Санкт-Петербургский политехнический университет Петра Великого
Институт компьютерных наук и технологий

Высшая школа программной инженерии

Работа допущена к защите

директора ВШ ПИ

П.Д. Дробинцев

«_____»_____20___г.

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА
БАКАЛАВРА

Разработка аналитического веб-приложения обработки звонков

по направлению «09.03.01 Информатика и вычислительная техника»
по образовательной программе
«09.03.01_09 Разработка программного обеспечения»

Выполнил студент гр. П.А. Храмцов

Руководитель к.т.н., доц. В.В. Шаляпин

Санкт-Петербург

2018 г.

Санкт-Петербургский политехнический университет Петра Великого

Институт компьютерных наук и технологий

Высшая школа программной инженерии

УТВЕРЖДАЮ

Директор ВШ ПИ

П.Д. Дробинцев

«_____»_____г.

З А Д А Н И Е
по выполнению выпускной квалификационной работы

Студенту группы 43504/23 Храмцову Павлу Антоновичу

Тема проекта (работы) Разработка аналитического веб-приложения обработки звонков

(утверждена распоряжением по ИКНТ от_____№_____)

2. Срок сдачи студентом оконченного проекта (работы)

3. Исходные данные к проекту (работе):

Call Data Record (CDR) – файл содержащий информацию по совершенным вызовам, где по каждому вызову содержится информация о начале, длительности, номера звонящего, набираемого номера, его уникальный идентификатор и другая информация.

4. Содержание расчетно-пояснительной записки (перечень подлежащих разработке вопросов).
Выбор технологий для создания веб-приложения.

Разработка системы базы данных.
Разработка серверного приложения на PHP.
Разработка клиентского приложения HTML и JavaScript.

5. Перечень графического материала (с точным указанием обязательных чертежей)

_____

_____

6. Консультанты по проекту (с указанием относящихся к ним разделов проекта, работы)

_____

_____

7. Дача выдачи задания

_____

Руководитель ВКР _____ В.В.Шаляпин

(подпись)

Задание принял к исполнению

_____

(дата)

Студент_____П.А. Храмцов

(подпись студента)

Оглавление

РЕФЕРАТ

На пятьдесят шесть страниц и девять рисунков.

Разработка аналитического веб-приложения обработки звонков

Данная бакалаврская работа посвящена разработке веб-приложения для анализа загрузки каналов клиентов оператора телефонии.

Для решения поставленной задачи были использованы язык программирования PHP, система управления базами данных MySQL.

Сокращения и определения

MySQL	Свободная реляционная система управления базами данных
PHP	Язык высокого уровня применяемый для разработки веб-приложений
HTTP	(англ. Hyper Text Transfer Protocol) – протокол передачи гипертекста
HTML	(англ. Hyper Text Markup Language) – стандартный язык гипертекстовой разметки документов
СУБД	Система **у**правления **б**азами **д**анных
CDR	(англ. Call Data Record) – подробная запись о вызове

ВВЕДЕНИЕ

В данной бакалаврской работе была поставлена цель создать веб-приложение, используя систему управления базами данных MySQL и язык программирования PHP, которое будет служить пользовательским интерфейсом для внутренних потребностей компании.

Актуальность темы

С каждым годом происходит увеличение доли пользователей в сети интернет среди населения России. Подходящим для этого в интернете является сайт (интернет- ресурс с URL-адресом) – к нему можно получить доступ с любого устройства где установлен веб-браузер и обладающим доступом в интернет. Большинство сайтов представляют собой веб-приложения, реализующие некий полезный функционал. С учетом увеличения интернет пользователей, появляется возможность удаленно выполнять большое количество операций. Можно отметить основные преимущества веб-приложения, которое способствует выполнению задач удаленно:

1) геонезависимость (возможность совершать действия с любой точки земли);

2) доступность (для любых устройств с установленным веб-браузером);
3) частичная или полная автоматизация процесса деятельности.
Представленные преимущества определили выбор разработки приложения с использованием веб-технологии.

Целью данной работы является разработка аналитического веб-приложения для обработки звонков. Требуется по спискам вызовов определить максимальную загрузку канала (максимальное количество занятых параллельных линий), количество совершаемых вызовов на определенный номер, количество вызовов на каждый из номеров клиента за задаваемый промежуток времени.

Для достижения указанной цели необходимо решить следующие задачи:
-проанализировать существующие веб-приложения;
-сформулировать функциональные требования;
-проанализировать существующие методы и технологии для создания веб-приложения;
-спроектировать веб-приложение в соответствии с требованиями;

-разработать веб-приложение;
-протестировать весь реализованный функционал.

Работа состоит из введения, трех разделов, заключения и библиографии.
Первый раздел описывает предметную область, в рамках которой ведется данная работа. Во втором разделе описывается проектирование и реализация веб-приложения для обработки звонков. В третьем разделе приведена информация о результатах тестирования разработанного веб-приложения. В заключении суммируются основные результаты работы.

Глава 1. Анализ существующих приложений и формулирование функциональных требований.

1.1. Обзор существующих приложений.

Веб-приложения – это вспомогательные программные средства, предназначенные для автоматизированного выполнения каких-либо действий на Веб серверах, например, как в нашем случае, для сбора статистики, поиска информации по звонкам и т.д.

Вся связь осуществляется между сервером и клиентом, т.е. данные расположены на серверах формируются и передаются по сети по запросу клиента. Главным достоинством такой работы веб-приложения является факт, при котором клиент никак не зависит от операционной системы, пользователя и веб-приложения, т.е. являются межплатформенными сервисами. Клиент имеет связь с сервером, используя любой доступный браузер, а за сервер отвечает веб-сервер, который принимает HTTP запросы от клиентов и передаёт их HTTP ответами, вместе с HTML страницей, файлами, изображениями и другими данными.

