Министерство образования Республики Беларусь Учреждение образования «Белорусский государственный

университет информатики и радиоэлектроники»

Факультет компьютерного проектирования

Кафедра проектирования информационно-компьютерных систем Дисциплина «Структуры и базы данных»

|  |
| --- |
| «К ЗАЩИТЕ ДОПУСТИТЬ» |
| Руководитель курсового проекта Магистр технических наук, ассистент  А.Д. Сыс |
| . .2021 |

## ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовому проекту на тему:

## «БАЗА ДАННЫХ ДЛЯ ПОДДЕРЖКИ СЕРВИСА ПО РЕМОНТУ ОРГТЕХНИКИ»

БГУИР КП 1-39 03 02 007 ПЗ

Выполнил студент группы 913831 Хасанбоев Зиёдбек Улугбек угли

(подпись студента)

Курсовой проект представлен на проверку .\_ 2021

(подпись студента)

Минск 2021

БГУИР КП 1-39 03 02 007 ПЗ

# РЕФЕРАТ

**Хасанбоев З.У** База данных для поддержки сервиса по ремонту оргтехники: пояснительная записка к курсовому проекту / **З.У. Хасанбоев** – Минск: БГУИР, 2021. – ? с.

Пояснительная записка с.38 рис.4 табл.4, 6 источников, 3 приложений

АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ И ЕЕ ФОРМАЛИЗАЦИЯ ДЛЯ ПРОЕКТИРОВАНИЯ БАЗЫ ДАННЫХ, ПРОЕКТИРОВАНИЕ БАЗЫ ДАННЫХ ДЛЯ ОСНОВНОГО ВИДА ДЕЯТЕЛЬНОСТИ РАССМАТРИВАЕМОЙ ОБЛАСТИ, ПРИМЕНЕНИЕ РАЗРАБОТАННОЙ БАЗЫ ДАННЫХ

*Цель проектирования:* разработка приложения для поддержки работы сервиса по ремонту оргтехники. Целью разработки приложения являются упрощение и ускорение процесса обработки данных сотрудников для сотрудников отдела кадров.

Для достижения цели работы должны быть решены следующие задачи:

* осуществить постановку задачи, выделить требования к приложению;
* изучить современные средства разработки приложений;
* определить требования и спроектировать приложение;
* реализовать и протестировать приложение.

*Методология проведения работы:* в процессе решения поставленных задач использованы принципы проектирования баз данных, принципы объектно-ориентированного программирования, паттерны проектирования.

*Результаты работы:* спроектирована база данных; написано приложение с использованием базы данных; разработана графическая часть проекта.

*Область применения результатов:* могут быть использованы в работе отделе кадров.

# СОДЕРЖАНИЕ

[Введение](file:///C:\Users\user\Downloads\Баходиров%20Б%20КУРСАВОЙ%201.docx#_bookmark0) 6

1. [Анализ предметной области и ее формализация](file:///C:\Users\user\Downloads\Баходиров%20Б%20КУРСАВОЙ%201.docx#_bookmark1)

[для проектирования базы анных 8](file:///C:\Users\user\Downloads\Баходиров%20Б%20КУРСАВОЙ%201.docx#_bookmark2)

* 1. [Описание предметной области 8](file:///C:\Users\user\Downloads\Баходиров%20Б%20КУРСАВОЙ%201.docx#_bookmark3)
  2. Назначения и область применения программы 9
  3. [Определение требований и ограничений к базе данныхс точки зрения](file:///C:\Users\user\Downloads\Баходиров%20Б%20КУРСАВОЙ%201.docx#_bookmark4) [предметной области 9](file:///C:\Users\user\Downloads\Баходиров%20Б%20КУРСАВОЙ%201.docx#_bookmark4)
  4. [Постановка решаемой задачи 10](file:///C:\Users\user\Downloads\Баходиров%20Б%20КУРСАВОЙ%201.docx#_bookmark5)

1. [Проектирование базы данных для основного вида деятельности рассматриваемой предметной области 10](file:///C:\Users\user\Downloads\Баходиров%20Б%20КУРСАВОЙ%201.docx#_TOC_250006)
   1. Проектирование базы данных для основного вида деятельности рассматриваемой предметной области 11
   2. Выбор и обоснование используемых типов, данных и ограничений

(доменов) 12

* 1. [Проектирование запросов к базе данных 15](file:///C:\Users\user\Downloads\Баходиров%20Б%20КУРСАВОЙ%201.docx#_TOC_250005)
  2. [Программная реализация и документирование базы данных 16](file:///C:\Users\user\Downloads\Баходиров%20Б%20КУРСАВОЙ%201.docx#_TOC_250004)
  3. [Проектирование разрабатываемого сайта 17](file:///C:\Users\user\Downloads\Баходиров%20Б%20КУРСАВОЙ%201.docx#_TOC_250003)

1. [Применение разработанной базы данных 17](file:///C:\Users\user\Downloads\Баходиров%20Б%20КУРСАВОЙ%201.docx#_TOC_250002)
   1. [Руководство пользователя 17](file:///C:\Users\user\Downloads\Баходиров%20Б%20КУРСАВОЙ%201.docx#_TOC_250001)
   2. Обоснование и реализация механизма обеспечения безопасности

и сохранности данных 19

[Заключение 23](file:///C:\Users\user\Downloads\Баходиров%20Б%20КУРСАВОЙ%201.docx#_TOC_250000)

