Некоторая информация по установке и запуску комплекса программы

Работа с базами в PhpMyAdmin идёт с теми параметрами, которые указаны в подключаемых файлах pdoshop.php и pdobank.php. Эти же параметры указаны и в файлах создания таблиц БД: createshoptables.sql и createbanktable.sql. Поэтому, если у вас параметры в PHP другие, их необходимо поменять на ваши в указанных файлах прежде установки и запуска программы.

Программа разрабатывалась в версии XAMPP 7.2.26 в Linux Mint 19.2 Xfce 64-bit.

Прежде всего надо подключиться в терминале Линукса к MySQL и создать таблицы баз данных. Для этого нужно учесть путь к сокету MySQL. На разных компьютерах этот путь может быть разным (проверяйте свой путь):

```
sudo mysql -S '/opt/lampp/var/mysql/mysql.sock'
```

Далее необходимо учесть путь к самой программе, который также может отличаться на разных компьютерах (проверяйте свой путь):

```
source /opt/lampp/htdocs/Codes/ShopCart/createshoptables.sql;
```

```
source /opt/lampp/htdocs/Codes/ShopCart/createbanktable.sql;
```

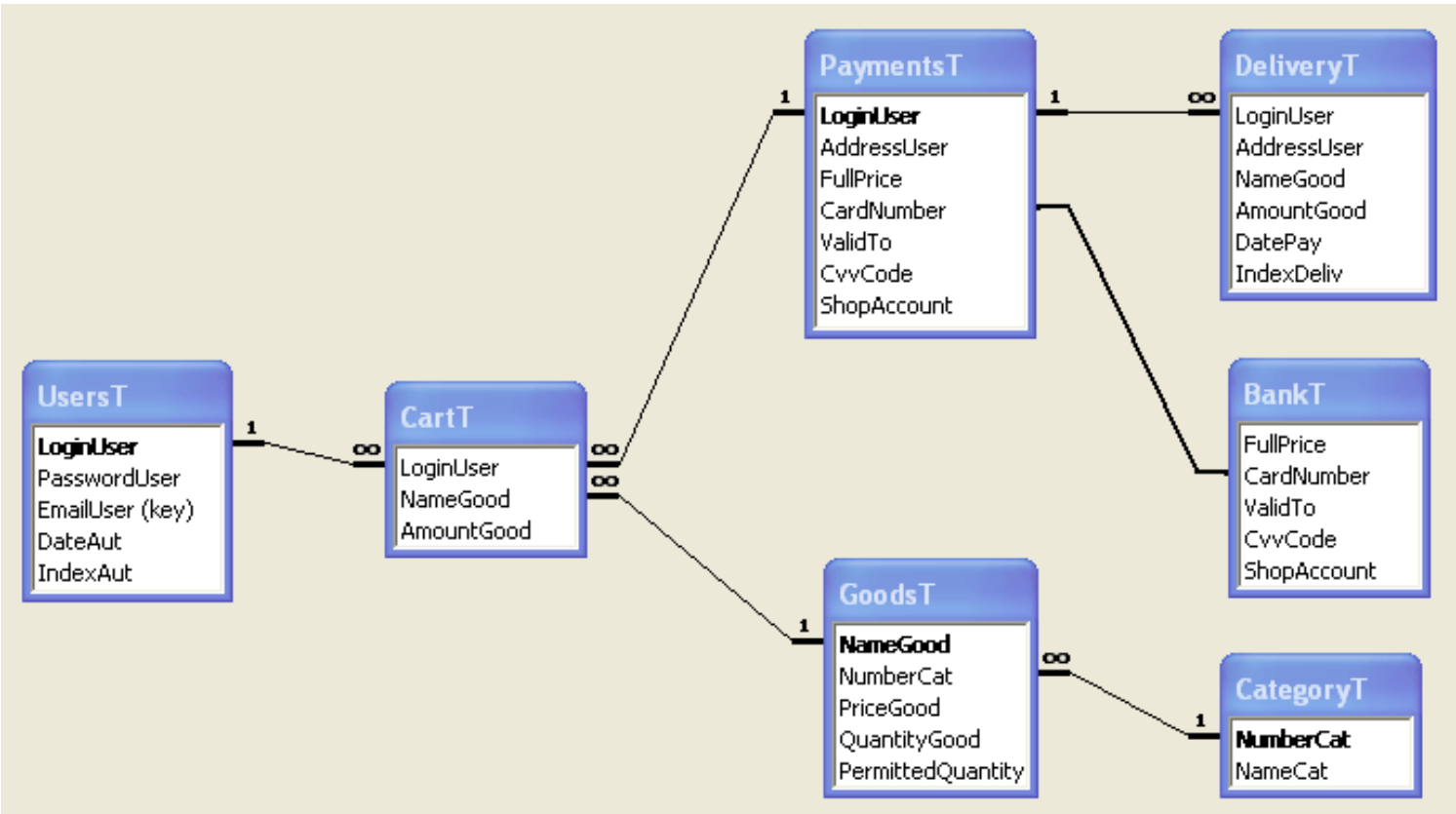
Прежде начала работы в таблицы UsersT, CategoryT и GoodsT необходимо внести какую-то информацию. Информация вносится, например, с помощью:

```
source /opt/lampp/htdocs/Codes/ShopCart/insertinformation.sql;
```

