказ» (уникальный номер постояльца, уникальный номер услуги, дата предоставления).

**2.2 Построение модели основанной на ключах**

Данная модель описывает основные структуры данных и сущности.

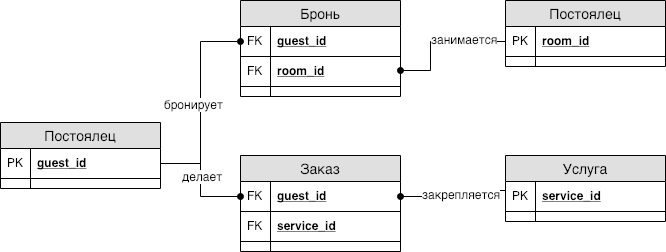


Рисунок 5 – Модель основанная на ключах

**2.3 Построение полной атрибутивной модели**

Нормализация – это разбиение таблицы на две или более, обладающих лучшими свойствами при включении, изменении и удалении данных. Окончательная цель нормализации сводится к получению такого проекта базы данных, в котором каждый факт появляется лишь в одном месте, т.е. исключена избыточность информации. Это делается не столько с целью экономии памяти, сколько для исключения возможной противоречивости хранимых данных. За основу возьмем выделенные ранее сущности с соответствующими им атрибутами.

Рассмотрим все выделенные нами сущности, и проверим их на атомарность. В каждой из сущности соблюдается атомарность, следовательно, делаем вывод о том, что наша база данных находится в первой нормальной форме.

Так как наша база данных, находится в первой нормальной форме, рассмотрим её на удовлетворение второй нормальной форме (Отношение R находится во второй нормальной форме в том и только в том случае, когда находится в 1НФ, и каждый не ключевой атрибут полностью зависит от первичного ключа). Каждая сущность содержит уникальный, искусственно введенный, первичный ключ, не обладающий смысловой нагрузкой, от которого полностью зависят все не ключевые атрибуты, это удовлетворяет условиям второй нормальной форме, следовательно, делаем вывод о том, что база данных находится во второй нормальной форме

Так как наша база данных находится во второй нормальной форме рассмотрим на её удовлетворение третьей нормальной форме (Отношение R находится в третьей нормальной форме в том и только в том случае, если находится в 2НФ и каждый не ключевой атрибут не транзитивно зависит от первичного ключа). Во всех созданных сущностях транзитивные зависимости отсутствуют, что говорит о том, что наша база данных находится в третьей нормальной форме.

В ходе нормализации была построена полная атрибутивная модель в нотации IDEF1X, которая представлена на рисунке 6.

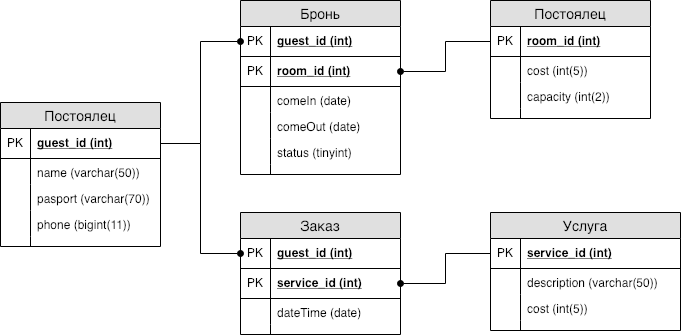


Рисунок 6 – Полная атрибутивная модель

**3 РАЗРАБОТКА ФИЗИЧЕСКОЙ МОДЕЛИ БАЗЫ ДАННЫХ**

**3.1 Выбор аппаратной и программной платформы для реализации базы данных**

Выбор системы управления базами данных представляет собой сложную многопараметрическую задачу и является одним из важных этапов при разработке приложений баз данных. Выбранная СУБД должна удовлетворять как текущим, так и будущим потребностям системы. В настоящее время существует достаточно много различных СУБД, но в данной курсовой работе была выбрана именно MySQL, т.к. она обладает следующими преимуществами:

* Быстродействие. СУБД MySQL является одной из самых быстрых баз данных из имеющихся на современном рынке.
* Простота использования. MySQL является высокопроизводительной и относительно простой в использовании СУБД.
* Возможности доступа. Сервер позволяет одновременно подключаться неограниченному количеству пользователей, одновременно работающих с базой данных.
* Аппаратная совместимость. MySQL отлично работает под управлением большинства операционных систем.
* Бесплатное использование. MySQL полностью бесплатна. Исходный код этой системы открыт и любой желающий может разработать на его базе собственные некоммерческие проекты, при условии соблюдения требований лицензии.