Список использованных литературы 24

Приложение А (обязательное) Скрипт генерации базы данных 25

Приложение Б (обязательное) Листинг НTML кода 31

Приложение В (обязательное)Листинг СSS кода… 32

**ПЕРЕЧЕНЬ УСЛОВНЫХ ОБОЗНАЧЕНИЙ, СИМВОЛОВ И ТЕРМИНОВ**

|  |  |
| --- | --- |
| БД | база данных |
| НФ | нормальная форма |
| ПО | программное обеспечение |
| СУБД | система управления базами данных |
| 1НФ | первая нормальная форма |
| 2НФ | вторая нормальная форма |
| 3НФ | третья нормальная форма |
| *ER* | *Entity-Relationship model* , модель «сущность — связь» |
| *MySQL* | компактная [встраиваемая СУБД](https://ru.wikipedia.org/wiki/%D0%92%D1%81%D1%82%D1%80%D0%B0%D0%B8%D0%B2%D0%B0%D0%B5%D0%BC%D0%B0%D1%8F_%D0%A1%D0%A3%D0%91%D0%94) |

# ВВЕДЕНИЕ

Автоматизированные системы являются неотъемлемой частью любого бизнеса и производства. Практически все управленческие и технологические процессы в той или иной степени используют средства вычислительной техники. Сегодня персональные компьютеры устанавливают на каждом рабочем месте, и никто не сомневается в их необходимости. Значительные объемы средств вычислительной техники и их особая роль в функционировании любого предприятия ставят перед руководством целый ряд новых задач. Бухгалтер или экономист, помогающий руководителю управлять имуществом предприятия, склонен рассматривать вычислительную технику просто как вид основных средств, не зная о том, что незначительное (с его точки зрения) изменение характеристик этого вида основных средств может кардинально отразиться на прибыли предприятия Конфигурация компьютеров изменяется на современном предприятии почти каждый день и проведение “вручную“ ежедневной ревизии требует немыслимых накладных расходов. Поэтому эффективное управление средствами вычислительной техники может быть только автоматизированным.  
 В основе управления средствами вычислительной техники лежит автоматический учет. Автоматический учет средств вычислительной техники заключается в диагностике, сборе и хранении информации о характеристиках компьютеров и периферии. Задачей данного курсового проекта является автоматизации учёта по ремонту и обслуживанию компьютерной и офисной техники организации. Программа учет вычислительной техники и орг. техники организации предназначена для автоматизации процесса диагностики, сбора и хранении информации о характеристиках компьютеров и периферийных устройств.

Решение данной задачи должно обеспечить:

* систематизацию информации по организационной технике;
* осуществление расчета амортизации организационной технике;
* сбор статистической информации;
* осуществление контроля по внутреннему перемещению техники;
* формирование необходимых выходных ведомостей;
* надежное хранение данных.

# АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ И ЕЕ

**ФОРМАЛИЗАЦИЯ ДЛЯ ПРОЕКТИРОВАНИЯ БАЗЫ ДАННЫХ**

Отдел кадров – это структура в организации, которая занимается управлением персоналом в организации, то есть деятельностью людей, выполняющих на предприятии или в организации функции, способствующие наиболее эффективному использованию человеческих ресурсов для достижения первичных целей предприятии (организации).

Цель отдела кадров – это способствование достижению целей предприятия (организации) путем обеспечения ее необходимыми кадрами и эффективного использования их квалификации, опыта, мастерства, работоспособности, творческого потенциала

Основные функции по работе отдела кадров предприятия:

* Принятие на работу.
* Расчет с работы.

Обучения, продвижения квалификации и переквалификации персонала:

* Планирования карьеры и продвижения.
* Нормирования труда, оценки резервов и аттестации.

## Описание предметной области

Решено, что база данных создается исключительно для работников отдела кадров. Сотрудники отдела кадров сможет получать информацию о сотрудниках, организация отбора, набора и найма персонала, необходимой квалификации и в требуемом объеме. Создание эффективной системы штатных сотрудников Исходя из этого в проектируемой базе данных были выделены следующие сущности: «Должность», «Сотрудник», «Отдел» «Вуз»

Между сущностями будут реализованы связи.

Таким образом, база данных будет ориентирована на функционал, и можно использоваться на работе.

## Назначения и область перемещения программы

Необходимо разработать программный комплекс для ведения учета по кадровому составу предприятия. В круг задач, выполняемых программным комплексом должно входить следующее:

Ввод данных по сотрудникам;

* принятия на работу
* поиск по данных;
* редактирование информации;
* удаление данных;
* вывод данных на печать.

Программа может быть использована для ведения учета по кадровому составу на предприятиях, в организациях, фирмах и т.п., где необходимо сохранять и обрабатывать данные о сотрудниках, работающих в данном предприятии.

## Определение требований и ограничений к базе данных с точки зрения предметной области

Особенности каждой предметной области могут накладывать разнообразные требования и ограничения на хранимые в базе данные. Для проектируемой базы данных следующие ограничения:

* требование уникальности записей в определенных атрибутах, то есть номер мобильного телефона не должен повторяться, тогда как другая информация может быть одинаковой;
* ограничения данных от других от других сотрудников;
* проставление отметок в личное дело и трудовую книжку;
* при наличии приказа об увольнении в личное дело вносится соответствующая отметка с указанием статьи увольнения и даты расчета, а также номер приказа об увольнении. В трудовую книжку при этом также ставится отметка о расчете;
* выдача документов.

## Постановка решаемой задачи

Целью данного курсового проекта является создание базы данных для поддержки работы отдела кадров. Работа с базой данных осуществляется с помощью веб сайта.

Первым этапом будет проектирование базы данных, что подразумевает под собой создание всех таблиц с требуемыми связями, полей с учетом формата вносимых данных. С помощью базы данных будут решаться такие задачи, как регистрация и авторизация пользователя, поиск сотрудников по заданным параметрам, поиск по разным параметром и распечатка, редактирование личных данных.

За созданием базы данных следует написание сайта с использованием языка программирования *HTML, CSS.* Данный процесс включает разработку пользовательского интерфейса, удобного для использования, позволяющего решать поставленные задачи.