Поле DateAut в UsersT первоначально для всех пустое, не содержит значений; в поле IndexAut первоначально для всех записаны нули.

Первый файл, который открывается в браузере, имеет имя loginform.html.

Схема базы данных



Пароли доступа для указанных примеров клиентов магазина

romashka12 — 7jgkl689

listik — hrkith9

karandash — ui5y9800lp

Общее описание работы программы корзины магазина

Фактически здесь две базы данных. База банка состоит только из одной таблицы (так как здесь не стоит задача разработать базу банка, а разрабатывается только корзина магазина). Эта таблица — BankT. (Суффикс "T" введён для удобства и означает таблицу). Таблица BankT нужна для проведения и подтверждения транзакции платежа. Остальные шесть таблиц — это база корзины Интернет-магазина:

1. UsersT — таблица зарегистрированных и авторизованных пользователей в магазине. Факт авторизации показывается с помощью полей DateAut (дата и время авторизации с точностью до секунды), IndexAut (авторизован ли данный пользователь: 1 — авторизован, 0 — не авторизован).
2. CartT — таблица корзины, собственно, корзина покупок, показывает сколько и какого конкретно товара данный пользователь выбрал. Если пользователь не оплатил выбранные товары и вышел из магазина, они сохраняются в CartT до следующей авторизации этого пользователя.
3. CategoryT — таблица категорий товаров. Категории товаров введены просто для удобства, соответствуют, например, отделам в обычном магазине в реале.
4. GoodsT — таблица товаров по данной категории.
5. PaymentsT — так сказать, кошелёк покупателя. Таблица, где хранится общая сумма покупки и данные банковской карты покупателя. Эта информация, вообще говоря, секретна, поэтому данные для каждого покупателя появляются в этой таблице (вводятся покупателем), когда покупатель дошёл до стадии оплаты и удаляются из таблицы, когда банк возвращает результат оплаты: прошла оплата или нет. Значение поля ShopAccount вносится в самой программе: это номер счёта магазина в банке, то есть покупатель его не вносит в форму.

   Кроме того, поскольку PaymentsT связана с CartT связью "один-ко-многим" по полю LoginUser, то в CartT вносятся записи только тогда, когда в PaymentsT уже внесена запись с данным LoginUser и пока пустыми (NULL) значениями остальных полей. То есть внесение записей в CartT — это транзакция БД, состоящая из двух запросов: внесение LoginUser в PaymentsT, затем внесение для данного LoginUser данных в CartT.
6. DeliveryT — таблица службы доставки. Сюда перемещаются после успешной оплаты товары из корзины. На них устанавливается единая дата платежа DatePay, которая в совокупности с LoginUser позволяет различать разные корзины, наборы товаров, набранные в разное время одним и тем же покупателем.