В задачи веб-приложения входит: получение запросов, проведение идентификации пользователя и проверку прав доступа, выполнение обработки данных, формирование веб страниц, отправка их обратно к пользователю, защита всей информации, а также обслуживание запросов других типов.

Чтобы понять, как работает Веб-сервер, следует иметь представление о принципе передаче информации в сети. В основе лежат правила, называемые протоколами. Рассмотрим один из протоколов прикладного уровня HTTP, который указывается в начале URL адреса (http://). Само понятие «Веб-сервер» может иметь отношения, как и к самому серверу, так и к программному обеспечению. Если рассматривать сам сервер (оборудование) — это компьютер, на котором размещены ресурсы сайта (HTML документы, CSS стили, JavaScript файлы и другое) и который передает их на устройство конечного пользователя (веб-браузер). Обычно сервер подключен к сети Интернет и может быть доступен через доменное имя, например mozilla.org, google.ru, yandex.ru и другие.

Для опубликации веб-сайта необходим, либо статический, либо динамический веб-сервер.

Статический веб-сервер, включает в себя компьютер с HTTP-сервером (ПО), т.е. сервер передает находящиеся на нем файлы клиенту в исходном виде.

Динамический веб-сервер состоит из статического веб-сервера, дополнительного ПО, сервера приложений и базы данных. Его называют “динамическим”, т.к. сервер, принимая запрос, обрабатывает его и отправляет результат клиенту по HTTP. К примеру, для получения итоговой страницы, отображаемой в браузере, сервер приложений может заполнить HTML шаблон данными из базы данных. Такой сайт как Википедия состоит из тысяч веб-страниц, но он не является реальными HTML документами (лишь несколько HTML шаблонов и гигантские базы данных). Эта структура упрощает и ускоряет сопровождение веб-приложений и доставку контента.

Существует большое количество веб-приложений для обработки звонков. Рассмотрим несколько из них более подробно:

1. «Asternic Call Centers Stats». Панель Call-центра Asternic Call Centers Stats работает в реальном времени и позволяет операторам и администратору очереди, видеть самую

свежую информацию, связанную с очередью. Панель Call-цента является модулем модуль отчетов AsternicCC, который, в свою очередь, дает возможность построения отчетов по работе самого call-центра:

- входящие звонки;
- принятые/непринятые звонки по очередям и операторам;
- распределение звонков между операторами;
- монитор отображения текущих разговоров в call-центр;
- отчет очереди входящих звонков и агентов;
- уровень обслуживания входящих звонков;
- нагрузка на очереди;
- причина разъединения;
- выбранный временной диапазон (час/день/месяц);
- время ожидания/ продолжительность разговора очереди;
- рейтинг покинувших очередь.

Задачей данного модуля отчетов является не только выбор звонков за определенный период, а группировка,

постройка графиков, а так же мониторинг состояния очередей, текущей занятости операторов и входящих звонков. Все экраны работают в режиме реального времени. Это дает возможность выявить, кто из агентов отвечает на звонок и, при желании, проследить уровень обслуживания клиента конкретным агентом в режиме «прослушки» или подключить функцию «суфлер» и помочь провести консультацию клиента. Аналитико-статистические отчеты по Asternic Call Centers Stats позволяют осуществлять мониторинг работы, как небольших call-офисов, так и крупных call-центров за короткий промежуток времени. Углубленный анализ происходящих процессов, статистических выкладок позволяет управлять процессами по улучшению качества работы в самом начале возникновения проблемных моментов, не дожидаясь, когда они вырастут и успеют снизить прибыль. Главное это то, что теперь, Вы можете получать необходимую информацию по работе Ваших сотрудников в любом месте и в любое время простым вводом адреса в своем браузере.

2.«VoIPTime». VoIPTime выпускает на рынок два – три обновления в год (новые версии) и время от времени

создает и улучшает весь свой функционал для call-центров. К примеру, в сентябре 2014 года в VoIPTime Contact Center внедрена система планирования рабочего времени Work Force Management, которая упрощает работу супервизоров и помогает избежать появления ошибок в планировании рабочих процессов. Чтобы находиться в авангарде индустрии IP-телефонии, VoIPTime непрерывно посещают отраслевые выставки, конференции и другие события. Их цель — быть всегда впереди, в курсе всех событий, чтобы предоставлять клиентам наиболее актуальные решения для IP-телефонии и контакт-центров.

«VoIPTime» предлагает решения для разных отрасле бизнеса:

Аутсорсеры:

- Обработка всех каналов связи: звонки, письма, факсы, чаты, sms-информирование;
- Получение полной истории обращений клиента;
- Выбор режима обзвона для кампании;
- Охват всех рынков и сферы.

Банки:

- Интеграция колл-центра с банковской системой;

- Максимальная загрузка операторов;
- Особый функционал для работы с должниками и поручителями — Collection;
- Получение полной статистики по телемаркетингам.

Интернет – магазины:

- Маршрутизация входящих звонков;
- Отправление SMS покупателям с акциями;
- Автоматический обзвон с IVR-информированием;
- Каналы связи в 1: email, web-chat, звонки.

CPA сети:

- Predictive — режим обзвона для исходящих кампаний;
- Перезванивание всегда вовремя с CallBack;
- Двусторонняя API-интеграция с CRM;
- Планирование рабочие графики операторов в системе.

3. «Infinity». Крупная компания «ИнтелТелеком», занимающаяся разработкой вопросов по решению задач по автоматизации и обработке звонков разработала прогруппу Infinity. Программой Infinity пользуются не только масштабные call-центры с сотней менеджеров в штате,

которые обрабатывают несколько миллионов звонков в месяц, но и некрупные компании из нескольких операторов. Infinity включает в себя современные функциональные возможности, которые нужны для организации call-центра под ключ. «Infinity» позволяет быстро и качественно обрабатывать звонок каждого клиента, обратившегося в данную компанию.