Все эти особенности сделали MySQL идеально подходящей СУБД под нашу базу данных.

**3.2 Реализация базы данных**

Так как у нас уже имеется логическая модель базы данных в третьей нормальной форме, а также выбрана СУБД, которая её будет реализовать, преобразуем логическую модель в физическую. На рисунке 7 представлена физическая модель нашей базы данных.

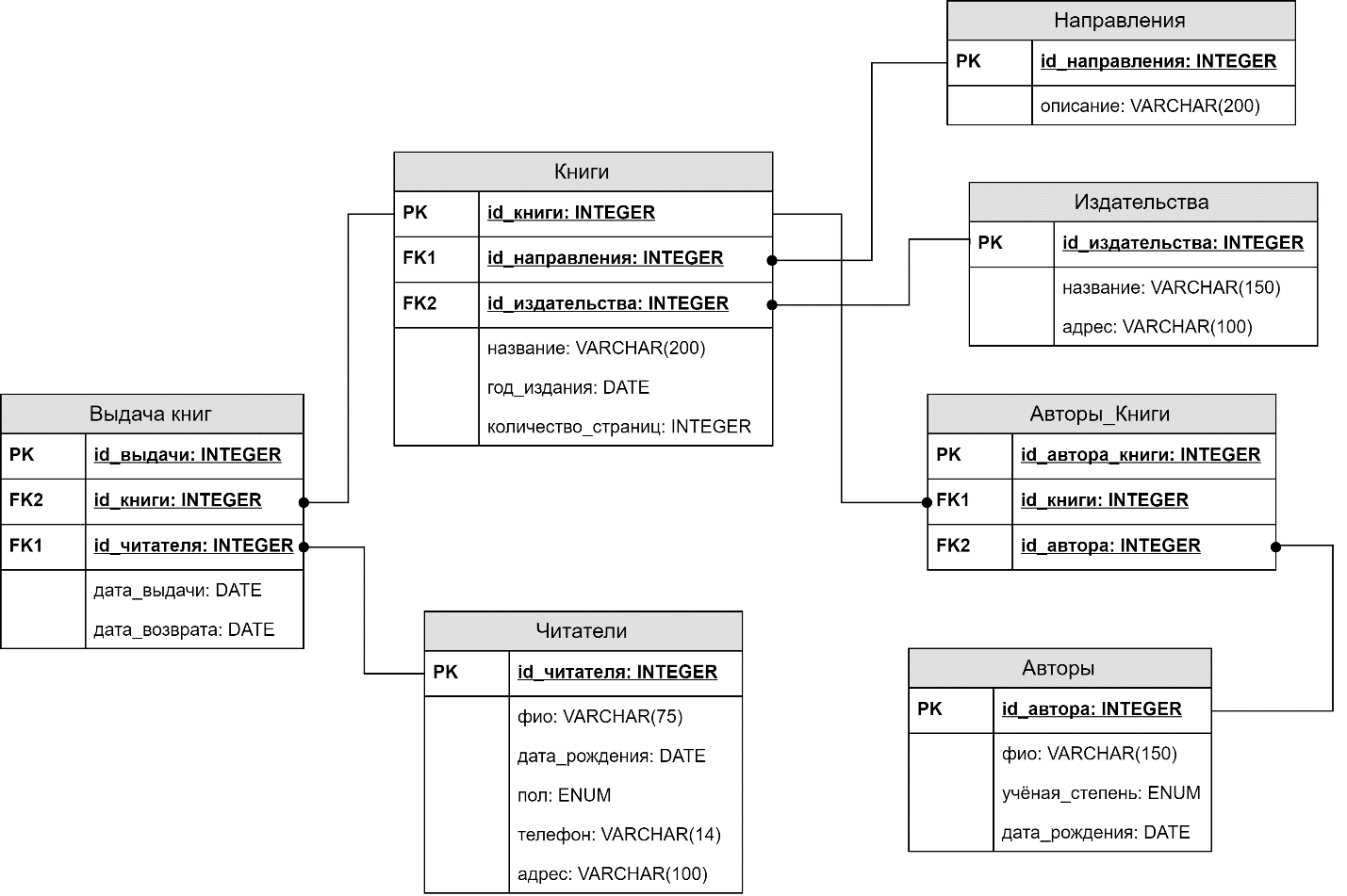


Рисунок 7 – Физическая модель базы данных

**3.3 Тестирование базы данных**

После создания таблиц и заполнения основными данными были произведены изменения в основных таблицах, удаление некоторых полей и наблюдение за тем, изменились ли данные в дочерних таблицах.