## ДЕЯТЕЛЬНОСТИ РАССМАТРИВАЕМОЙ ПРЕДМЕТНОЙ ОБЛАСТИ

Качество проектирования базы данных может влиять на работу сайта, с хорошо спроектированной базой данных легче работать и писать запросы. Для качественного проектирования существуют различные подходы:

* Требование на принадлежность данных (кадров, определение и формирование источников удовлетворения кадровой потребности – вопросы подбора, расстановки, подготовки и повышения квалификации кадров).
* Выделение сущностей и их атрибутов, которые будут храниться в базе данных, и формирование по ним таблиц;
* Определение уникальных идентификаторов объектов, которые хранятся в строках таблицы;
* Определение отношений между таблицами с помощью внешних ключей;
* Нормализация базы данных.

На первом этапе происходит выделение сущностей. Каждая таблица в базе данных должна представлять одну сущность. У каждой сущности определяют набор атрибутов. Каждый столбец должен хранить один атрибут сущности. А каждая строка представляет отдельный объект или экземпляр сущности.

## Разработка инфологической модели предметной области базы данных

В данном курсовом проекте используется *MySQL* – это компактная встраиваемая СУБД. Слово «встраиваемый» означает, что *MySQL e* не использует парадигму клиент-сервер. Такой подход уменьшает накладные расходы, время отклика и упрощает программу. *MySQL* хранит всю базу данных в единственном стандартном файле, поэтому несколько процессов или потоков могут одновременно читать данные из одной базы. Запись в базу можно осуществить только в том случае, если никаких других запросов в данный момент не обслуживается; в противном случае попытка записи оканчивается неудачей, и в программу возвращается код ошибки.

Нормальная форма – это способ избавиться от избыточности информации и некоторых проблем, связанных с обработкой данных.

Для того, чтобы привести таблицу к первой нормальной форме необходимо, чтобы значения в каждом поле были автономным. Помимо этого, строки таблицы не должны зависеть друг от друга, аналогичная ситуация со столбцами записей, то есть их порядок не должен влиять на понимание

информации. Каждая строка должна быть уникальная, поэтому для нее определяется первичный ключ.

Таблица находится в третьей нормальной форме, если любой ее атрибут, не входящий в состав первичного ключа, функционально полно зависит от первичного ключа. Функционально полно означает, что атрибут зависит от всего первичного ключа, но не зависит от его какой-либо части.

Далее необходимо привести таблицы к третьей нормальной форме, для этого любой не ключевой атрибут зависит от первичного ключа.

В результате приведения таблиц к третьей нормальной форме появляются дополнительные таблицы.

На рисунке 2.1 показана база данных, приведенная к 3 нормальной форме.

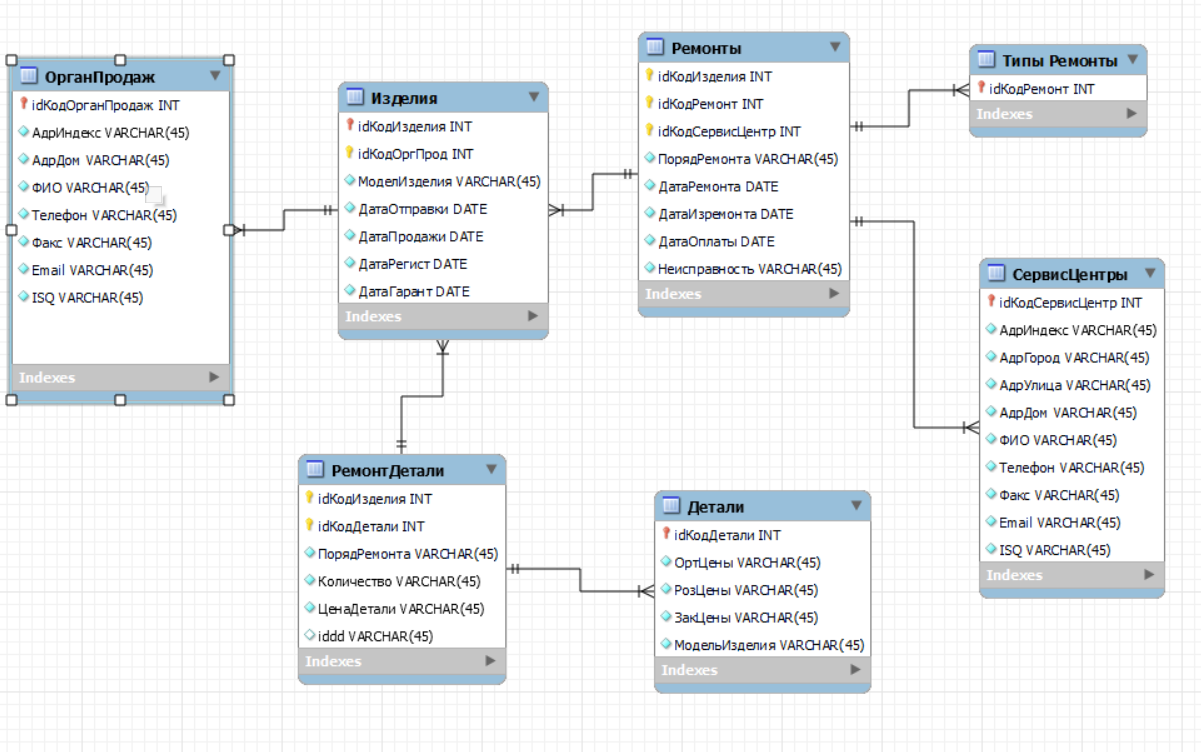


Рисунок 2.1 – Представление

сущностей проектируемой базы данных

Таким образом, база данных будет ориентирована на функционал, и удобство в ползание.

## Выбор и обоснование используемых типов, данных и ограничений (доменов)

Тип данных *MySQL* – это атрибут, определяющий тип данных любого объекта. Каждый столбец, переменная и выражение имеет связанный тип данных в My*SQL*.

Каждое значение, хранящееся в базе данных My*SQL*, имеет один из следующих классов хранения, представленных в таблице 2.1.

