**Министерство науки и образования Республики**

**Молдова**

**Технический университет Молдовы**

**Курсовая Работа  
  
По дисциплине «Базы Данных»**

Выполнил: Фадин Александр MI-192

Приняла: A. Chertari

Дата сдачи: 02.12.2020

**Кишинев 2020**

Курсовая Работа

Глава 1: Введение в БД

База данных - совокупность самостоятельных материалов, систематизированных таким образом, чтобы эти материалы могли быть найдены и обработаны с помощью электронной вычислительной машины.  
Обычно говорится так, или же что База Данных – это своего рода ящик или же шкаф с ящиками для хранения документов и прочих бумаг. Думаю, у каждого дома есть такой, всем нам нужны такие контейнеры, так или иначе. В IT сфере они используются чуть ли ни повсеместно. Порой мы даже не замечаем этого, но взять к примеру коммерческие приложения, сайты, где необходимо использовать данные компании и их обработку в реальном времени. Или же всем известные социальные сети? А что насчет таких известных сервисов, как Amazon, EBay, IHerb и прочих используемых нами ежедневно? Мы сталкиваемся с потребностью этого аспекта разработки, даже если Вы учитель в школе и не имеете опыта работы с PC , ваша база данных – это ваш архив с данными учеников.

В моём же случае, Базы Данных используются в области менеджмента и предпринимательства, что в свою очередь, так - же может считаться коммерческим проектом. Однако я использую БД в данной работе для сравнительного анализа, что несёт в себе крайне высокую ценность в перспективе.

* 1. Концепция Базы Данных. Обзор.

В данной работе БД представляет собой набор объектов для демонстрации структуры управления товаром в компании или же в магазине (сети магазинов). Так как моей темой является «Производитель и Молочная продукция», среди объектов: *Категории* продуктов – *Виды* каждого из продуктов, *Заказчики* (потенциальные реализаторы) – *Заказы* (в которых содержится информация о:

* Количестве товара в партии
* Цене этого товара
* Наименовании товара
* Цена транспортировки и т.д.)

*Отчеты*, в которых будет содержаться информация о каждом из заказчиков согласно прошлым поставкам, а именно:

* Кол-во закупаемого товара
* Кол-во реализованного товара
* Времени реализации
* Коэффициент, благодаря которому и будет сравниваться перспектива работы с тем или иным заказчиком.
  1. Системы управления Базами Данных

Благодаря курсу по Базам Данных, я ознакомился с такой СУБД, как MariaBD, что является ответвлением от системы управления базами данных MySQL, разрабатываемое сообществом под лицензией GNU GPL. Разработку

и поддержку MariaDB осуществляет компания MariaDB Corporation Ab и фонд MariaDB Foundation.



MySQL в свою очередь, система управления базами данных. Разработку и поддержку MySQL осуществляет корпорация Oracle, получившая права на торговую марку вместе с поглощённой Sun Microsystems, которая ранее приобрела шведскую компанию MySQL AB.  
Что касается общего определения, MySQL это система управления реляционными базами данных с открытым исходным кодом (СУРБД) с моделью клиент-сервер. [СУРБД](https://ru.wikipedia.org/wiki/%D0%A0%D0%B5%D0%BB%D1%8F%D1%86%D0%B8%D0%BE%D0%BD%D0%BD%D0%B0%D1%8F_%D0%A1%D0%A3%D0%91%D0%94) — это программное обеспечение или служба, используемая для создания и управления базами данных на основе реляционной модели.

Глава II BD Дизайн для данного проекта.

2.1 Презентация и краткое описание DS аудита проекта.

Для реализации проекта предусмотрен план его разработки. Он разделен на 5 главных частей:

* 1) Теоретическая часть для ознакомления с основами.
* 2) Этап подготовки. Моделирование БД, определение задач и планирование дальнейшей работы, построенное схематически.
* 3) Веб – программирование, концепции, обзор языков и архитектуры построения связей с сервером.
* 4) Структура БД, проектирование и практическая реализация всех технологий.
* 5)Презентация готового проекта с демонстрацией функционала.

На данный момент моей задачей является построение Веб- или Desktop – приложения, которое бы было связано с сервером и имело возможность к взаимодействию с базой данных, посредством:

* Добавления
* Изменения
* Поиска
* Удаления

Проще говоря, у меня имеется вид конечного результата, но неизвестно, с чего начинать. На момент у меня нет уже сконструированной базы данных, веб- или desktop – приложения, но есть представление того, как оно будет реализовано: выше описанная БД, Веб-приложение для менеджмента.

2.2-4 Формулировка Задачи. Определение Целей.

Задачей проекта является создание эффективного Веб-приложения, которое позволит менеджерам и head кадрам управлять потоком заказов, клиентов и сравнивать перспективность и эффективность того или иного клиента. Для примера приведем производителя молочных продуктов, которому нужно обеспечить поставки товара заказчикам, которые реализуют эту продукцию. Таких заказчиков, а именно «магазинов» может быть множество. Как же справиться? Как обеспечить ответственный персонал комфортным инструментом для отслеживания и оценки всех заказов и реализационной эффективности каждого из заказчиков? На помощь приходит именно это приложение, которое не только экономит огромное количество времени, но и будет вычислять коэффициент эффективности исходя из отношения кол-ва товара в партии к кол-ву и скорости его реализации. Таким образом можно будет определить для компании самые эффективные варианты поставки. Ведь если потребитель недоволен просроченными продуктами, это бьет по репутации как магазина, так и производителя. Таким образом можно определить цели так: Исправная работа приложения и попытка внедрения технологии на местный рынок. Так как на данный момент, это невозможно, для проверки исправности и демонстрации будет использоваться искусственная симуляция поведения заказчика, определенная алгоритмом.

2.5 Описание информационного контента.

При входе в приложение производителя, ему будет предоставлена форма для заказа, где будет указываться кол-во и наименование заказываемых единиц, их стоимость, а так - же стоимость поставки, то есть расходы на топливо. При следующем заказе будет обязательно выведена форма для заполнения отчёта о реализованной продукции с прикрепленным файлом с копией оригинала. Таким образом, как я уже говорил, будет выведен коэффициент, согласно формуле (a+b)/c.

2.6 Презентация концептуально модели

Производитель Молочных Продуктов

В модели видно, как строится иерархия объектов в БД. В будущем, при конструировании, заполняться будут в порядке – *от меньшего к большему,* так как самые низкие объекты – независимы, в то время, как остальные зависимы или взаимозависимы.