Теоретически, таблица PaymentsT может быть из базы удалена. Данные для оплаты можно передавать сразу в банк, а потом, если оплата прошла успешно, из корзины CartT товары — сразу в службу доставки DeliveryT. Однако такая схема

представляется несогласованной, не обеспечивающей целостность данных, так как между CartT и DeliveryT будет иметься связь вида "многие-ко-многим". Промежуточная таблица PaymentsT позволяет избежать этого и упрощает процесс написания программы. Слишком большую избыточность таблица PaymentsT не создаст, поскольку временно хранит только корзины тех пользователей, которые в текущий момент дошли до стадии оплаты покупки. В остальное время эта таблица хранит только LoginUser тех пользователей, которые в магазине хоть раз доходили до стадии наполнения корзины товарами.

Что касается связи "многие-ко-многим" между PaymentsT и BankT, то это фактически связь между совершенно разными базами данных, а не между таблицами в одной базе.

Данная программа выполняется для функций клиента магазина. Обслуживание корзины магазина администратором магазина может выполняться путём прямого обращения к представленной здесь базе. При этом логика нарушена не будет, так как естественно, что администратор имеет администраторские права для доступа и изменения записей в БД.

Первоначально в таблице UsersT существуют записи, состоящие из логинов (уникальное поле), хешей паролей и имейлов (уникальное поле). Это записи зарегистрированных в магазине покупателей.

Покупатель входит в магазин по форме авторизации. Форма авторизации — ввод данных: логин (или имейл) и пароль. Проверка существующего пользователя в базе в таблице UsersT. Если пользователь не существует, не зарегистрирован, то выдаётся сообщение "Вернитесь назад, попробуйте снова". Если пользователь существует (зарегистрирован) — для данного пользователя обновляется информация в поле даты/времени авторизации DateAut и обновляется индекс авторизованности IndexAut.

Для чего нужны эти значения DateAut и IndexAut? Дата и время последней авторизации показывает, например, администратору базы, сколько времени сохраняется для данного пользователя набранная им корзина. Например, пользователь в последний раз авторизовался год назад, набрал корзину товаров, тем самым отложив товары для себя в магазине, а потом вышел из магазина и за год больше не появлялся. Разумно ли держать товары отложенными и не оплаченными для данного пользователя столько времени? Разумеется, нет. Администратор, следовательно, может написать соответствующий запрос, регулярно запускаемый в базе магазина, и чистить корзины таких покупателей. Индекс авторизованности показывает сколько покупателей в данный момент авторизовано в магазине.

После успешной авторизации — доступ к категориям товаров, представленных переключателями. Категории товаров просто вводятся для удобства конечных пользователей магазина. Категории товаров хранятся в таблице CategoryT по номеру категории NumberCat и названию категории NameCat. Здесь же присутствуют кнопка корзины и кнопка выхода из магазина. При выходе из магазина новый вход в следующий раз начинается с авторизации. При нажатии на кнопку "Выход" — IndexAut приравнивается нулю, и выдаётся сообщение: "До новых встреч!"

> Показываются те категории товаров, для номеров которых NumberCat в таблице GoodsT существуют записи, у которых QuantityGood не равно нулю. Однако возможно, что пока клиент будет выбирать, в какую категорию товаров пойти, товары из этой категории уже будут распроданы другим покупателям.

После выбора категории происходит доступ к выбору товаров данной категории, представленных в таблице GoodsT. В этой таблице хранятся данные: названия товаров NameGood, номер категории, к которой относится товар NumberCat, цена единицы товара PriceGood, количество доступных в магазине единиц товара QuantityGood, максимальное число единиц товара, разрешённых для одномоментного заказа PermittedQuantity (описание PermittedQuantity и пояснение необходимости использования этого поля дано далее).

В случае, если товары по данной категории товаров уже распроданы, выдаётся соответствующее сообщение. Тогда пользователь может вернуться назад и выбрать другую категорию товаров. В принципе, такая категория либо пополняется администратором магазина, либо удаляется из таблицы категорий.

Выбор товаров осуществляется пользователем с помощью флажков и указания в соответствующих полях того количества единиц товара, которое пользователь хочет купить. По умолчанию в этих полях стоит 1. Пользователю показываются только те товары, у которых QuantityGood не равно нулю.