Таблица 2.1 – Классы хранения данных

|  |  |
| --- | --- |
| **Название** | **Описани**  **е** |
| *NOT NULL* | Значение – значение не может равняться *NULL*. |
| *INTEGER* | Значение представляет собой целое число со знаком, сохраненное в 1, 2, 3, 4, 6 и 8 битах в зависимости от  величины значения. |
| *TEXT* | Значение представляет собой строку, хранящуюся с использованием кодировки базы данных. |

Класс хранения – более общее понятие, чем тип данных. К примеру, класс хранения *INTEGER* включает 6 различных типов целочисленных данных различной длины.

*MySQL* не имеет отдельного логического класса хранения. Вместо этого логические значения хранятся как целые числа 0 (*false*) и 1 (*true*).

*MySQL* не имеет отдельного класса хранения дат и времени, но способен хранить даты и время как значения *VARCHAR*, или *DATE* Описание способов хранения даты и времени представлено в таблице 2.2.

Таблица 2.2 – Хранение даты и времени

|  |  |
| --- | --- |
| **Название** | **Опи сание** |
| *VARCHAR* | Дата в формате «*YYYY-MM-DD HH:MM:SS.SSS*» |
| *DATE* | Количество секунд с 1970-01-01 00:00:00 *UTC* |

В таблице 2.3 представлены типы данных полей для каждой таблицы.

Таблица 2.3 – Организация продаж

|  |  |
| --- | --- |
| **Поле** | **Тип данных** |
| *idОрганПродаж* | *INT* |
| *АдрИндекс* | *Varchar(45)* |
| *АдрДом* | *Varchar(45)* |
| *ФИО* | *Varchar(45)* |
| *Телефон* | *Varchar(45)* |
| *Факс* | *Varchar(45)* |

Продолжение таблицы 2.3

|  |  |
| --- | --- |
| **Таблица Изделия** | |
| *IdИзделия* | *INT* |
| *idКодОргПрод* | *INT* |
| *МоделИзделия* | *Varchar(45)* |
| *ДатаОтправки* | *DATE)* |
| *ДатаПродажи* | *DATE* |
| *ДатаРегист* | *DATE* |
| *ДатаГарант* | *DATE* |
| **Таблица *Ремонты*** | |
| *idКодИзделия* | *INT* |
| *idКодРемонт* | *INT* |
| *idКодСервисЦентр* | *INT* |
| *ПорядРемонта* | *Varchar(45)* |
| *ДатаРемонта* | *DATE* |
| *ДатаИзремонта* | *DATE* |
| *ДатаОплаты* | *DATE* |
| **Таблица Детали** | |
| *idКодДетали* | *INT* |
| *ОртЦены* | *Varchar(45)* |
| *РозЦены* | *Varchar(45)* |
| *ЗакЦены* | *Varchar(45)* |
| *МодельИзделия* | *Varchar(45)* |

|  |  |
| --- | --- |
| **Таблица РемонтДетали** | |
| **Поле Тип данных** | |
| *idКодИзделия* | *INT* |
| *idКодДетали* | *INT* |
| *ПорядРемонта* |  |
| *Количество* |  |
| *ЦенаДетали* |  |

Преимуществом My*SQL* является файловая структура, т. е. вся база данных состоит из одного файла. В процессе разработки приложений часто появляется необходимость масштабирования и My*SQL* предлагает все, что необходимо для этих целей, так как состоит всего из одного файла и библиотеки, написанной на языке *C*.

## Проектирование запросов к базе данных

MySQL представляет собой наиболее популярную и распространенную СУБД для сети интернет. Данная СУБД не предназначена для того, чтобы работать с большим количеством информации, но ее возможности позволяют справляться со всеми задачами, необходимыми для работы интернет сайтов как скромных, так и ресурсов приличных размеров.

Чтобы обращаться и собственно работать с БД MySQL существуют определенные команды, которые обычно обозначают как запросы MySQL. Именно с помощью запросов можно выполнять такие манипуляции с базой как удаление, сортировка, изменение, добавление, получение и т.д.

Все существующие команды MySQL можно разделить на простые или легкие для понимания и запоминания и сложные запросы.

Простые запросы MySQL — это команды, для исполнения которых достаточно одной таблицы, так как они производят манипуляции только с одной таблицей из базы.

Сложные запросы MySQL — это команды, которые могут затрагивать одновременно две или даже больше таблиц из одной базы данных.

Чтобы более детально разобраться, в чем же заключается различия между запросами — давайте рассмотрим каждый вид команд по отдельности. Простые MySQL запросы Существует два запроса, которые можно назвать максимально или супер простыми. «*Show databases;*» — данный запрос выведет на экран список всех имеющихся баз данных. «*Show tables in bd;*» — данная команда позволит узнать количество и наименования всех таблиц, которые есть в базе данных под названием bd. Ну а теперь от супер простых запросов перейдем просто к простым. Запросы MySQL типа Select Зная значение слова select, можно понять, что все команды, связанные с данным словом, будут направлены на выбор и просмотр информации хранящейся только в одной таблице принадлежащей к базе.

«*SELECT count(\*) FROM cart;*» — данный запрос позволит узнать, сколько записей содержится в упомянутой таблице:

Для получения информации о сотруднике выполняется следующий запрос:

*INSERT INTO orders (Id\_отдел, название) values();.*

После добавления сотрудника можно просмотреть более подробную информации сотруднике: *SELECT p.образование, вуз, должности, from orders o inner join place p on o.place\_id = p.id inner join shows s on o.specification\_id = s.id inner join theatre t on s.theatre\_id = t.id;*

Для отображения информации о пользователе необходимо использовать следующий запрос: *SELECT образование, вуз, должности,.* Для изменения информации о пользователе: *UPDATE customer set имя = “”, фамилия = “”, телефон = “”, адрес = “”. Дата рождения = “”.[1]*

## Программная реализация и документирование базы данных

Для написания сайта и организации взаимодействия с базой данных требуется выбрать HTML, CSS JS.