Виды связей между объектами:

Как видно, в модели объекты связаны при помощи двух видов связей:  
Один к одному и Один ко многим.

Категории 🡪Виды ----------------------------------------------------------------------1х1

Заказчики🡪 Заказы----------------------------------------------------------------------1х1

Заказы🡪Товар, Кол-во, Цена товара, Цена доставки----------------------------1хN

Отчёты🡪Коэф, Кол-во Т, Кол-во Р, Время----------------------------------------1Хn

2.8 Ограничения

* Связи между объектами таблиц присутствуют только двух видов, исключая связь множества к одному, а так - же много ко многому.
* Невозможность, в данный момент, испытания на приложения на реальных людях в реальных условиях.
* Нет возможности изменить заказ в рамках ранее оформленного заказа. Появляется необходимость в новом заказе.
* При появлении нового товара или изменения цен, необходимо применять изменения в базе данных.
* Нет возможности связи непосредственно с заказчиком в случае экстренной ситуации.

2.7 Презентация Логической модели

|  |
| --- |
| Kinds |
| id\_kind |
| NameKind |

|  |
| --- |
| Сategories |
| id\_categories |
| NameCat |
| id\_kind |

|  |
| --- |
| Client |
| id\_client |
| NameClient |
| id\_order |

|  |
| --- |
| Orders |
| id\_order |
| NameOrd |
| id\_product |
| id\_count |
| id\_pricePr |
| id\_priceDel |

|  |
| --- |
| Reports |
| id­\_report |
| NameRep |
| id\_coeff |
| id\_countPr |
| id\_countRe |
| id\_time |

|  |
| --- |
| Coefficient |
| id\_coeff |
| NameCoeff |

|  |
| --- |
| Product |
| id\_product |
| NameProd |

|  |
| --- |
| CountProd |
| id\_countPr |
| NameCountPr |

|  |
| --- |
| Count |
| id\_count |
| NameCount |

|  |
| --- |
| PriceProduct |
| id\_pricePr |
| NamePricePr |

|  |
| --- |
| PriceDeliv |
| id\_priceDel |
| NamePriceDel |

|  |
| --- |
| CountReal |
| id\_countRe |
| NameCountRe |

|  |
| --- |
| Time |
| id\_time |
| NameTime |

2.9Таблица покрытия.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Атрибуты и информация | Вывод информации | | | | | | | | Запросы | | | | | | Сервисы | | | | | | | |
|  | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 1 | 2 | 3 | 4 | 5 |  | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | |
| id\_categories | ***+*** |  |  |  | ***+*** | ***+*** |  |  | ***+*** |  |  |  |  |  | | ***+*** |  |  |  |  |  |  | |
| NameCat | ***+*** |  |  |  |  |  | ***+*** | ***+*** |  |  |  |  |  |  | |  |  |  |  |  |  |  | |
| id\_kind | ***+*** |  |  |  |  |  |  |  | ***+*** |  |  |  |  |  | | ***+*** |  |  |  |  |  | ***+*** | |
| id\_client |  | ***+*** |  |  |  |  |  |  | ***+*** |  |  |  |  |  | |  | ***+*** |  |  |  |  | ***+*** | |
| NameClient |  | ***+*** |  |  |  | ***+*** |  |  |  |  | ***+*** |  |  |  | |  |  |  |  |  |  |  | |
| id\_order |  | ***+*** |  |  |  |  |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  | |
| id\_kind | ***+*** |  |  |  |  |  |  |  |  | ***+*** |  |  |  |  | | ***+*** | ***+*** |  | ***+*** |  |  |  | |
| NameKind | ***+*** |  |  |  | ***+*** |  |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  | |
| id\_order |  | ***+*** | ***+*** |  |  |  |  |  |  | ***+*** |  |  |  |  | | ***+*** |  | ***+*** |  |  |  |  | |
| NameOrd |  |  | ***+*** |  |  | ***+*** |  |  |  |  | ***+*** |  | ***+*** |  | |  |  |  |  |  |  |  | |
| id\_product |  |  | ***+*** |  |  |  |  |  | ***+*** |  |  |  |  |  | |  |  |  |  |  |  |  | |
| id\_count |  |  | ***+*** |  | ***+*** |  |  |  |  | ***+*** |  |  |  |  | |  |  |  |  |  |  |  | |
| id\_pricePr |  |  | ***+*** |  |  | ***+*** |  |  |  |  | ***+*** |  | ***+*** |  | |  |  |  |  |  |  |  | |
| id\_priceDel |  |  | *+* |  |  | *+* |  |  |  |  |  | *+* | *+* |  | |  |  |  |  |  |  |  | |
| id­\_report |  |  |  | *+* |  |  | *+* |  |  |  |  | *+* |  |  | |  |  |  | *+* |  |  |  | |
| NameRep |  |  |  | *+* |  |  |  | *+* |  |  |  | *+* |  |  | |  |  |  |  | *+* | *+* |  | |
| id\_coeff |  |  |  | *+* | *+* |  |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  | |
| id\_countPr |  |  |  | *+* |  | *+* |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  | |
| id\_countRe |  |  |  | *+* |  |  | *+* |  |  |  |  |  |  |  | |  |  |  |  |  |  |  | |
| id\_time |  |  |  | *+* |  |  |  | *+* |  |  |  |  |  |  | |  |  |  |  |  |  |  | |
| id\_coeff |  |  |  | *+* | *+* |  |  |  |  |  |  |  |  |  | |  |  |  | *+* | *+* |  |  | |
| NameCoeff |  |  |  | *+* | *+* |  |  |  |  |  |  |  |  |  | |  |  |  |  |  | *+* |  | |
| id\_countPr |  |  |  | *+* |  | *+* |  |  |  |  |  |  |  |  | |  |  |  | *+* |  |  | *+* | |
| NameCountPr |  |  |  | *+* |  | *+* |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  | |
| id\_countRe |  |  |  | *+* |  |  | *+* |  |  |  |  |  |  |  | |  |  |  | *+* |  |  | *+* | |
| NameCountRe |  |  |  | *+* |  |  | *+* |  |  |  |  |  |  |  | |  |  |  |  |  |  |  | |
| id\_time |  |  |  | *+* |  |  |  | *+* |  |  |  |  |  |  | |  |  |  | *+* |  |  | *+* | |
| NameTime |  |  |  | *+* |  |  |  | *+* |  |  |  |  |  |  | |  |  |  |  |  |  |  | |
| id\_product |  |  | *+* |  |  |  |  |  | *+* |  |  |  |  |  | |  |  | *+* |  |  |  | *+* | |
| NameProd |  |  |  |  |  |  |  |  | *+* |  |  |  |  |  | |  |  |  |  |  |  |  | |
| id\_count |  |  | *+* |  |  |  |  |  |  | *+* |  |  |  |  | |  |  | *+* |  |  |  | *+* | |
| NameCount |  |  |  |  |  |  |  |  |  | *+* |  |  |  |  | |  |  |  |  |  |  |  | |
| id\_pricePr |  |  | *+* |  |  |  |  |  |  |  | *+* |  |  |  | |  |  | *+* |  |  |  | *+* | |
| NamePricePr |  |  |  |  |  |  |  |  |  |  |  | *+* |  |  | |  |  |  |  |  |  |  | |
| id\_priceDel |  |  | *+* |  |  |  |  |  |  |  |  |  | *+* |  | |  |  | *+* |  |  |  | *+* | |
| NamePriceDel |  |  |  |  |  |  |  |  |  |  |  |  |  | *+* | |  |  |  |  |  |  |  | |