ПО, которым пользуются call-центры определяют номера абонента и выводят на экран сотрудника карточку клиента, в которой находятся детальные сведения о звонящем. Находящаяся информация в карточке сортируется из разных корпоративных баз компании.

Во время разговора с контрагентом оператор дополняет информацию по работе с клиентом, внося изменения в имеющуюся информацию. И имеет возможность самостоятельно настроить карточку клиента через графический редактор.

1.2. Требования к веб-приложению.

Важную роль играет то, насколько разрабатываемое приложение соответствует, заложенным на стадии

проектирования системы, требованиям. Этот факт и определяет качество приложения. Требования к веб-приложениям можно разделить на два типа: функциональные и нефункциональные.

Функциональными называются такие требования, которые определяют функциональность системы, которая строится разработчиками, чтобы пользователь имел возможность выполнить ту или иную задачу в рамках своих процессов.

Веб-приложение должно иметь следующие функции:

1) периодически (раз в час) автоматически обновлять данные, приходящие на сервер;

2) иметь возможность вручную собрать и отправить данные о звонках и иной информации клиенту в виде диаграмм, таблиц и т.п.;

3) возможность внесения изменений в хранящуюся информацию;

4) предоставлять доступ к серверу и имеющейся на нем информации;

5) предоставление возможности поиска определенной информации в базе данных;

6) возможность сортировки данных, полученных на сервере по различным параметрам;

7) при открытии приложения проверять насколько актуально последнее обновление, и если оно устарело более чем на одну итерацию расписания (более чем на час), то обновить текущий статус вручную, не дожидаясь следующего срабатывания расписания;

8) обеспечить долговременное хранение информации, с автоматическим удалением каждые 6 месяцев и возможностью удаления вручную;

9) дает возможность обработки информации и вывода ее в виде сводных таблицах;

10) поддержка русского и английского языков.

Нефункциональными требования называются такие требования, которые определяются описанием характеристик приложения, являющимися важными для пользователя.

Рассмотрим, какие характеристики задают нефункциональные требования к системе:

• НАДЕЖНОСТЬ.

Надежность приложения можно определить условиями функционирования приложения (параметры сервера, максимальное количество пользователей приложения) и допустимыми показателями качества работы системы в этих условиях (время обработки запроса пользователя к системе, количество отказов системы). Т.о., надежным приложением можно назвать такое приложение, которое может обеспечить доступ ко всем функциям для пользователя при любых условиях (т.е. все возможные условия для данного приложения должны быть рассмотрены и учтены при проектировании системы);

• БЫСТРОДЕЙСТВИЕ ПРИЛОЖЕНИЯ.

Быстродействие приложения можно определить как среднее время обработки запроса пользователя к системе. Максимальным оптимальным временем отклика для веб-приложений считается 5 секунд;

• БЕЗОПАСНОСТЬ.

Требование безопасности веб-приложения включает в себя:

1) дифференциация прав доступа к функциям и данным каждого компонента веб-приложения;

2) управление уровнем доступа компонентов и/или пользователей;

3) авторизация и верификация пользователей.

• МАСШТАБИРУЕМОСТЬ.

Масштабируемость определяется возможностью увеличения системой своей производительности при повышенной нагрузке и росте ресурсов. Для пользователя масштабируемого веб-приложения должен оставаться незаметным момент (т.е. время отклика системы на запросы пользователя не должно заметно изменяться), когда возрастет нагрузка (например, к приложению получают доступ одновременно еще несколько пользователей), и при изменении конфигурации приложения (например, если на уровень бизнес-логики будет добавлен дополнительный компонент обработки данных).

Глава 2. Анализ существующих методов и технологий для создания веб-приложения и реализации функциональных требований.

Рассмотрим основные этапы создания веб-приложения. Процесс разработки веб-приложения, как и любой информационной системы, определяется понятием «жизненный цикл». Моделью жизненного цикла называют структуру, которая состоит из процессов, работ и задач, включающих в себя разработку, эксплуатацию и сопровождение программного продукта, охватывающая жизнь системы от установления требований к ней до прекращения ее использования. Основным нормативным документом, регламентирующим жизненны цикл программного обеспечения, является международный стандарт ISO/IEC 12207. Этот стандарт определяет процессы, работы и задачи, которые используются:

- при приобретении системы, содержащей программные средства, или отдельно поставляемого программного продукта;
- при оказании программной услуги;

- при поставке, разработке, эксплуатации и сопровождении программных продуктов. Перейдём к технологии создания данного веб-приложения.

2.1 Выбор технологий для создания веб-приложения.

2.1.1 Используемые языки программирования.

Язык программирования — формальный язык, предназначенный для записи компьютерных программ. Язык программирования включает в себя правила (лексические, синтаксические, семантические), которые оапеделяют внешний вид программы и действия, выполняющие исполнительем под ее управлением. Рассмотрим несколько языков программирования, которыми мы планируем пользоваться для разработки данного веб-приложения:

1.PHP (Personal Home Page) — сценарный язык и программное средство для создания веб-приложений. На данный момент Personal Home Page является лидером среди языков, которые применяются для разработки сайтов, а так же он поддерживается большенством

хостинг-провайдерами.

Personal Home Page включает в себя:

CGI интерфейс, интерпретатор языка и набор функций для доступа к базам данных и различным объектам WWW. Personal Home Page дает возможность создавать страницы в интерактивном режиме взаимодействия «клиент-сервер».