В данном курсовом проекте для написания сайта был выбран HTML CSS язык программирования *JS*, являющимся одним из самых распространенных и популярных языков программирования. *Java Script* является объектно- ориентированным языком. Он поддерживает полиморфизм, наследование, статическую типизацию. Объектно-ориентированный подход позволяет решить задачи по построению крупных, но в тоже время гибких, масштабируемых и расширяемых приложений.

Программная реализация базы данных осуществлена с использованием библиотеки *JQUERY*. Библиотека jQuery помогает легко получать доступ к любому элементу DOM, обращаться к атрибутам и содержимому элементов DOM, манипулировать ими. Также библиотека jQuery предоставляет удобный API для работы с AJAX. Библиотека обеспечивает уровень абстракции над My*SQ*, чтобы обеспечить более надежный доступ к базе данных. Библиотека помогает создать кеш данных приложения на устройстве, на котором запущено сайт. MySQL – это система управления реляционными базами данных. В реляционной базе данные хранятся в отдельных таблицах, благодаря чему достигается выигрыш в скорости и гибкости. Таблицы связываются между собой при помощи отношений, благодаря чему обеспечивается возможность объединять при выполнении запроса данные из нескольких таблиц. SQL как часть системы MySQL можно охарактеризовать как язык структурированных запросов плюс наиболее распространенный стандартный язык, используемый для доступа к базам данных. Программное обеспечение MySQL – это ПО с открытым кодом. ПО с открытым кодом означает, что применять и модифицировать его может любой желающий. Такое ПО можно получать по Internet и использовать бесплатно. При этом каждый пользователь может изучить исходный код и изменить его в соответствии со своими потребностями**.**

## Проектирование разрабатываемого сайта

Для написания сайта и организации взаимодействия с базой данных требуется выбрать язык программирования.

В данном курсовом проекте для написания сайта был выбран язык программирования HTML CSS *Java Script*, являющимся одним из самых распространенных и популярных языков программирования. *Java Script* является объектно-ориентированным языком. Он поддерживает полиморфизм, наследование, статическую типизацию. Объектно-ориентированный подход позволяет решить задачи по построению крупных, но в тоже время гибких, масштабируемых и расширяемых приложений.

Механизм использования сокетов подразумевает технологию клиент- сервер, а это означает, что в системе должна быть запущена специальная программа MySQL-сервер, которая принимает и обрабатывает запросы от

программ. Так как вся работа происходит в действительности на одной машине, накладные расходы по работе с сетевыми средствами незначительны (установка и поддержание соединения с MySQL-сервером обходится довольно дешево).

Структура MySQL трехуровневая: базы данных -- таблицы -- записи. Базы данных и таблицы MySQL физически представляются файлами с расширениями frm, MYD, MYI. Логически – таблица представляет собой совокупность записей. А записи – это совокупность полей разного типа. Имя базы данных MySQL уникально в пределах системы, а таблицы – в пределах базы данных, поля – в пределах таблицы. Один сервер MySQL может поддерживать сразу несколько баз данных, доступ к которым может разграничиваться логином и паролем. Зная эти логин и пароль, можно работать с конкретной базой данных. Например, можно создать или удалить в ней таблицу, добавить записи и т. д. Обычно имя-идентификатор и пароль назначаются хостинг провайдерами, которые и обеспечивают поддержку MySQL для своих пользователей.

## ПРИМЕНЕНИЕ РАЗРАБОТАННОЙ БАЗЫ ДАННЫХ

## Руководство пользователя

Главная страница сайта появляется при запуске и должна объединять все важнейшие функции управления. Из нее же запускаются все страницы. На верней частье сайта установлены быстрые силки как: «Главная», «О нас», «Контакты».