\*Таблица покрытия показывает эффективность базы данных и её функционал в рамках выполнения запросов, выведения информации, а так - же обработки на сервере.

2.10 Презентация Макета

|  |  |  |  |
| --- | --- | --- | --- |
| **CATEGORIES** | id\_categories | INT | PRIMARY KEY (PK) NOT NULL |
| NameCat | VARCHAR | INSERT |
| id\_kind | INT | INSERT |
| **CLIENT** | id\_client | INT | PRIMARY KEY NOT NULL |
| NameClient | VARCHAR | INSERT |
| id\_order | INT | INSERT |
| **KINDS** | id\_kind | INT | PRIMARY KEY NOT NULL |
| NameKind | VARCHAR | INSERT |
| **ORDERS** | id\_order | INT | PRIMARY KEY NOT NULL |
| NameOrd | VARCHAR | INSERT |
| id\_product | INT | FOREIGN KEY (FK) |
| id\_count | INT | FOREIGN KEY (FK) |
| id\_pricePr | INT | FOREIGN KEY (FK) |
| id\_priceDel | INT | FOREIGN KEY (FK) |
| **REPORT** | id­\_report | INT | PRIMARY KEY NOT NULL |
| NameRep | VARCHAR | INSERT |
| id\_coeff | INT | FOREIGN KEY (FK) |
| id\_countPr | INT | FOREIGN KEY (FK) |
| id\_countRe | INT | FOREIGN KEY (FK) |
| id\_time | INT | FOREIGN KEY (FK) |
| **Coeffecient** | id\_coeff | INT | PRIMARY KEY NOT NULL |
| NameCoeff | VARCHAR | INSERT |
| **CountProd** | id\_countPr | INT | PRIMARY KEY NOT NULL |
| NameCountPr | VARCHAR | INSERT |
| **CountReal** | id\_countRe | INT | PRIMARY KEY NOT NULL |
| NameCountRe | VARCHAR | INSERT |
| **Time** | id\_time | INT | PRIMARY KEY NOT NULL |
| NameTime | VARCHAR | INSERT |
| **Product** | id\_product | INT | PRIMARY KEY NOT NULL |
| NameProd | VARCHAR | INSERT |
| **Count** | id\_count | INT | PRIMARY KEY NOT NULL |
| NameCount | VARCHAR | INSERT |
| **PriceProduct** | id\_pricePr | INT | PRIMARY KEY NOT NULL |
| NamePricePr | VARCHAR | INSERT |
| **PriceDeliv** | id\_priceDel | INT | PRIMARY KEY NOT NULL |
| NamePriceDel | VARCHAR | INSERT |

\*Макет демонстрирует предварительную структуру Базы Данных. При начале работы над созданием БД и таблиц, будут использоваться именно эти типы данных и ключи, как первичные, так и вторичные.

Глава 3. Введение в Web-программирование.

3.1 HTML концепт.

HTML – это язык гипертекстовой разметки, который получил очень широкое распространение. Язык HTML определяет структуру страниц, которую мы видим в браузере. Каждый сайт использует для отображения информации язык HTML. Как и протокол HTTP, язык HTML был разработан в ЦЕРНе Тимом Бернерсом-Ли в 1991 году и изначально он использовался ученными для обмена научными документами. HTML четко определял структуру документа и позволял выделять некоторые особенности текста документа, благодаря этому и тому, что синтаксис языка HTML был прост, он получил огромное распространение не только в научной среде, но и ушел в массы. Для осуществления разметки, используются так называемые «Теги», которые обладают своими индивидуальными функциями и свойствами. При работе мне были необходимы теги HTML, чтобы внести базовую информацию на страницу, чтобы её можно было обработать и оформить приемлемым образом. Среди таких тегов: div, a, form, scr, li и т.д.   
Сайты на HTML состоят из отдельных статичных страниц, написанных на html-коде и хранящихся в готовом виде. Эти страницы (в отличии от CMS) существуют реально и хранятся на каком-либо сервере. В динамичных сайтах, использующих CMS системы, страницы генерируются “на лету”, т.е. при запросе посетителя и используют информацию, хранящуюся в базе данных. У каждого типа сайтов есть свои плюсы и минусы и целесообразность использования конкретного варианта зависит от каждого конкретного проекта. Значимость тегов на веб - странице кажется очевидной, ведь если без CSS контент будет выглядеть непривлекательно и беспорядочно, то без HTML контента не будет вовсе. Совершенно необходимый тег – контейнер <div> позволяет работать намного эффективнее и быстрее, чем традиционные таблицы. Создание таблиц с помощью div даёт неограниченные возможности манипуляции контента на странице, что мне и помогло составить привлекательный для user’а интерфейс.