// Скрины удаления и редактирования

**3.4 Разграничения прав доступа**

Для работы с проектируемой базой данных было выделено 2 пользователя, имеющие различные уровни доступа – это администратор и читатель.

Администратор управляет выдачами книг и имеет полный доступ к остальным таблицам в базе данных.

Читатель имеет право просматривать и выполнять поиск по всем книгам, хранящимся в библиотеке, авторам, направлениям и издательствам.

**3.5 Расчет информационных параметров базы данных**

Длина логической записи j-ого файла определяется как сумма длин полей, её составляющих и указателей, если они организуются разработчиком

где – число групп полей в записях; – длина группы [байт].

Для переменных длин полей возьмем максимальную длину поля, следовательно, размер типов данных для хранения в памяти будет составлять:

1. INTEGER – 4 байта
2. VARCHAR(x) – x байт
3. DATE – 3 байта
4. ENUM – 2 байта

Вычислим:

Объем памяти, необходимый для размещения информационного фонда без учёта системных данных и указателей составит

где – число типов записей в информационном фонде; – количество записей -го файла.

Предположим, что в базе содержится:

20 записей о выдаче книг, 45 книг, 30 записей о читателях, 60 направлений, 15 издательств, 60 авторов и 45 связей авторы-книги, тогда

Приращение информационного фонда вычисляется как

где – число добавленных типов записей; – интенсивность добавления записей в файл -го типа.

Получаем:

Время заполнения информационного фонда определяется как

Получаем:

Время резервного копирования определяется интенсивностью отказов, сопровождающихся потерей данных

**4 РАЗРАБОТКА КЛИЕНТСКОГО ПРИЛОЖЕНИЯ**

**4.1 Обоснования выбора языка программирования**

В качестве языка программирования был выбран PHP: исполняемый на стороне сервера скрипт. Его плюсами является простота внедрения в html страницы, простота кода, ранее использовался для написания дополнительного задания в лабораторных работах по дисциплине «Технологии баз данных», легкость подключения к БД и оперирования данными с помощью html запросов.

**4.2 Разработка интерфейса пользователя**

На странице «Гости» (рисунок 12) отображается список гостей, находящихся в отеле, выселившихся за прошлый месяц и список всех гостей. Так же сверху панель с добавлением информации о новом постояльце: Имя, паспортные данные, телефон, дата въезда, выезда и номер заселения.

// Скрины сайта

**4.3 Алгоритм работы каждого из модулей**

Страница «Гости»

Во время загрузки страницы выполняется запрос к базе данных и в форму добавления клиента загружаются свободные номера. После в таблицы «В отеле», «Выселившиеся» загружаются данные из таблиц “Booking” и “Guests”.

После добавления информации о новом пользователе и нажатия на кнопку «Добавить», производится запрос на сервер и в базу добавляется информация о новом постояльце.

Страница «Номера»

На странице номер при загрузке страницы со стороны сервера выполнятся запрос к базе, выводятся 2 таблицы: занятые и свободные номера. Если в нижней форме ввести данные о номере и нажать «Добавить», выполняется запрос на сервер и в таблицу “Rooms” добавляется запись о новом номере.

Страница «Услуги»

После выбора в форме «Применить услугу» значений номера и вида услуги, введения даты и нажатия на кнопку «применить», выполняется запрос на сервер и в таблицу “Order” добавляется новая запись. По аналогии с гостями выполняется и редактирование услуг.

**4.4 Тестирование работы приложения**

Тестирование проводилось путем добавления, удаления, редактирования данных в формах и таблицах.

// Скрины таблицы на сайте

В результате тестирования приложения не было выявлено дефектов, вызывающих нарушение структуры базы данных, приложение выполняет необходимые действия по добавлению, изменению, удалению данных в таблицах.

**ЗАКЛЮЧЕНИЕ**

В данном курсовом проекте была спроектирована реляционная база данных на тему «База данных мини-отеля».