> Необходимо отметить, что выбор товаров флажками и полями, где указывается их количество, подразумевает взаимосвязь флажков и соответствующих им полей. Такая взаимосвязь в программе осуществляется набором массивов, через которые информация передаётся дальше в обработку. То есть, например, показано десять разных товаров данной категории. Это означает, что будет десять массивов значений установленных флажков и значений соответствующих флажкам полей. Каждый массив, вообще говоря, представляет собой пару «флажок-поле». Однако если флажок не установлен, то количество элементов данного массива равно 1 (только значение поля). Если значение поля не установлено, то в массив помещается пустое значение, пустой стринг «». Каждый массив пары флажок-значение таким образом состоит либо из одного элемента, когда флажок не установлен, либо из двух элементов, когда флажок установлен. Имя флажка — это поле NameGood, передаётся через значение флажка value.
>
> При этом количество доступных в магазине товаров (значения QuantityGood) пользователю не показываются. Такая политика представляется разумной, потому что иначе возможна атака на магазин со стороны недоброжелательного пользователя. Такой пользователь сможет добавить себе в корзину максимально доступное число товаров (здесь имеется в виду само значение QuantityGood), выйти затем из магазина (с сохранением корзины товаров, как говорилось выше), при этом не оплачивая выбранные товары. Такое действие заблокирует выбор этих товаров другими пользователями и фактически остановит работу магазина.
>
> С другой стороны, недоброжелательный пользователь может достаточно быстро подобрать такое значение количества заказываемых товаров, которое будет равно доступному количеству товаров в магазине. Технически делается это путём атаки с помощью "множителя-два". Суть процесса. Первоначально недоброжелательный пользователь вводит заведомо нереальное число товаров для добавления себе в корзину. Например, миллиард рубашек. Естественно, (гипотетическая) программа

магазина выдаст сообщение, что предоставить такое количество не может. Тогда этот пользователь уменьшит запрашиваемое количество в два раза (применит "множитель-два"; в данном случае — "одна вторая"; "два" применяется, когда процесс атаки идёт наоборот — в сторону увеличения), и запросит 500 миллионов. Ответ магазина — нет. Пользователь уменьшает ещё в два раза — 250 миллионов. Ответ магазина — нет. Пользователь уменьшает ещё в два раза. И так далее. Очень быстро наступит такой момент, когда магазин ответит — да. Например, пусть при 2000 ответ магазина — нет, а при 1000 — да. Уже даже здесь пользователь может существенно попортить работу магазина, забрав себе в корзину (и оставив её без оплаты), минимум, половину доступных товаров (забирает 1000, а доступно не более, чем 1999, так как на 2000 ответ магазина — нет). Нетрудно посчитать, что с миллиарда до тысячи единиц недоброжелательный пользователь пройдёт всего 20 итераций уменьшения в два раза. Если такой недоброжелательный пользователь напишет автоматическую программу, которая сделает подобные запросы к магазину, то выполнение этой программы займёт доли секунды, но после неё работа магазина практически остановится.

Однако недоброжелательный пользователь может пойти дальше и, получив доступное число товаров, находящееся в диапазоне $1000 <= x < 2000$, запросит у магазина $1000 + 500 = 1500$ товаров. Допустим, ответ будет "да". Тогда он запросит $1500 + 250 = 1750$ товаров. Здесь, допустим, "нет". Тогда $1750 — 125 = 1625$. И так далее. Надо полагать, логика действий такого пользователя понятна. Как этого избежать?

Для этого в таблицу GoodsT вводится дополнительное поле PermittedQuantity. В нём для каждого товара в магазине будет храниться то максимальное количество единиц конкретного товара, которое будет разрешено пользователю одномоментно заказать в магазине. Как это работает? Например, пусть некоего товара доступно в магазине $QuantityGood = 715$ единиц. Предположим, что для этого товара PermittedQuantity установлено в 23 единицы. Как только покупатель отправил себе в корзину 23 единицы этого товара, двадцать четвёртую единицу этого товара добавить ему не разрешается, пока он полностью не оплатит 23. Как только он оплатит 23, то сможет заказать двадцать четвёртую единицу, двадцать пятую и так далее, хоть ещё 23.

Для каждого товара значение PermittedQuantity вычисляется маркетологами магазина на основании их изучения рынка. Например. Пусть в магазине доступна для продажи тысяча телевизоров. Вряд ли стоить разрешать одномоментно покупателю заказать сто телевизоров по той простой причине, что практически никогда одномоментно современные магазины такое количество телевизоров не продают. Логично для такого товара, как телевизор, поставить PermittedQuantity равным 2, от силы, 3 единицы товара. Для каждого товара определить оптимальное значение PermittedQuantity — задача маркетологов.