Основное различие между текстовыми редакторами и обработчиками HTML заключается в том, что последний не заботится о том, как элементы документа будут отображаться на экране. За очень немногими исключениями, HTML не описывает представление документа в целом. Он предлагает (на данный момент) небольшую поддержку в установке размещения или способа, которым будут отображаться элементы документа. HTML-дизайнеры намеренно выбрали этот вариант по той причине, что возможности платформы, на которой будет отображаться документ (размер экрана, установленные шрифты и т. Д.), Неизвестны. Отделение структуры документа от способа его отображения дает большую свободу программе, которая понимает HTML и отображает документ. Он может принимать решения о форматировании документа в зависимости от возможностей соответствующей платформы. Это то, чем занимаются веб-браузеры, помимо функций связи и поиска документов в сети. Когда браузер загружает документ HTML, он читает документ в поисках тегов HTML, форматирует текст и изображение и отображает их на экране. Вот почему один и тот же HTML-документ выглядит несколько иначе при просмотре в разных браузерах. Хотя на этой стадии разработки предлагаемые возможности форматирования по-прежнему весьма ограничены, предлагая довольно ограниченный контроль над формой документа, преимущество в том, что документы можно передавать и просматривать в любом месте в сети, независимо от используемой платформы и навигационной программы, привело к его очень быстрому распространению.

Любая версия HTML включает такие элементы, как текст с выравниванием по центру или по правому краю, таблицы, математические формулы, выравнивание изображений и текста. Расширения, появившиеся в большом количестве в последнее время, представляют собой наборы HTML-тегов, введенные различными компаниями (обычно производителями браузеров), которые позволяют авторам HTML-документов избегать некоторых ограничений стандарта. Наиболее распространены расширения Netscape и Internet Explorer.

Но возникает вопрос: почему HTML был предпочтен для веб-публикаций, когда существует несколько технологий для создания электронных публикаций? Первая причина - простота. Во-вторых, он позволяет форматировать текст ASCII метками в формате ASCII. Результат - хорошее сжатие, поддержка гипертекстовых ссылок и простота написания навигационных программ для просмотра документов.

3.2 Языки сценариев (scripts).

Основу связи между серверами и веб-клиентами на уровне приложений обеспечивает протокол HTTP (HyperText Transfer Protocol). Протокол HTTP используется в основном для гипертекста, но это общий протокол, который может поддерживать распределенную систему управления объектами данных. Важной особенностью протокола является независимость от платформы, различные компьютеры, которые обмениваются данными через HTTP, могут использовать различные операционные системы и гипертекстовые приложения.

Основные концепции, с которыми работает протокол, - это запрос и ответ: веб-клиент отправляет сообщение (запрос) на сервер. Сообщение содержит требуемый идентификатор ресурса, заданный в форме URI (унифицированного идентификатора ресурса), используемый метод доступа, а также серию метаинформации, которая может быть полезна для сервера. Ответ сервера включает в себя код, указывающий состояние сервера после интерпретации запроса, пояснительное сообщение для переданного кода состояния, метаинформацию, которая будет обрабатываться клиентом, и, возможно, контент (запрошенный ресурс).

Фактический стандарт взаимодействия веб-клиентов с веб-серверами, общий интерфейс шлюза в настоящее время имеет версию 1.1. Программа CGI, обычно называемая сценарием, выполняется на сервере WWW либо явно, вызывается изнутри страницы специальной директивой, либо при приеме информации, содержащейся в полях интерактивной формы или координатах конфиденциальной области. CGI обеспечивает интерактивность веб-страниц, HTML-документы могут динамически изменять свое содержимое и обеспечивать сложную обработку данных. Программы CGI также могут поддерживать аутентификацию пользователей на стороне сервера.

Программы CGI могут быть написаны на любом языке, интерпретированы в случае PHP, Perl, Python, TCL или скомпилированы, как в случае C и Delphi.

При разработке сценария CGI необходимо соблюдать определенные основные правила:

* программа записывает данные для передачи в веб-браузер на стандартный вывод
* программа генерирует заголовки, которые позволяют веб-серверу правильно интерпретировать вывод скрипта

Большинство скриптов CGI предназначены для обработки данных, вводимых в формы. Форма определяется в XHTML с использованием специальных маркеров для отображения содержимого и ввода данных клиентам, а сценарий, вызываемый и выполняемый веб-сервером, принимает содержимое этой формы и обрабатывает его, возможно, возвращая результаты в браузер.

3.3 Концепт PHP

PHP, известный в более ранних версиях как PHP / FI (Personal Homepage / Form Interpreter), изначально задумывался как простое приложение CGI для интерпретации HTML-формы и обработки программы, написанной на -a Perl-язык, сценарий оболочки, выполняемый на сервере. Интерфейс CGI требовал разрешения для запуска программ на сервере, что приводило к лазейкам в безопасности, а также к отделению HTML-документов от программы, обрабатывающей данные.

PHP (в текущей версии PHP 5.0) - это мощный пакет, который предоставляет доступный программный язык в файле HTML, язык, подобный Perl или C, плюс поддержку для управления базами данных на диалекте SQL (dBase, Informix, MySQL, mSQL, Oracle, PostgreSQL, Solid, Sybase, ODBC и т. д.) и доступ к гипермедийным системам, таким как Hyperwave. PHP также поддерживает загрузку файлов с клиентского компьютера: загрузка (стандарт, предложенный Э. Небелем и Л. Масинтером из Xerox, описанный в RFC 1867) и обеспечивает поддержку файлов cookie (механизм хранения данных в клиентском браузере для идентификации пользователей ), предложенный Netscape). Эти приложения доступны бесплатно в Интернете для сред Unix и более новых для сред Windows (включая исходные коды), интегрируясь в популярный сервер Apache.

История PHP начинается в конце 1994 года, когда Расмус Лердорф разработал первую версию в качестве личного проекта. PHP был обнародован в начале 1995 года как Personal Home Page Tools, будучи упрощенным парсером, который интерпретировал несколько макросов, которые могли быть включены в документы HTML, обеспечивая доступ к веб-страницам или доступ к гостевой книге. ). Парсер был переписан в середине того же года и получил название PHP / FI 2.0, где FI было еще одним приложением, написанным Расмусом Ленфордом, интерпретатором HTML-форм. Также была добавлена ​​поддержка баз данных mSQL, и PHP / FI начал пользоваться успехом, будучи доступным бесплатно в Интернете. По оценкам, в конце 1996 г. не менее 15 000 веб-сайтов использовали PHP / FI, а в 1997 г. их число составляло 50 000.