Так же Personal Home Page является одним из самых популярных языков в области веб-программирования, в часности серверной части. Большой спрос этого языка обьясняется тем, что в него включены множества встроенных средств для создания веб-приложения. Перечислим главные из них:

- автоматическое извлечение POST и GET-параметров, а также переменных окружения веб-сервера в предопределённые массивы;
- взаимодействие с множеством разных систем управления базами данных (MySQL, MySQLi, SQLite, PostgreSQL, Oracle (OCI8), Oracle, Microsoft SQL Server и др.);
- автоматизированная отправка HTTP заголовков;
- работа с HTTP-авторизацией;
- работа с cookies и сессиями;

- работа с локальными и удалёнными файлами, сокетами;
- обработка файлов, загружаемых на сервер;
- работа с XForms.

2.JavaScript. Чаще всего JavaScript используют как встраиваемый язык для программного доступа к объектам приложений. Наиболее широкое применение находит в браузерах как язык сценариев для придания интерактивности веб-страницам. Этот язык дает возможность создавать приложения, включающиеся в Html-код.

Использование JavaScript дает следующие возможности:

- внесение изменений в страницу;
- изменение стилей элементов;
- добавление или удаление тегов;
- узнать о действиях пользователя на странице (уменьшение или увеличение рабочей области экрана, клики с помощью мышки, нажатия любых клавиш, прокрутка страницы);
- получение доступа к элементам Html-кода и манипулирование с этими элементами;

- загрузка данных без перезагрузки страницы;
- ввод сообщений;
- установка или считывание cookie, а так же выполнение множества других действий.

Помимо вышеуказанных возможностей, существуют и ограничения использования сценариев:

- недоступны файлы, находящиеся на пользовательском компьютере;
- отсутствует доступ за пределами данной веб-страницы;
- нет возможности выполнения кроссдоменных запросов, то есть получение доступа к веб-страницам, которые находятся на другом домене, даже если они открыты в соседних вкладках;
- открывающиеся с его помощью вкладки и окна не закрываются;
- нет защиты исходных данных, которые находятся на странице и, нет запрета на копирование информации со страницы.

Подобные ограничения в некотором смысле затрудняют выполнение вредоносного кода.

3.JSON. JSONом называют такой формат данных, нацеленность которого определяется на отправку данных между браузерами, веб-серверами или моб.приложениями. JSON является подмножеством JavaScript.

JSON так же имеет ряд недостатков, даже, не смотря на то, что он заявлен как относительно сжатый и гибкий формат данных, с которым легко работать на многих языках программирования. Рассмотрим ниже недостатки формата данных JSON:

- у JSON отсутствует структуры;

- имеется всего 1 тип чисел. Поддерживается формат с плавающей запятой и двойной точностью IEEE-754. Это довольно много, но нет возможности использовать то многообразие числовых типов, что есть в других языках;
- нет типа даты. Разработчики должны использовать строковые представления дат, что может вызвать несоответствие форматирования. Или же использовать в качестве даты количество

миллисекунд, прошедших с начала эпохи Unix (1 января 1970);

- нет комментариев — нет возможности делать аннотации для полей, которые требуют этого прямо в коде;
- подробность — хотя JSON менее подробный, чем XML, это не самый сжатый формат обмена данными.

JSON имеет и ряд преимуществ, в который входят: гибкость, простота дизайна, JSON легок в прочтении и понимании, он доступен на многих языках программирования. Отсутствие строгой схемы обеспечивает гибкость формата, но такая гибкость иногда затрудняет чтение и понимание данных. Несмотря на происхождение от JavaScript, формат принято считать независимым от языка и может использоваться практически с любым языком программирования. Для многих языков существует готовый код для создания и обработки данных в формате JSON. За счёт своей лаконичности формат JSON является подходящим для сериализации сложных структур. Если говорить о веб-приложениях, в таком ключе он уместен в задачах обмена

данными как между браузером и сервером, так и между самими серверами

2.1.2. Общая структура приложения.

Практически любое веб-приложение можно описать с помощью нескольких компонентов. Эти компоненты представлены на рис.1.

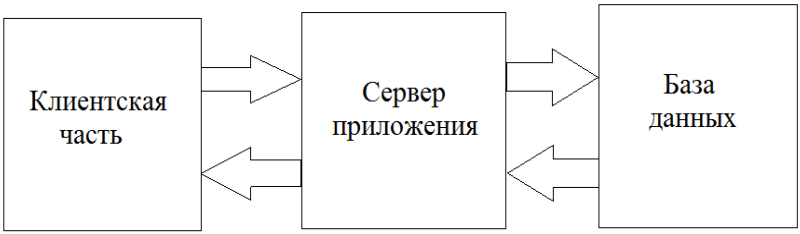

Рис.1. Общая структура БД

- Клиентская часть является обязательной для любого веб-сайта. Клиентская часть дает возможность клиенту взаимодействовать с

сервисом. Иногда ее может не быть, если сервис является очень узко-направленным приложением, но такое встречается крайне редко. Приведем элементарный пример: – база данных, веб-сайта у нее нет, однако она функционирует через предоставленный API.

- Сервер приложения – обязательная, минимальная часть веб-сервиса. Без сервера не сможет существовать клиентская часть.
- База данных явдяется обязательной частью веб-приложения. Но иногда, так же как и клиентская часть может отсутствовать. Чаще всего, для постоянной работы пользователей в своем сервере, который не пользуется хранилищем какого-либо другого сервиса, является необходимой частью.

2.2. Разработка структуры базы данных

Базой данных называют набор данных для информационных сетей и пользователей, хранящихся в специальной организованной форме. Тип хранилища

данных определяется указанной структурой (схемой) базы данных и ее правилами управления.