Для выбранной предметной области был проведен анализ, построена логическая модель и проведена нормализация, которая позволяет решить проблемы рационального выбора вариантов схем отношений из возможного множества альтернативных решений. В полученной в результате нормализации логической схеме содержатся следующие таблицы: гости, номера, услуги, заказы, бронирование.

Так же разработано приложение для работы с базой на PHP. В нем можно выполнять введение, изменение, удаление и отображение данных.

Были закреплены навыки организации баз данных, разработки программного обеспечения для него и обеспечения целостности.

**СПИСОК ЛИТЕРАТУРЫ И ИНФОРМАЦИОННЫХ РЕСУРСОВ**

1. Гарсиа-Молина Г. Системы баз данных. Полный курс: Пер. с англ./Г.Гарсиа-Молина, Д.Д.Ульман, Д.Уидом. – М.: Издательский дом «Вильямс», 2004. – 1088с.

2. Дейт К.Дж. Введение в системы баз данных, 7-е издание: Пер. с англ./ К.Дж.Дейт. – М.: Издательский дом «Вильямс», 2002. – 1072с.

3. Ульман Д. Основы систем баз данных/Д.Ульман. – М.: Финансы и статистика, 1983. – 334 с.

**ПРИЛОЖЕНИЕ А**

**SQL скрипт создания базы данных**

CREATE TABLE выдача\_книг (

id\_выдачи int NOT NULL AUTO\_INCREMENT,

id\_книги int NOT NULL,

id\_читателя int NOT NULL,

дата\_выдачи date NOT NULL,

дата\_возврата date NOT NULL,

PRIMARY KEY(id\_выдачи)

);

ALTER TABLE выдача\_книг ADD FOREIGN KEY(id\_читателя) REFERENCES читатели(id\_читателя) ON DELETE CASCADE ON UPDATE CASCADE;

ALTER TABLE выдача\_книг ADD FOREIGN KEY(id\_книги) REFERENCES книги(id\_книги) ON DELETE CASCADE ON UPDATE CASCADE;

CREATE TABLE читатели (

id\_читателя int NOT NULL AUTO\_INCREMENT,

фио varchar(75) NOT NULL,

дата\_рождения date NOT NULL,

пол enum("мужчина", "женщина") NOT NULL,

телефон varchar(14) NOT NULL,

адрес varchar(100) NOT NULL,

PRIMARY KEY(id\_читателя)

);

CREATE TABLE книги (

id\_книги int NOT NULL AUTO\_INCREMENT,

название varchar(200) NOT NULL,

id\_направления int NOT NULL,

id\_издательства int NOT NULL,

год\_издания date NOT NULL,

количество\_страниц int NOT NULL,

PRIMARY KEY(id\_книги)

);

ALTER TABLE книги ADD FOREIGN KEY(id\_направления) REFERENCES направления(id\_направления) ON DELETE CASCADE ON UPDATE CASCADE;

ALTER TABLE книги ADD FOREIGN KEY(id\_издательства) REFERENCES издательства(id\_издательства) ON DELETE CASCADE ON UPDATE CASCADE;

CREATE TABLE направления (

id\_направления int NOT NULL AUTO\_INCREMENT,

описание varchar(200) NOT NULL,

PRIMARY KEY(id\_направления)

);

CREATE TABLE издательства (

id\_издательства int NOT NULL AUTO\_INCREMENT,

название varchar(150) NOT NULL,

адрес varchar(100) NOT NULL,

PRIMARY KEY(id\_издательства)

);

CREATE TABLE авторы (

id\_автора int NOT NULL AUTO\_INCREMENT,

фио varchar(150) NOT NULL,

учёная\_степень enum('нет','кандидат наук','доктор наук') NOT NULL,

дата\_рождения date NOT NULL,

PRIMARY KEY(id\_автора)

);

CREATE TABLE авторы\_книги (

id\_автора\_книги int NOT NULL AUTO\_INCREMENT,

id\_книги int NOT NULL,

id\_автора int NOT NULL,

PRIMARY KEY(id\_автора\_книги)

);

ALTER TABLE авторы\_книги ADD FOREIGN KEY(id\_книги) REFERENCES книги(id\_книги) ON DELETE CASCADE ON UPDATE CASCADE;

ALTER TABLE авторы\_книги ADD FOREIGN KEY(id\_автора) REFERENCES авторы(id\_автора) ON DELETE CASCADE ON UPDATE CASCADE;

**ПРИЛОЖЕНИЕ Б**

**Код клиентского приложения**

Код