Программисты Зеев Сураски и Энди Гутманс переписывают синтаксический анализатор PHP, и новое приложение формирует ядро ​​PHP версии 3.0, которое включает некоторые из старых исходных кодов PHP / FI 2.0. Относительно недавно, в начале 2000 года, был выпущен PHP 4.0, использующий мощный механизм сценариев Zend и обеспечивающий поддержку не только сервера Apache, но и других веб-серверов. Кроме того, PHP 4.0 предлагает возможность доступа к XML-документам через DOM.

По оценкам, количество сайтов, в настоящее время использующих PHP, превышает один миллион. В Интернете уже существует множество приложений и утилит, разработанных на PHP, которые также сгруппированы в так называемом PEAR (репозиторий расширений и надстроек PHP).

PHP - это язык сценариев, который работает с веб-сервером.

Обратите внимание на разницу с другими языками сценариев CGI, написанными на таких языках, как Perl или C: вместо написания программы с множеством командных строк, наконец отображаемых на странице HTML, вы напишете страницу HTML с кодом, включенным в выполнить точное действие.

Код PHP находится между специальными начальным и конечным тегами, которые позволяют пользователю переключаться из «режима HTML» в «режим PHP». По сравнению с другими языками сценариев, такими как Javascript, код PHP выполняется на веб-сервере, а не в браузере (веб-клиенте). В результате PHP может получить доступ к файлам, базам данных и другим ресурсам, недоступным для JavaScript. Они содержат богатые источники динамического контента, привлекающие посетителей. Если на веб-сервере есть аналогичный сценарий, клиент получит только результат выполнения сценария, без какой-либо возможности доступа к коду, создавшему результат. Вы можете настроить свой веб-сервер для обработки (анализа) всех файлов HTML как файлов PHP. Таким образом, нет способа отличить страницы, которые создаются динамически, от статических страниц.

Язык PHP имеет те же функции, что и другие языки, позволяя вам писать сценарии CGI, собирать данные и динамически создавать веб-страницы или отправлять и получать файлы cookie. Отличное качество и самое большое преимущество языка PHP - это поддержка большого количества баз данных. Создать динамическую веб-страницу с базой данных чрезвычайно просто. Следующие базы данных поддерживаются PHP: Соответствие (см. Таблицу 1) PHP поддерживает множество типов баз данных, что является для него большим преимуществом. Язык PHP также поддерживает различные серверные службы, использующие такие протоколы, как IMAP, SNMP, NNTP, POP3 и HTTP. Каждая программа PHP включает две специальные строки, которые указывают серверу PHP, что текст между двумя строками состоит из инструкций PHP.

<? php?>

В основном эти строки можно уподобить обложкам книги, что сохраняет единство программы PHP. Программы PHP выполняют три категории основных операций:

получить данные от пользователя

выполняет обработку данных, соответственно получает доступ к данным, хранящимся в файлах и базах данных, и манипулирует ими

отображает данные, чтобы пользователь мог их просматривать.

Чтобы запустить сценарий, вы должны сначала загрузить его на сервер, где установлен PHP. Удаленный доступ к серверу Linux или Unix может осуществляться через протоколы Telnet или SSH, если вы ранее получили разрешение на использование этих протоколов от системный администратор для конкретного пользователя. Если используется сервер Microsoft Windows, расположенный в той же локальной сети, что и рабочая станция, администратор сети может выделить для этой цели файловый раздел. Чтобы загрузить скрипт по протоколу FTP, системный администратор должен предоставить авторизованным пользователям следующую информацию:

* имя хост-сервера
* имя пользователя и пароль для входа по протоколу FTP
* каталог, в котором должны быть найдены скрипты PHP
* унифицированный указатель ресурсов (URL), который необходимо использовать для получения доступа к сценариям PHP. После успешной загрузки сценария PHP он запускается в исполнение, открывая веб-браузер (клиент) с URL-адресом, связанным с сценарием PHP.

3.3 Apache Web - Server

Согласно исследованию Netcraft Web Server, веб-сервер Apache является основой более 60% доменов. Интернет-провайдеры и компании, предоставляющие хостинг веб-приложений, часто обращаются к Apache из-за его поддержки многих платформ, простых в реализации возможностей виртуального хостинга и его возможностей, расширяющих его возможности.

Функция Apache Portable Routine (APR) оптимизирует такие возможности, как управление процессами для каждой операционной системы, но позволяет самому серверу игнорировать различия, связанные с платформой. Это улучшило производительность и стабильность реализаций Windows за счет удаления эмулятора. Дополнительные APR также доступны веб-разработчикам, которые пишут кроссплатформенные программы на C, который быстрее интерпретируемых языков, таких как PHP и Perl, обычно используемых для веб-разработки.

К этому добавляется тот факт, что сам сервер не зависит от протоколов. Хотя это в первую очередь HTTP (веб-сервер), он предназначен для поддержки таких альтернатив, как FTP. Такая структура упрощает администрирование и снижает риски безопасности. Важной частью привлекательности Apache является его универсальность. Его открытый API позволял разработчикам писать различные модули, которые меняли их поведение. Если на сервере отсутствует функция, которая нужна сайту, велика вероятность, что где-то будет доступен подключаемый модуль.

Основными функциями веб-серверов являются прием анонимных запросов от клиентов и предоставление информации таким образом, чтобы они были эффективными и быстрыми. Фактически, веб-сервер - это демон, который принимает HTTP-совместимые соединения, отвечая на запросы, полученные от клиентов. Для предоставления HTTP-сервисов в системе должен быть установлен сервер Apache (обычно это пакет RPM в Linux или исполняемая программа .exe в Windows) и демон httpd должен быть включен. Apache - это модульная система, состоящая из базового сервера и нескольких модулей, которые динамически загружаются аналогично работе модулей в ядре Linux.

Apache можно настроить с помощью графического интерфейса apacheconf (инструмент настройки Apache). Основной файл конфигурации - это http-conf и обычно находится в каталоге / etc / httpd (в версиях Linux или Unix).

В некоторых случаях необходимо ограничить доступ к определенным документам путем аутентификации по имени пользователя и паролю или в зависимости от адреса компьютера веб-клиента.  
Для аутентификации пользователей мы проделаем два шага:

* создается файл, содержащий имена и пароли пользователей, которые будут иметь доступ к определенным данным на веб-сервере (в частности, Apache)
* сервер настроен на установку того, какие ресурсы будут защищены и каким пользователям разрешен доступ к ним, после ввода действительного пароля

Apache также предлагает возможность обслуживания нескольких веб-сайтов одновременно, другими словами, виртуальный хостинг (виртуальный хостинг). Существует два метода реализации виртуального хостинга: первый на основе имен и второй на основе IP-адресов. Виртуальные машины на основе адресов используют IP-адрес подключения для определения правильной виртуальной машины. Таким образом, для каждого виртуального хостинга на основе имени виртуальная машина определяется на основе ее имени.