PermittedQuantity, как правило, устанавливается для каждого товара один раз и далее не меняется. Поэтому может наступить такой момент,

что по мере распродажи, PermittedQuantity станет больше, чем QuantityGood. Эта ситуация совершенно нормальна. В нашем примере, где PermittedQuantity равно 3 телевизора, а магазин со временем распродал свои 1000 телевизоров так, что их на складе осталось 2, покупатель сможет эти 2 телевизора заказать одновременно, так как 2 не превышает PermittedQuantity.

На странице выбора товаров присутствуют ещё три кнопки: "Добавить в корзину", "Корзина", "Выйти из магазина" ("Выход"). Есть также возможность вернуться назад и выбрать ещё другую категорию и затем другие товары. Возврат назад осуществляется методами браузера (в каждом современном браузере уже реализовано как кнопка навигации).

При нажатии на кнопку добавления в корзину, выбранные товары добавляются в корзину, в таблицу CartT, которая хранит уникальный логин пользователя, название добавленного товара и количество единиц этого товара, которое хочет приобрести пользователь AmountGood.

Добавление в корзину осуществляется при условии, что значение в поле AmountGood в CartT меньше либо равно значению в поле QuantityGood в GoodsT и одновременно меньше либо равно PermittedQuantity. То есть пользователь покупает товаров не больше, чем есть в наличии в магазине и не больше, чем разрешено продать магазину одномоментно.

При добавлении в корзину уменьшается QuantityGood на соответствующее значение AmountGood. Тем самым как бы блокируются выбранные товары для заказа другими пользователями.

При добавлении также осуществляется проверка того, что пользователь хочет купить товары только в количестве натуральных чисел (вводит натуральные числа в соответствующие поля при выборе количества заказываемых товаров). То есть проверка, что не будет такого, что, например, пользователь захочет купить пять десятых магнитофона или минус три четверти кондиционера.

Переход в корзину по кнопке корзины. Для текущего LoginUser из CartT показывает выбранные товары с ценами и общую сумму платежа (или пустую корзину, если в CartT ничего нет). В корзине оплачиваются сразу все товары, в ней находящиеся. Возможность выбрать флажками и удалить часть товаров из корзины.

При удалении части товаров из корзины, QuantityGood в таблице GoodsT увеличивается на соответствующую величину.

Возможность вернуться назад кнопкой навигации браузера для изменения выбора. Кнопка выхода. При выходе выбранные товары сохраняются в корзине за данным пользователем (в таблице CartT). Кнопка оплаты товаров.

При нажатии на кнопку оплаты товаров появляется форма введения адреса доставки товаров, введения данных банковской карты (номер, срок действия и CVV-код). Кнопка «Оплатить». Также кнопка выхода из магазина.

Нажатие на кнопку "Оплатить" приводит к следующему.

Данные, которые ввелись на этом шаге, попадают в таблицу PaymentsT, которая содержит полную стоимость покупки FullPrice набора для корзины пользователя LoginUser, адрес доставки товаров AddressUser,

номер банковской карты покупателя CardNumber, срок действия банковской карты ("месяц-год") ValidTo, трёхзначный CVV-код на обороте банковской карты CvvCode, счёт, на который переводить деньги, счёт магазина в банке ShopAccount.

Далее эти данные передаются во вторую базу данных, в базу банка, в таблицу BankT. Банк своей собственной программой проверяет, может ли покупатель (владелец данной банковской карты) оплатить указанную сумму.

Реализация комплекса программы банка выходит за границы задачи написания корзины магазина, поэтому предлагается минимальный подход. Происходит подключение к таблице банка BankT с логином другого пользователя MySQL и возвращается успешная оплата картой покупателя, допустим, с вероятностью 75%. Соответственно, при 25% оплата не проходит. Реализовать такой случайный процесс можно так. Формируется случайное натуральное число в диапазоне от 1 до 100 включительно, а затем проверяется, находится ли это число в диапазоне от 1 до 25 включительно. Если находится, значит возвращается из банка в магазин значение, что оплата прошла неуспешно. Если не находится, значит — успешно.