Рисунок 3.1 – Главная страница

На рисунке 3.1 представлена главная страница сайта «Service-Centre

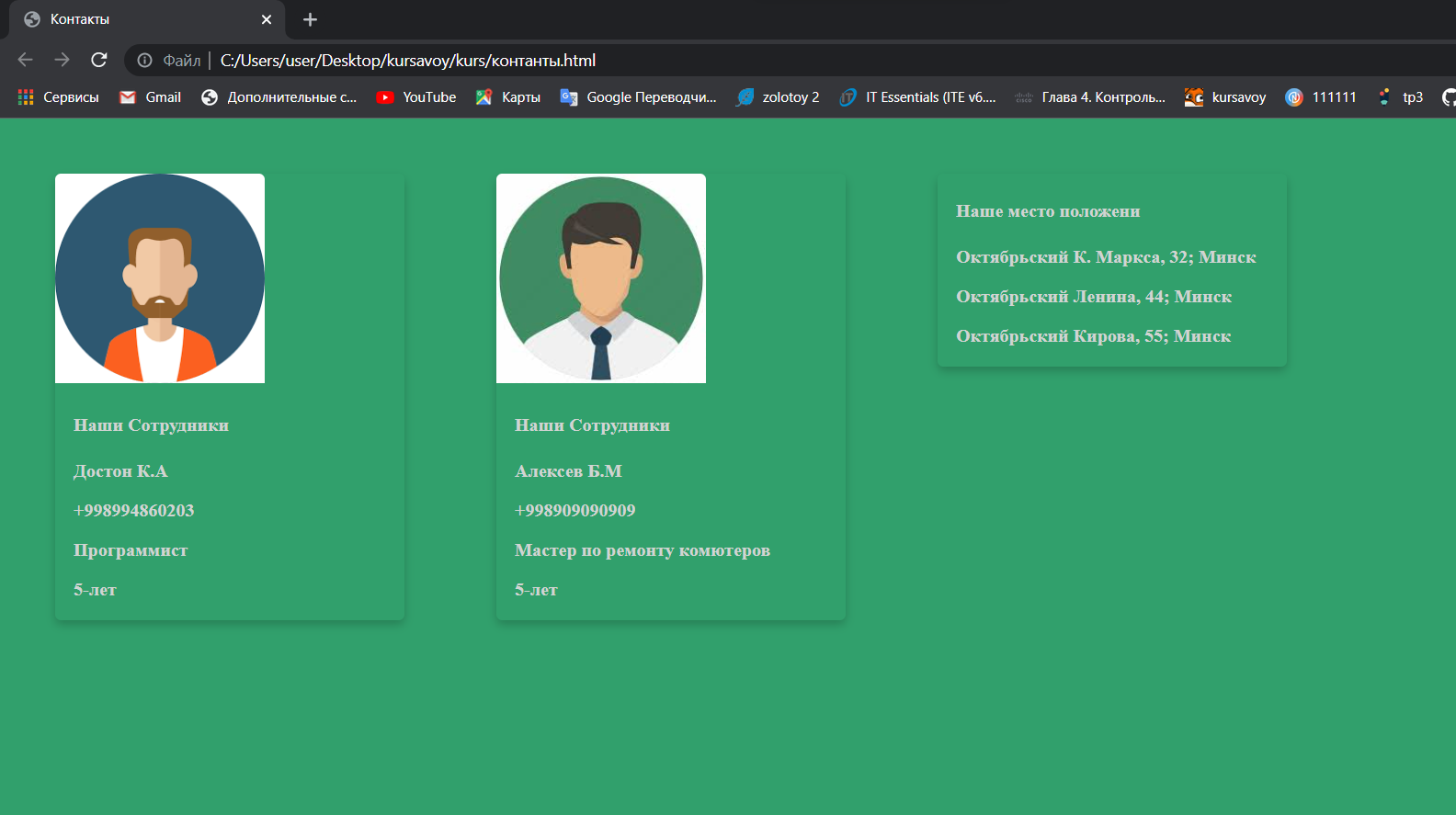


Рисунок 3.2

На рисунке 3.1 представлена контакты страница сайта

## Администрирования базы данных

Современные базы данных – это сложные многофункциональные программные системы, работающие в открытой распределенной среде. Они уже сегодня доступны для использования в деловой сфере и выступают не просто в качестве технических и научных решений, но как завершенные продукты, предоставляющие разработчикам мощные средства управления данными и богатый инструментарий для создания прикладных программ и систем.

Администрирование базами данных предусматривает выполнение функций, направленных на обеспечение надежного и эффективного функционирования системы баз данных, адекватности содержания базы данных информационным потребностям пользователей, отображения в базе данных актуального состояния предметной области. и. Администратор базы данных – это: – управляющий данными, а не хозяин;

– системный программист определенного профиля, а также эксперт высшего уровня, обеспечивающий службу эксплуатации решениями по 21 процедурам и регламентам работы;

– лицо, принимающее окончательное решение в своей области, и человек, обладающий способностями к общению, совместному планированию и компромиссам Надежность и достоверность – это ключевые понятия в деятельности администратора базы данных. Он должен уметь вести тщательное документирование всех действий по управлению базой данных.

Существует несколько видов администраторов БД, а их обязанности вполне могут отличаться от компании к компании. Вот характеристики некоторых типов АБД и занимаемых ими положений:

Оперативные (operational) Администраторов БД:

* манипулируют дисковым пространством
* наблюдают за текущей производительностью системы
* реагируют на возникающие неисправности БД
* обновляют системное ПО и ПО базы данных
* контролируют структурные изменения БД
* запускают процедуры резервного копирования данных
* выполняют восстановление данных
* создают и управляют тестовыми конфигурациями БД

*Прикладные (application)* Администраторов БД*:*

* в курсе информационных нужд компании
* помогают в разработке прикладных задач
* отвечают за разработку схемы и ее изменения
* вместе с системным АБД обеспечивают должный уровень – резервирования/ восстановления данных
* занимаются построением тестовых БД

*Системные (system) АБД:*

* отвечают за все необходимое для резервирования и восстановления данных
* контролируют производительность системы в целом
* осуществляют поиск и устранение неисправностей
* в курсе нынешних и будущих потребностей БД в плане емкости
* в курсе текущего состояния и нужд БД

*Наемные (contract) АБД :*

* приглашаются под конкретную задачу или в качестве консультантов
* передают персоналу необходимые знания
* фиксируют свои действия!
* должны прекрасно разбираться в соответствующей области
* хороши в качестве временного персонала, для оценки проекта или системы

обеспечение безопасности баз данных MySQL является серьезной задачей.

Стандартные настройки

Поскольку базы данных под управлением MySQL широко распространены, хакеры уже успели изучить все прорехи, существующие в стандартной системе защиты.

Основными узкими местами признаются:

* инсталляция, параметры которой задаются по умолчанию;
* применение пустого пароля, также по умолчанию;
* возможность переполнения буфера.

Эти три зоны риска, создающие уязвимость при реализации целевых атак, должны быть защищены.

Предпосылки и возможности обеспечения безопасности

Модель построения защиты базы данных должна строиться, исходя из стандартных настроек и возможностей ПО.

При ее разработке следует принять во внимание следующие предпосылки:

* к базам данных MySQL обращаются только приложения в формате PHP, установленные и запускаемые на том же сервере;
* управление базами осуществляется штатными средствами – mysqladmin, mysql, mysqldump;
* если требуется удаленное резервирование базы данных, применяется исключительно протокол SSH.

Условия для выстраивания систем защиты Уровни обеспечения безопасности баз данных MySQL могут быть различными.

Для реализации максимально возможного уровня защиты конфигурация системы должна выглядеть следующим образом:

* база данных MySQL реализуется только в chrooted-среде, это значит, что исключено взаимодействие с программами в других корневых каталогах;
* все процессы базы данных выполняются с уникальными значениями UID (User identifier) и GID (Group identifier) – числовых идентификаторов пользователя и группы, которые не используются ни в каких других процессах и программах;
* доступ к базе может быть только локальный, удаленный по возможности необходимо исключить;
* чем более сложный пароль используется для основной учетной записи, тем лучше;
* изначально заданная в свободной лицензии учетная запись администратора требует переименования;
* анонимный доступ к базе данных MySQL с использованием идентификатора nobody должен быть исключен;
* стандартные форматы баз данных и таблиц придется удалить.