Виртуальный хостинг на основе имен проще реализовать, и его использование рекомендуется. Чтобы использовать услугу виртуального хостинга, необходимо сначала установить IP-адрес и порт для сервера, который будет принимать запросы для этой виртуальной машины.

3.5 СУБД. MySQL

MySQL - это бесплатный сервер баз данных с открытым исходным кодом, разработанный шведской компанией MySQL AB, который предлагает реальную надежность и преимущества. Красота сервера MySQL в его фундаментальной простоте. Администраторы компании и разработчики баз данных пришли к выводу, что MySQL активно и лояльно развивается, не говоря уже о с трудом заработанной репутации безопасной базы данных. MySQL работает на основе модели клиент / сервер.

Любая машина, которая хочет обрабатывать запросы к базе данных MySQL, должна запускать сервер MySQL (mysqld), который отвечает за весь входящий / исходящий трафик с базой данных. Как и любой сервер, mysqld "слушает" на определенном порту (3306) любые "клиентские" запросы на соединение - любое приложение, которое отправляет запросы к базе данных через mysqld. Этот клиент может быть простым сценарием PHP, который может отправлять запрос в базу данных через сервер MySQL или даже клиент mysql. Клиент mysql - это интерактивный интерфейс для отправки команд на сервер.

Модель безопасности, используемая MySQL, основана на имени пользователя, пароле, имени сервера (имени хоста) или IP-адресе и привилегиях, что аналогично общей модели, используемой системами Unix. В случае MySQL привилегии означают операции, которые будут разрешены для баз данных, таблиц или индексов, такие как SELECT, INSERT, UPDATE, DELETE, CREATE, DROP

Данные являются предметом большинства операций обработки, а системы управления базами данных предоставляют самые сложные и мощные средства для работы с данными. PHP включает библиотеку функций, которая обеспечивает интерфейс для системы управления базами данных MySQL. Используя эти функции, программа PHP может получать доступ к резидентным данным в базе данных MySQL и изменять их.

Большинство взаимодействий с базой данных MySQL можно выполнять в соответствии с простой последовательной моделью:

* открывается соединение с сервером MySQL
* указать базу данных, к которой получен доступ
* Выдаются SQL-запросы, получается доступ к результатам запроса, выполняются не-SQL операции
* закрытие соединения с сервером MySQL

С точки зрения языка PHP, есть две категории SQL-запросов:

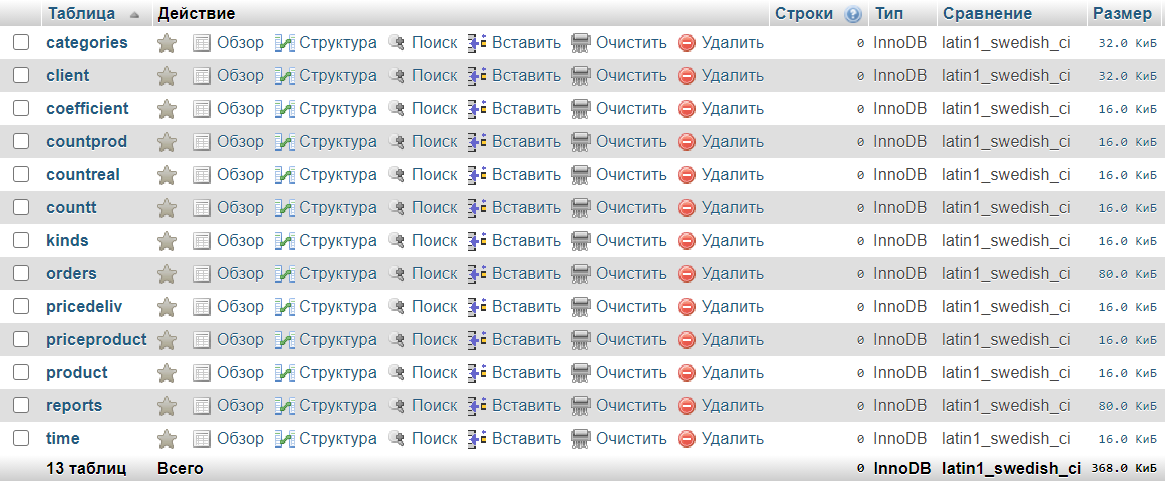
* SELECT запросы, которые возвращают строки таблицы
* Запросы UPDATE, INSERT и DELETE, которые не возвращают строки таблицы

Обе категории запросов выдаются с помощью функции mysql\_query (), но проверка и обработка двух категорий результатов запроса обрабатываются совершенно по-разному.