Классическая технология разработки реляционных баз данных связана с теорией нормализации, основанной на анализе функциональных зависимостей между атрибутами отношений. Процесс нормализации имеет своей целью устранение избыточности данных (любая информация, которая находится в базе данных может храниться только в единственном экземпляре, т.к. повторение данных может привести к тому, что между копиями одних и тех же данных может возникнуть несогласованность. Нормализация дает возможность кардинально уменьшить объем хранящейся в базе данных информации и удалить аномалии в организации хранения данных. Степень нормализации данных может быть различной. Приведение модели к требуемому уровню нормальной формы является основой построения реляционной базы данных.

Выделяются шесть нормальных форм, три из которых являются основными: 1,2,3,4,5 нормальная форма, а так же Бойса-Кодда, которая находится по середине, между 3 и 4. Базу данных (БД) принято считать нормализованной при условии, что ее таблицы представлены минимум в 3

нормальной форме. Не редко большое количество таблиц нормализуют до 4 нормальной формы, реже, наоборот, производится денормализация. Использования таблиц в 5 нормальной форме в реальных базах данных встречается редко.

Рассмотрим более подробно эти формы нормализации:

1 нормальная форма определяется неделимостью каждого поля таблицы базы данных и отсутствием повторяющихся одинаковых полей. Неделимость поля означает, что содержащиеся в нем значения не должны быть разделены на более мелкие. Повторяющимися являются поля, содержащие одинаковые по смыслу значения. Требует определить основной ключ для таблицы, то есть тот столбец или комбинацию столбцов, которые однозначно определяют каждую строку.

Под второй нормальной формой подразумевается, что все поля таблицы БД зависят от основного ключа, если в одной из таблиц зависимость полей идет не от первичного ключа, то необходимо объединить их в отдельную новую таблицу. При таком раскладе необходимо определить

первичным ключом новой таблицы ту часть основного ключа, от которой зависят данные поля, и установить связь "один ко многим" от новой таблицы к старой.

Требования третьей нормальной формы состоят в следующем: таблицы не имеют транзитивные зависимости между не ключевыми полями, т.е., значение любого поля, не входящего в первичный ключ, не зависит от значения другого поля, также не входящего в первичный ключ.

Легко поддерживающаяся, не содержащая в себе неопределенных и повторяющихся данных модель данных и является результатом нормализации.

Поддерживать работу без помощи БД с включающим в себя несколько HTML страниц статическим проектом достаточно просто. Однако, стоит отметить, что у сайтов есть стремление разрастаться. С динамичным проектом такой метод работы уже не актуален. Хранить массивы различной информации в сотнях файлов, а затем требовать от них определенные строки при работе вэб-сервера – дело хлопотное и медленное. Базы данных дают возможность структурировать и систематизировать данные. Базы данных имеют простой код использования, а так же

преимущество в виде того, что времени на запрос уходит на много не много.

Отметим, что удобнее всего хранить свои данные в виде списков, таблиц и т.д., но база данных не всегда является статичным образованием, даже, чаще всего, она регулярно дополняется, развивается, корректируется и растет, с ней трудно справиться. Для облегчения процесса работы с БД была разработана Система управления базами данных (СУБД)

СУБД (система управления базами данных) представляет собой набор программных продуктов, которые используются для составления баз данных различными пользователями. Посредником между пользователем и БД является СУБД. Рассмотрим основные функции СУБД:

- поиск нужных данных;
- физическое размещение данных и их описаний;
- обновление и пополнение баз данных в соответствии с изменениями в реальном мире (поддержка актуального состояния);
- защита данных от взлома, некорректных изменений и запрещенного доступа;

- регулирование и направление одновременных запросов к базе от нескольких пользователей (такая функция выполняется с помощью специальных прикладных программ).

SQL – это структурированный язык запросов, который был разработан с целью управления реляционными базами данных.

Рассмотрим перечень возможностей SQL:

- создание новой таблицы в базе данных;
- редактирование и полное удаление записей;
- производить запросы из таблиц;
- выбор записи из разных таблиц, в соответствии с заданными условиями;
- изменение вида и структур одной или нескольких таблиц.

SQL обладает несколькими видами запросов, подразумевающими под собой запрос данных из нужной базы или обращение к БД с обязательным изменением в ней информации.

В связи с этим принято выделять следующие виды запросов:

- создание или изменение в базе данных новых или уже существующих в ней объектов;
- получение данных;
- добавление новых данных в таблицу;
- удаление данных;
- обращение к системе управления базами данных (СУБД).

Стоит отметить преимущества и недостатки данной системы работы:

Преимуществами SQL можно назвать следующие факты:

- независимость от существующей в данной системе СУБД;
- тексты SQL являются универсальными для многих СУБД (только в простых задачах, связанных с обработкой данных в таблицах)
- наличие стандартов SQL способствует "стабилизации" языка;
- декларативность. Декларативность объясняется тем, что программист способен выбрать исключительно

те данные, которые должны быть изменены или модифицированы. Способ решения данной задачи выбирается автоматически в самом программном уровне СУБД.

Так же необходимо отметить и недостатки SQL:

- SQL не соответствует реляционной модели построения данных, т.к. SQL заменяет язык Tutorial D, являющийся по настоящему реляционным;
- Язык SQL имеет очень большую сложность. Пользоваться таким языком способен только программист, не смотря на то, что и первоначально он создавался как средство управления, с которым будет работать обычный пользователь;
- некоторое несоответствие стандартов. Многие компании, разрабатывающие СУБД, добавляют свои особенности в диалект языка SQL, что существенно влияет на универсальность языка.

Существует множество СУБД поддерживающих SQL язык запросов: *MySQL, mSQL, PostgreSQL, MSSQL* и многие другие. Каждая из них имеет преимущества в

определенной сфере, но не именно MySQL завоевала широкое признание и популярность в Интернете благодаря своей гибкости и универсальности.