При выполнении этих условий защита станет эффективнее нельзя давать им доступ к администрированию самой БД и ее утилит. Доступ с возможностью использования таких привилегий администратора, как FILE, GRANT, ALTER, SHOW DATABASE, RELOAD, SHUTDOWN, PROCESS, SUPER должен быть

исключен.

Для упрощения загрузки БД и во избежание необходимости каждый раз вручную вводить ряд команд, обеспечивающих ее работоспособность и связь с приложениями, необходимо прописать сценарий загрузки.

Итогом выполнения мер, призванных обеспечить высокий уровень защищенности БД, должно стать:

* ограничение прав администраторов;
* отключение внешних портов;
* запуск БД только в защищенной среде.

Эти меры позволят снизить уровень критических рисков, обезопасить базу данных от внешних вторжений и несанкционированного доступа внутренних пользователей

# ЗАКЛЮЧЕНИЕ

В ходе курсовой работы была разработана веб сайт и система хранения обработки информации о сотрудниках, Разработанный программный продукт имеет удобный интерфейс, интуитивно-понятные связи между диалогами ввода информации.

В итоге проделанной работы был создан готовый программный продукт и достигнуты основные цели разработки:

* уменьшение времени выполнения каждой функции;
* автоматическое создание документации и отчетов;
* автоматическое проставление дат и табельных номеров.

Которая способствует быстрому поиску необходимых данных при минимальных затратах времени, а также программная реализация системы, в основе которой лежит проектирование предметной области и логической схемы БД, являющейся информационным ресурсом разрабатываемой системы. информационно-справочная система реализует следующие функции:

* ввод данных в БД;
* поддержание целостности и безопасности данных в БД при их хранении;
* поддержание семантического смысла данных.

С помощью данной БД можно получить следующие, актуальные в данной работе сведения: о сотрудниках.

Разработанная система позволяет решать задачи поиска информации в БД по запросу пользователя и выдавать информацию в удобном для работы и наглядном для непрофессионала виде.

Современные базы данных являются основой многочисленных информационных систем. Информация, накопленная в них, является чрезвычайно ценным материалом, и в настоящий момент широко распространяются методы обработки баз данных с точки зрения извлечения из них дополнительных знаний, методов, которые связаны с обобщением и различными дополнительными способами обработки данных

## Список литературы

1. Майкл, Дж. Хернандес SQL – запросы для простых смертных. Практическое руководство по манипулированию данными в SQL / Майкл Дж. Хернандес, Джон Л. Вьескас. – М.: ЛОРИ, 2013.
2. Дронов, В.А PHP, MySQL, HTML5 и CSS 3. Разработка современных динамических Web-сайтов (pdf+epub) / В.А Дронов. – 2016 : БХВ-Петербург, 2016. – 206 с
3. Чаллавала, Ш MySQL 8 для больших данных / Ш Чаллавала, Дж Лакхатария, Ч Мехта. – 2020 : ДМК Пресс, 2020. – 156 с. Чаллавала, Ш MySQL 8 для больших данных / Ш Чаллавала, Дж Лакхатария, Ч Мехта. – 2020 : ДМК Пресс, 2020. – 156 с.CIT Forum URL: [http://citforum.ru/,](http://citforum.ru/) свободный доступ.
4. Википедия – свободная энциклопедия [Электронный ресурс]. – URL: https://ru.wikipedia.org/wiki/SQL, свободный доступ.
5. Дейт К. SQL и реляционная теория. Как грамотно писать код на SQL.

– Символ-Плюс, 2010. – С.123

1. Иванова, Г.С. "Основы программирования" Учебник для вузов. / Г.С. Иванова. – 2010 : МГТУ им. Н.Э. Баумана,, 2010. – 156 с.

## ПРИЛОЖЕНИЕ А

**(обязательное)**

## Скрипт генерации базы данных

-- MySQL Script generated by MySQL Workbench

-- Fri Dec 17 07:43:29 2021

-- Model: New Model Version: 1.0

-- 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 office

-- -----------------------------------------------------

-- -----------------------------------------------------

-- Schema office

-- -----------------------------------------------------

CREATE SCHEMA IF NOT EXISTS `office` DEFAULT CHARACTER SET utf8 ;

USE `office` ;

-- -----------------------------------------------------

-- Table `office`.`Ремонты`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `office`.`Ремонты` (

`idКодИзделия` INT NOT NULL,

`idКодРемонт` INT NOT NULL,

`idКодСервисЦентр` INT NOT NULL,

`ПорядРемонта` VARCHAR(45) NOT NULL,

`ДатаРемонта` DATE NOT NULL,

`ДатаИзремонта` DATE NOT NULL,

`ДатаОплаты` DATE NOT NULL,

`Неисправность` VARCHAR(45) NOT NULL,

PRIMARY KEY (`idКодИзделия`, `idКодРемонт`, `idКодСервисЦентр`))

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `office`.`РемонтДетали`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `office`.`РемонтДетали` (

`idКодИзделия` INT NOT NULL,

`idКодДетали` INT NOT NULL,

`ПорядРемонта` VARCHAR(45) NOT NULL,

`Количество` VARCHAR(45) NOT NULL,

`ЦенаДетали` VARCHAR(45) NOT NULL,

`iddd` VARCHAR(45) NULL,

PRIMARY KEY (`idКодИзделия`, `idКодДетали`))

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `office`.`Изделия`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `office`.`Изделия` (