Глава 4 Структура базы данных Курсовой Работы

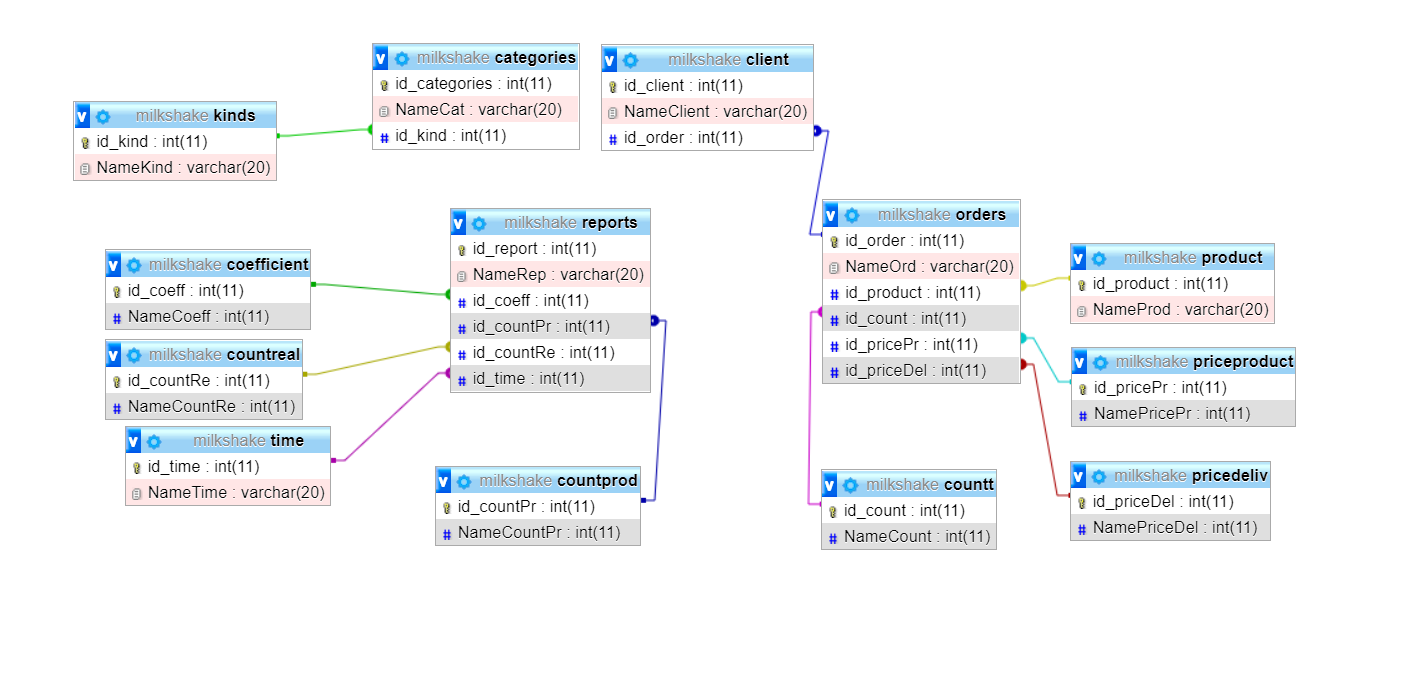
4.1 Таблицы БД

\*Пример созданной Базы Данных в PhpMyAdmin.

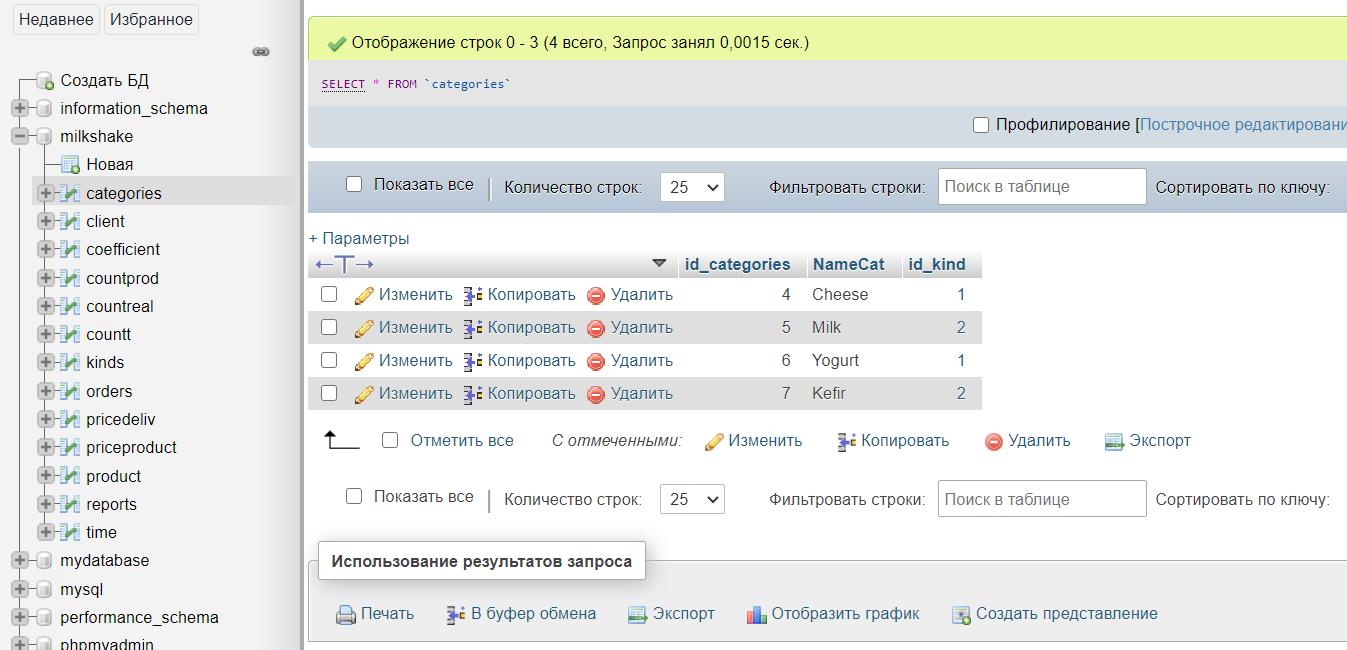


Отображенная на локальном сервере (localhost) база данных с созданными таблицами. Определяется множество функций, к которым можно прибегнуть прямо из php кабинета: просмотреть, удалить, вставить, очистить и др.

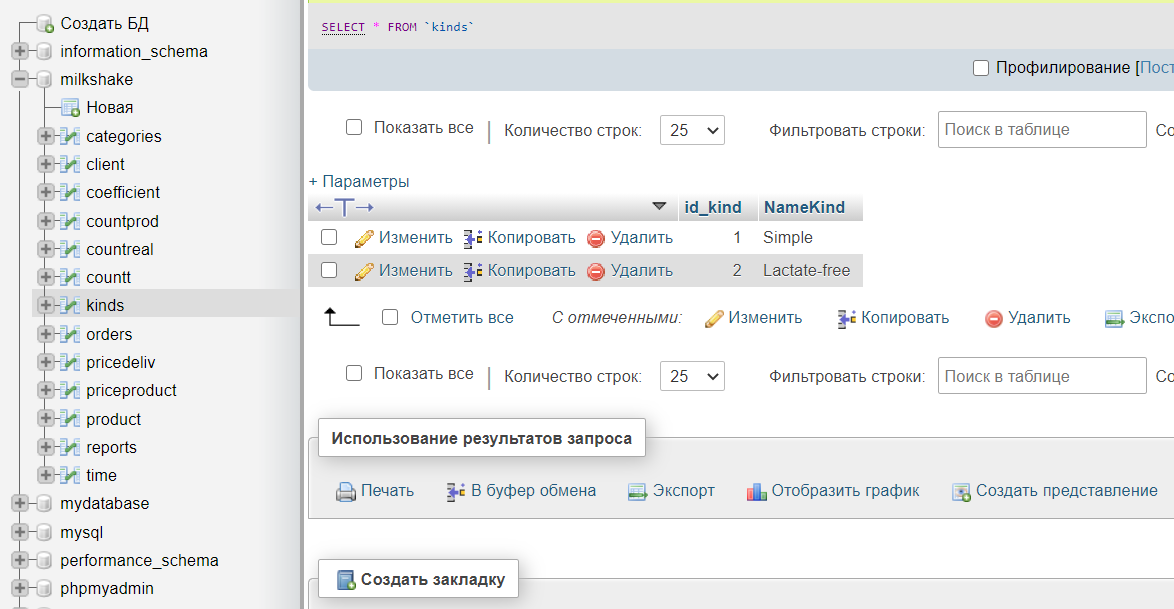
4.2\*Структура Базы Данных в Designer 🡪 PhpMyAdmin