Основными преимуществами в работе MySQL можно назвать высокую скорость работы, быстроту обработки данных и оптимальную надежность. Имеет большое значение то бесплатное распространение данной СУБД и ее ПО с открытым кодом. За счет этого можно вносить свои изменения и модифицировать код, что весьма полезно для веб-мастеров. Стоит отметить, что работа с Mysql может производиться не только в текстовом режиме, но и в графическом. Разработан очень популярный визуальный интерфейс для работы с этой СУБД, а именно PhpMyAdmin. Этот интерфейс дает возможность значительного упрощения работы с БД в Mysql. PhpMyAdmin помогает в полной мере пользоваться браузером, включая прокрутку изображения, если оно не умещается на экран. Многие из базовых SQLфункций работы с данными в PhpMyAdmin сведены к интуитивно понятным интерфейсам и действиям, напоминающим переход по ссылкам в Internet.

PhpMyAdmin. Это приложение системы управления базами данных с открытым кодом, которое написано на языке PHP и представляет собой веб-интерфейс для администрирования СУБД MySQL. PHPMyAdmin предоставляет возможность выполнять администрирование сервера MySQL, запуска команды SQL и просмотр содержания таблиц и БД с помощью браузера и не только. PhpMyAdmin является очень актуальным и популярным приложением среди веб-разработчиков, потому что оно дает возможность управлять СУБД MySQL без участия SQL и ввода команд, предоставляя дружественный интерфейс. На сегодняшний день PHPMyAdmin широко применяется на практике. Это происходит по причине того, что веб-разработчики актично развивают продукт, учитывая все нововведения СУБД MySQL. Многочисленные российские провайдеры применяют приложение PhpMyAdmin, как панель управления для передачи своим заказчикам возможность администрирования выделенных им баз данных.

В ходе выполнения работы была разработана следующая структура базы данных:

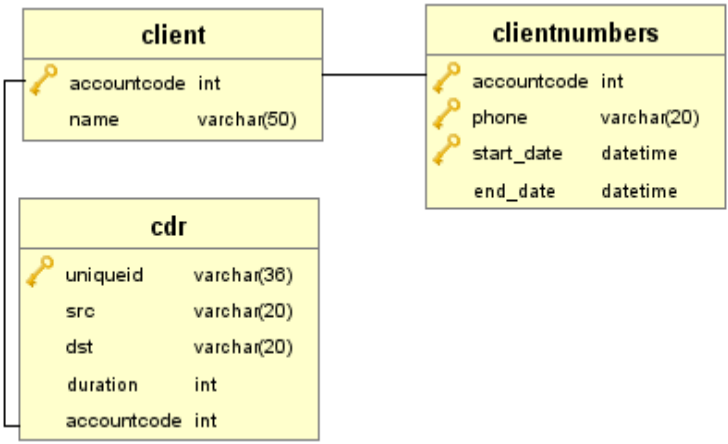

Рис.2. Структура БД

Client – таблица со списком клиентов, содержащая номер клиента (accountcde), получаемый от телефонной станции и наименование клиента. Наименование клиента уникально. ClientNumbers – таблица, содержащая соответствие номеров телефонов и клиентов с учетом времени, когда конкретный номер был выдан определенному клиенту, т.е. один и тот же номер сначала мог быть выдан одном клиенту, а затем другому. При этом создана связь таблицы clientNumbers с Client, при которой в случае наличия записей в clientNumbers при попытке

удаления клиента из Client будет выдана ошибка. Cdr (Call Data Record) – таблица содержит звонки клиентов. При этом в таблице могут быть переадресованные звонки (т.е. те, у которых ни src ни dst не соответствуют полю phone в таблице clientNumbers). Создана связь таблицы cdr с Client, при которой в случае наличия записей в cdr при попытке удаления клиента из Client будет выдана ошибка. Т.е. если необходимо удалить клиента, необходимо сначала удалить все его записи в cdr, и все его записи из clientNumbers.

```
CREATE DATABASE pavel;
USE pavel;

CREATE TABLE client (
    accountcode INT NOT NULL PRIMARY KEY,
    name VARCHAR(50) NOT NULL UNIQUE
);

CREATE TABLE clientNumbers (
    accountcode INT NOT NULL,
    phone VARCHAR(20) NOT NULL,
    start_date DATETIME NOT NULL,
    end_date DATETIME,
    PRIMARY KEY(accountcode,phone,start_date),
    FOREIGN KEY (accountcode) REFERENCES client(accountcode) ON DELETE RESTRICT
);

CREATE TABLE cdr (
    uniqueid varchar(36) NOT NULL PRIMARY KEY,
    src VARCHAR(20) NOT NULL,
    dst VARCHAR(20) NOT NULL,
    calldate DATETIME NOT NULL,
    duration INTEGER,
    accountcode INT  NOT NULL,
    FOREIGN KEY (accountcode) REFERENCES client(accountcode) ON DELETE RESTRICT
);
```

Далее рассмотрим основные запросы:

1. Выборка количества звонков, сгруппированных по диапазонам в 5 минут:

```
SELECT concat(date(calldate), ' ',
lpad(hour(calldate),2,'0'), ':',
lpad(floor(minute(calldate)/5)*5,2,'0')) as
calldate5min, count(uniqueid)
FROM cdr
WHERE calldate BETWEEN $start_date AND $end_date
GROUP BY date(calldate), lpad(hour(calldate),2,'0'),
lpad(floor(minute(calldate)/5)*5,2,'0')
ORDER BY calldate5min;
```

В запросе с использованием функции minute выбирается минута из каждой даты, затем делится на 5 и усекается функцией floor(т.е. отбрасывается дробная часть), затем умножается на 5 и функцией lpad дополняется в начале нулями. Таким образом для каждого звонка высчитывается в какую пятиминутку он был совершен. Затем собирается дата каждой пятиминутки, используя функцию concat.

Оператором GROUP BY группируются все звонки, совершенные в одну пятиминутку и для каждой из пятиминуток высчитывается количество данных звонков с использованием функции count.