В случае, если оплата прошла успешно, из таблиц CartT и PaymentsT информация перемещается (транзакцией баз данных, то есть сначала копированием, а затем удалением записей из таблиц CartT и PaymentsT) в таблицу DeliveryT.

Таблица DeliveryT — это таблица службы доставки, в которой в поле DatePay записывается дата и время платежа (единое для всей корзины данного пользователя), а поле IndexDeliv сначала равняется нулю, а потом, когда товары будут доставлены, предполагается, что в это поле сами сотрудники службы доставки введут значение 1, показывающее, что товары доставлены ими по назначению.

Когда оплата прошла успешно, выдаётся сообщение, что товары поступили в службу доставки и будут доставлены по указанному адресу в ближайшее время. Кнопка выхода из магазина.

Теоретически возможно, что после того, как товары поступят в службу доставки (в таблицу DeliveryT), клиент не совершит выход из магазина, нажав соответствующую кнопку выхода, а совершит возврат назад методами браузера на ранее пройденные страницы-этапы совершения покупки. Такой возврат назад позволит клиенту набирать новые товары в корзину, затем оплатить их, и новые товары попадут для того же клиента LoginUser в таблицу доставки DeliveryT. Но с другим единым временем платежа DatePay! Такая совокупность LoginUser и DatePay различает оплаченные корзины в магазине по покупателям и для каждого покупателя в отдельности, не смешивая их.

Кроме того, необходимо отметить, что в программном комплексе по возможности учтены различные нюансы, требующиеся по требованиям безопасности магазина, его товаров и денег клиента. Однако в жизни возможны ситуации, которые принципиально невозможно запрограммировать. Такие ситуации, как всем хорошо известно, называются форс-мажоры. Например. Допустим, вы в некотором онлайн-

магазине набираете корзину товаров, затем оплачиваете её, банк возвращает сообщение в магазин, что оплата прошла успешно, деньги с вашей карты списаны, и в этом момент прямо на сервер магазина падает метеорит. Как вы думаете: вы дождётесь доставки своих уже оплаченных товаров? :)

Ясно, что подобные форс-мажоры, к счастью, достаточно редки, и если всё-таки происходят, то необходимо решать вопросы не путём программы, а путём личных контактов клиента магазина с его администрацией. В приличных магазинах администрация открыта к общению и к контактам.

Это тоже следует учитывать, но запрограммировать это (написать программу реакции компьютера на падение метеорита :)) принципиально невозможно.

Что видит и делает пользователь в браузере

Первое окно браузера — это форма авторизации: логин или имейл плюс пароль и кнопка "Войти в магазин".

Если вход осуществлён успешно, то второе окно — это категории товаров (переключатели) и две кнопки: "Корзина" и "Выход из магазина".

При выборе некоторого переключателя и переходе далее, появляются наименования и цены доступных в магазине товаров по данной категории. Возле каждого наименования флажок, который можно отметить для выбора товара, а также поля для ввода количества товаров для покупки. Ещё три кнопки: "Добавить в корзину", "Корзина" и "Выход из магазина".

При нажатии на кнопку "Добавить в корзину", в случае успешного выполнения добавления выдаётся информация, что выбранные товары в корзину добавлены. Отсюда можно перейти в саму корзину, выйти из магазина или вернуться назад и выбрать ещё товары. Также корзина открывается при нажатии на кнопку "Корзина".

В корзине показываются выбранные для покупки товары и их количества, цены и общая стоимость покупки. Есть возможность флажками выбрать и удалить часть товаров из корзины. Две кнопки: "Выход" и "Оплатить покупку".

В случае если часть товаров из корзины удаляется, показывается обновлённая корзина.

При нажатии "Оплатить покупку" открывается форма, где вводятся: адрес и данные банковской карты. Две кнопки: "Выход" и "Оплатить".

При нажатии "Оплатить", если оплата прошла успешно, то данные товары попадают в службу доставки. Если оплата не прошла успешно, выдаётся соответствующее уведомление.

Далее — кнопка выхода из магазина. При её нажатии — сообщение "До новых встреч!"