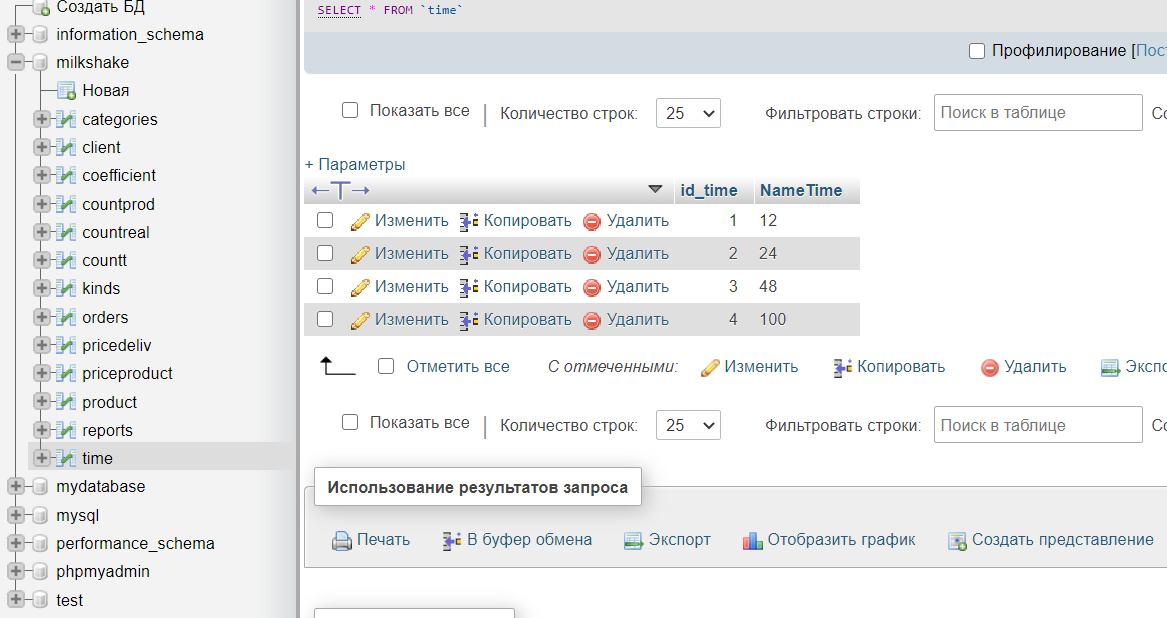
4.3 Таблица Категорий



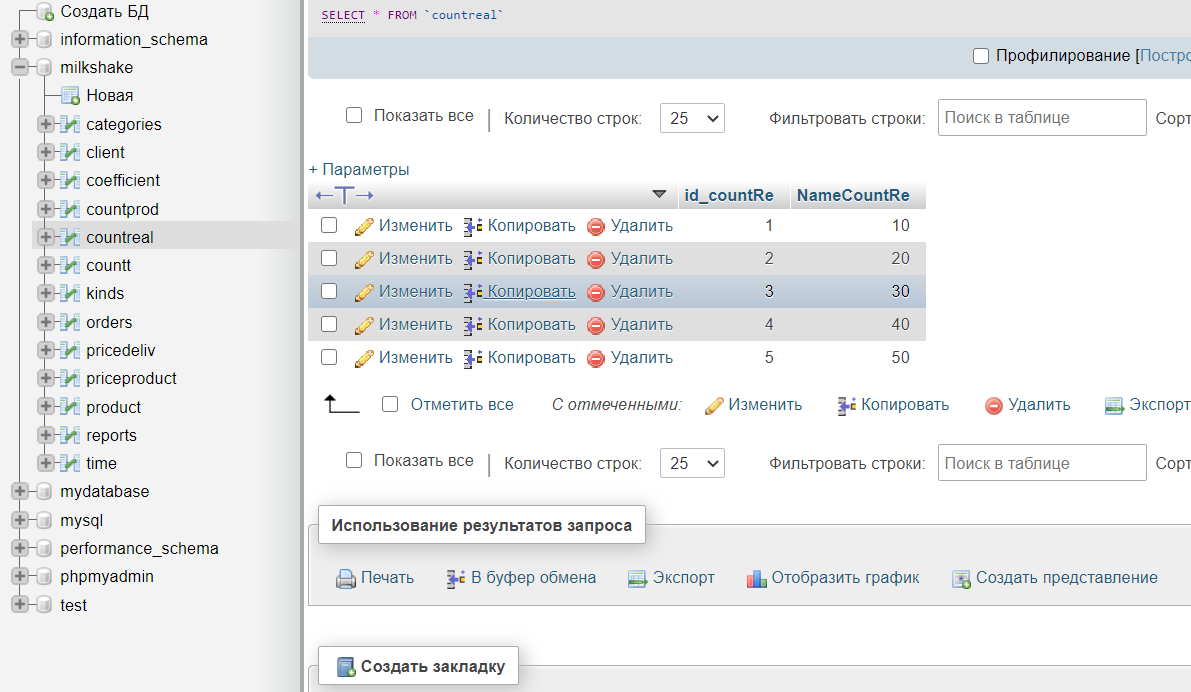
4.4 Таблица разновидностей товара



4.5 Таблица с показателем Времени



4.6 Таблица Реализованного товара



4.7 Таблица количества закупаемого товара