Рис. 3 Количество вызов в пятиминутные промежутки

На графике видно, что происходит большое количество автоматически совершаемых вызовов. Это связано с использованием системы контроля работоспособности телефонии в организации, которые раз в 15 минут производят автоматический «прозвон» номеров.

2. Выборка максимального количества параллельных вызовов за определенный промежуток времени, сгруппированный по диапазону в 5 минут:

```
DELIMITER $$
CREATE PROCEDURE filldates(dateStart DATETIME, dateEnd
DATETIME)
BEGIN
      WHILE dateStart <= dateEnd DO
        INSERT INTO MyDates (mydate) VALUES (dateStart);
        SET dateStart = date_add(dateStart, INTERVAL 5
MINUTE);
      END WHILE;
    END $$
DELIMITER ;
```

Была создана процедура Filldates ,которая заполняет таблицу Mydates, значением времени по каждому пятиминутному отрезку.

```
use pavel;
Set @start_date='2018-05-27';
Set @end_date='2018-05-28';
drop table if exists MyDates;
CREATE TEMPORARY TABLE MyDates
(
  mydate datetime unique
);

CALL filldates(@start_date,@end_date);

insert ignore into MyDates SELECT calldate from cdr
where calldate between @start_date and @end_date;

select
       concat(date(mydate), ' ',
lpad(hour(mydate),2,'0'), ':',
lpad(floor(minute(mydate)/5)*5,2,'0')) as mydate5min,
       max(call_count) as concurent
from (
       select
              mydate,
```

```
            count(cdr.uniqueid) as call_count
       from MyDates
        Left join cdr on (MyDates.mydate between
cdr.calldate and date_add(cdr.calldate, interval
cdr.duration second))
        group by mydate
) as call_count
GROUP BY date(mydate), lpad(hour(mydate),2,'0'),
lpad(floor(minute(mydate)/5)*5,2,'0')
```

Создается таблица со временем, которая заполняется отрезками по 5минут, в выбранном диапазоне. Затем в эту таблицу добавляются времена всех звонков, попавших в этот диапазон. При этом игнорируется дубликаты по времени, за счет использования индекса типа Unique. После этого высчитывается количество параллельных вызовов для каждого из значений времени (таблица call_count). Далее происходит группировка этой таблицы с выбором максимального значения параллельных вызовов, в каждой из пятиминутных интервалов. В итоге получается таблица, где отображается максимальное количество параллельных звонков по пятиминутным диапазонам.

Рис. 4. Количество параллельных вызовов

При сравнение первого и второго графика видно, что кол-во звонков в 5 раз превышает максимальное кол-во параллельных линий, что свидетельствует о том, что большинство вызовов нулевой длительности.

3. Выбор количества звонков на номера задаваемого клиента и задаваемый промежуток времени.

```
select phone, count(cdr.uniqueid) as count from client
left join clientnumbers on
client.accountcode=clientnumbers.accountcode
left join cdr on (cdr.accountcode=client.accountcode and
                (cdr.dst like clientnumbers.phone or
cdr.src like clientnumbers.phone) and
                cdr.calldate>clientnumbers.start_date
and
                (cdr.calldate<clientnumbers.end_date or
clientnumbers.end_date is null))
where name like 'возовоз'
      and cdr.calldate between $start_date AND $end_date

group by phone
```

Сначала по заданному имени клиента выбираются его номера. Затем в его звонках находятся вызовы на его номера или с его номеров, причем проверяется, что во время совершения вызова, данный номер принадлежал клиенту. После этого подсчитывается количество вызовов для каждого из номеров.

2.3. Разработка серверного приложения на PHP

Приложение на PHP выполняет следующие функции:

1. Проверка и анализ входных параметров HTTP запросов клиента.

2. Формирование SQL запроса к базе данных.
3. Обработка ответа от базы данных и отправка данных в формате JSON клиенту.

Проверка и анализ входных параметров HTTP запросов клиента – осуществляет проверку формата и полноты передаваемых данных. В случае обнаружения несоответствия выдается ошибка, которая обрабатывается и отображается на стороне клиента, с использованием JavaScript.

Формирование SQL запроса к базе данных – в подготовленный шаблон SQL запроса вставляются полученные данные из HTTP запроса. И сформированный запрос отправляется в СУБД, используя библиотеку php-mysql.

Обработка ответа от базы данных и отправка данных в формате JSON клиенту – Полученные данные от СУБД преобразуются в необходимый формат массива и отправляется в виде JSON используя соответствующую библиотеку.

2.4. Разработка клиентского приложения HTML и JavaScript.

1. HTML, представляет из себя страницу содержащую форму для заполнения пользователем, данные из которых отправляются на сервер, в виде HTML запроса.
2. Полученный ответ обрабатывается в canvasjs javascript библиотеке, которая отображает полученные данные в виде графиков.
3. При получении ответа содержащего описание возникшей ошибки на стороне сервера – она выводится на экран пользователя.

Глава 3. Тестирование и проверка соответствия разработанного портала требования.

Во время тестирования всех запросов рассмотрены следующие ситуации:

1. Звонки, которые начинались до начала выбранного нами интервала (пятиминутки), могли закончится (вешалась трубка) внутри интервала.

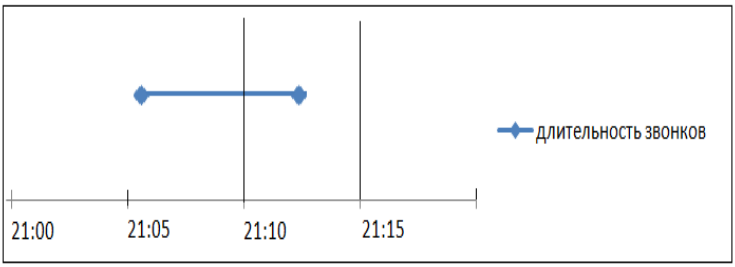

Рис.5.