`idКодИзделия` INT NOT NULL,

`idКодОргПрод` INT NOT NULL,

`МоделИзделия` VARCHAR(45) NOT NULL,

`ДатаОтправки` DATE NOT NULL,

`ДатаПродажи` DATE NOT NULL,

`ДатаРегист` DATE NOT NULL,

`ДатаГарант` DATE NOT NULL,

PRIMARY KEY (`idКодИзделия`, `idКодОргПрод`),

CONSTRAINT `1`

FOREIGN KEY (`idКодИзделия`)

REFERENCES `office`.`Ремонты` (`idКодИзделия`)

ON DELETE NO ACTION

ON UPDATE NO ACTION,

CONSTRAINT `2`

FOREIGN KEY (`idКодИзделия`)

REFERENCES `office`.`РемонтДетали` (`idКодИзделия`)

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `office`.`ОрганПродаж`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `office`.`ОрганПродаж` (

`idКодОрганПродаж` INT NOT NULL,

`АдрИндекс` VARCHAR(45) NOT NULL,

`АдрДом` VARCHAR(45) NOT NULL,

`ФИО` VARCHAR(45) NOT NULL,

`Телефон` VARCHAR(45) NOT NULL,

`Факс` VARCHAR(45) NOT NULL,

`Email` VARCHAR(45) NOT NULL,

`ISQ` VARCHAR(45) NOT NULL,

PRIMARY KEY (`idКодОрганПродаж`),

CONSTRAINT `123`

FOREIGN KEY (`idКодОрганПродаж`)

REFERENCES `office`.`Изделия` (`idКодОргПрод`)

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `office`.`Типы Ремонты`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `office`.`Типы Ремонты` (

`idКодРемонт` INT NOT NULL,

PRIMARY KEY (`idКодРемонт`),

CONSTRAINT `1`

FOREIGN KEY (`idКодРемонт`)

REFERENCES `office`.`Ремонты` (`idКодРемонт`)

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `office`.`Детали`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `office`.`Детали` (

`idКодДетали` INT NOT NULL,

`ОртЦены` VARCHAR(45) NOT NULL,

`РозЦены` VARCHAR(45) NOT NULL,

`ЗакЦены` VARCHAR(45) NOT NULL,

`МодельИзделия` VARCHAR(45) NOT NULL,

PRIMARY KEY (`idКодДетали`),

CONSTRAINT `1`

FOREIGN KEY (`idКодДетали`)

REFERENCES `office`.`РемонтДетали` (`idКодДетали`)

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `office`.`СервисЦентры`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `office`.`СервисЦентры` (

`idКодСервисЦентр` INT NOT NULL,

`АдрИндекс` VARCHAR(45) NOT NULL,

`АдрГород` VARCHAR(45) NOT NULL,

`АдрУлица` VARCHAR(45) NOT NULL,

`АдрДом` VARCHAR(45) NOT NULL,

`ФИО` VARCHAR(45) NOT NULL,

`Телефон` VARCHAR(45) NOT NULL,

`Факс` VARCHAR(45) NOT NULL,

`Email` VARCHAR(45) NOT NULL,

`ISQ` VARCHAR(45) NOT NULL,

PRIMARY KEY (`idКодСервисЦентр`),

CONSTRAINT `1`

FOREIGN KEY (`idКодСервисЦентр`)

REFERENCES `office`.`Ремонты` (`idКодСервисЦентр`)

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 Б

**(обязательное) Листинг НTML кода**

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta http-equiv="X-UA-Compatible" content="IE=edge">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Applicants</title>

    <link rel="stylesheet" href="style.css"

</head>

<body>

    <header>

        <nav class="navbar">

            <div class="container">

                <div class="navigation">

                    <div class="logo">

                        <a href="/Сервис-Центр.html">Сервис-Центр</a>

                    </div>

                    <div class="navbar-nav">

                        <a href="/Все-услуги.html">Все услуги</a>

                        <a href="/контанты.html">Контакты</a>

                        <a href="/vedim.html">О комрании</a>

                    </div>

            </div>

        </nav>

    </header>

</body>

</html>

## ПРИЛОЖЕНИЕ В

**(обязательное) Листинг СSS кода**

@import url('https://fonts.googleapis.com/css2?family=Playfair+Display:wght@400;500;700;900&family=Quicksand:wght@400;500;600&family=Roboto:wght@400;500;700;900&display=swap');

:root{

    --Playfair:  'Playfair Display', serif;

    --Quicksand: 'Quicksand', sans-serif;

    --Roboto: 'Roboto', sans-serif;

    --dark: #1a1d1a;

    --extraDark:#2b2b2b;

}

\*{

    padding: 0;

    margin: 0;

    box-sizing: border-box;

    font-family: 0;

}

   body {

        line-height: 0.2;

        color: var(--dark)

    }

    img{

        width: 100%;

        display: block;

    }

    .container{

        max-width: 1320px;

        margin: 0 auto;

        padding: 0 1.2rem;

    }

    header{

        height: 100vh;

        background-image: url(img/glavniy.jpg);

        background-position: center;

        background-size: cover;

        background-repeat: no-repeat;

        background-attachment: fixed;

    }

    .navbar{

        background: rgba(0, 255, 55, 0.6);

        padding: 1.2rem;

    }

    .navigation{

        display: flex;

        justify-content: space-between;

    }

    .navigation .logo a{

        color: rgb(0, 0, 0);

        font-size: 2rem;

        display: block;

        text-decoration: none;

        font-family: var(--Playfair);

        letter-spacing: 1px;

    }

    .navbar-nav{

        padding: 0.8rem 0 0.2;

    }

    .navbar-nav a{

        text-transform: uppercase;

        font-family: var(--Roboto);

        letter-spacing: 3px;

        font-weight: 500;

        color: rgb(0, 0, 0);

        text-decoration: none;

        display: inline-block;

        padding-right: 1.2rem;

        padding-left: 1.2rem;

    }

    .navbar-nav a{

        opacity: 0.8;

    }