2. Звонки, которые начинались до начала выбранного нами интервала (пятиминутки), закончились (вешалась трубка) после интервала.

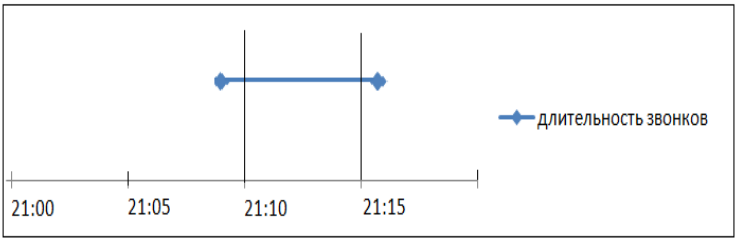

Рис. 6.

3. Звонки, совершенные “внутри” пятиминутки, длятся, не выходя за рамки выбранного интервала. При этом они могут пересекаться внутри интервала, а могут идти последовательно.

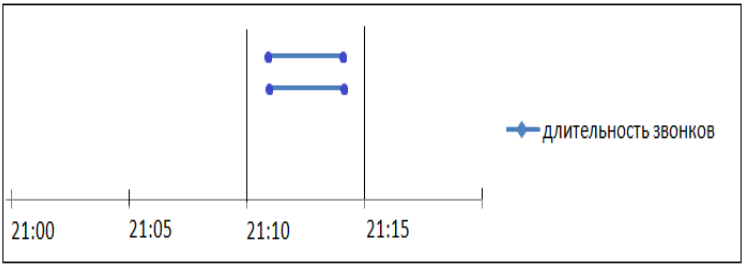

Рис. 7

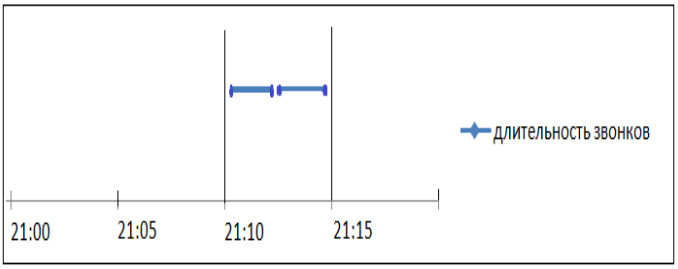

Рис. 8.

4. Звонки совершенные “внутри” пятиминутки заканчиваются вне выбранного интервала.

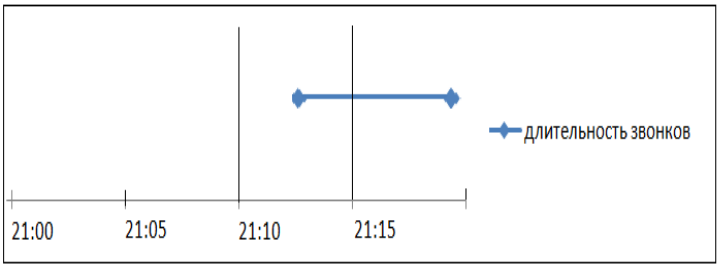

Рис. 9.

Все звонки в первом запросе, начало которых попадает в пятиминутный отрезок времени были учтены. Те звонки, начало которых выходят за рамки промежутка, в этом отрезке не учитывались.

Второй запрос, о максимальном количестве параллельных звонков за выбранный промежуток был разделен также на пятиминутные отрезки. Ни один звонок, за выбранный нами промежуток, не остался не зафиксированным и был учтен, с последующим отображением на графике.

Третий запрос, представляющий собой задачу о поисках менее используемых номеров определенно взятого клиента. Проверяется принадлежность номера клиенту (что он активен и не был отключен) в момент совершения вызова. Рассматриваются варианты, при которых номера в определенный промежуток времени принадлежали сначала одному клиенту, а затем другому.

Результаты соответствовали ожидаемым значениям количества совершенных вызовов.

Выводы

В ходе работы были изучены принципы работы СУБД, HTTP-сервера и языки PHP, Javascript.

Было разработано полноценное приложение, позволяющее в онлайн режиме анализировать текущую загрузку серверов телефонии.

Использование разработанного приложения позволяет формировать рекомендации клиентам заказчика по оптимизации выбора набора услуг (количество параллельных линий, количество используемых номеров). Помимо этого система позволяет обнаруживать наиболее загруженные часы, с наибольшим количеством поступающих звонков и количества линий.

СПИСОК ЛИТЕРАТУРЫ

1. Бенкена Е. PHP, MySQL, XML. Программирование для Интернета; БХВ-Петербург - М., 2017.
2. Гарнаев А. WEB-программирование на Java и JavaScript - БХВ-Петербург, 2013.
3. Веллинг, Л. Разработка Web-приложений с помощью PHP и MySQL / Веллинг, Л. Томсон. - М.: Вильямс, 2013.
4. Вигерс К.И. Разработка требований к программному обеспечению – М.: 2014
5. Колесниченко, Д.Н. PHP и MySQL. Разработка веб-приложений. Профессиональное программирование / Д.Н. Колисниченко. –СПб. : BHV, 2015
6. Никсон Н. Создаем динамические веб-сайты с помощью PHP, MySQL, JavaScript, CSS и HTML5 - СПб, 2016.
7. Посконин А. Web-приложения и данные: проблемы абстракции и масштабируемости / Труды Института системного программирования РАН. – 2012.

8. Пьюривал С. Основы разработки веб-приложений - СПб, 2014.

9. Хэррон Д. Nodje.js Разработка серверных веб-приложений на JavaScript / Хэррон Д. – М.: ДМК, 2014