Текстовое описание базы данных `ovoshi`.

Содержание:

Цели и задачи БД

База данных `ovoshi` призвана организовать систему хранения информации о продаваемых продуктах (фрукты/овощи), о заказчиках, поставщиках, а также о заказах. Предполагается некий сайт, через который можно оформить заказ как физическим лицам, так и юридическим (то есть подразумеваются и оптовые закупки и розничные). Коробки с фруктами и овощами хранятся на складах при требуемой температуре.

Физическая реализация

База данных создана с помощью языка sql. В ней есть 9 таблиц. Опишем каждую из них. Серым обозначаются колонки, имеющие внешнюю связь с другими таблицами.

`categories`

column	type	Def. value	nullable	extra	comments
Id	Int		NO	Auto_increment	
name	Varchar(128)		YES		Названия категорий товаров. Например: яблоко, груша и т.д

`temperature`

column	type	Def. value	nullable	extra	comments
Id	int		NO	Auto_increment	
max	tinyint		YES		Максимальная температура хранения
min	tinyint		YES		Минимальная температура хранения

Примечание: максимальная температура и минимальная всегда идут парой.

`sort`

column	type	Def. value	nullable	extra	comments
id	int		NO	Auto-increment	

name	Varchar(128)		YES		Название сорта (яблок, груш, огурцов и т.д)
price	Decimal(19,2)		NO		Цена за кг, руб
categorie_id	Int		NO		Номер категории из таблицы `categories`
temp_of_storage_id	int		YES		Id из таблицы `temperature` с минимальной и максимальной температурой хранения данного сорта товара

`supplier` – поставщики

column	type	Def. value	nullable	extra	comments
Id	Int		NO	Auto_increment	
name	Varchar(128)		NO		Имя поставщика или название организации поставшика
tel	Bigint unsigned		YES		Телефон для связи с поставщиком
Created_at	Datetime	Current_timestamp	YES	Default_generated	
Updated_at	datetime	Current_timestamp	YES	Default_generated	

`users` – зарегистрировавшиеся покупатели

column	type	Def. value	nullable	extra	comments
Id	Bigint		NO	Auto_increment	
Firstname	Varchar(50)		NO		Имя
Lastname	Varchar(50)		YES		Фамилия
Tel	Bigint unsigned		YES		Телефон для связи
Email	Varchar(120)		YES		
Address	Varchar(256)		YES		
Personal_disc	Int		YES		Персональная скидка
Date_of_birth	Date		YES		Дата рождения
Fav_prod_id	Int		YES		Любимый продукт
Created_at	Datetime	Current_timestamp	YES	Default_generated	
Updated_at	Datetime	Current_timestamp	YES	Default_generated	

`storages` – склады где хранятся коробки с товарами

column	type	Def. value	nullable	extra	comments
Id	Int		NO	Auto_increment	
Address	Varchar(256)		NO		Адрес склада
tel	Bigint unsigned		YES		Телефон для связи

`box` – пронумерованные ящики, в которых по отдельности хранятся продукты каждого сорта

column	type	Def. value	nullable	extra	Comments
id	Bigint		NO	Auto_increment	
Weight	Decimal(5,3)		YES		Вес ящика (может меняться)
Sort_id	Int		NO		Какой сорт он в себе содержит – id из таблицы `sort`
Supplier_id	Int		YES		Кто привёз этот ящик – id из таблицы `supplier`
Date_of_arrival	Date		YES		Когда был привезён
status	Varchar(50)		YES		Статус: пустой (empty), наполнен (filled), испорчен (broken).
Storage_id	int		YES		На каком складе лежит – id из таблицы `storages`

`deliveries` – таблица заказов

column	type	Def. value	nullable	extra	comments
Id	Int		NO	Auto_increment	
User_to_deliver	Bigint		NO		Пользователь, который оформил заказ – id из таблицы `users`
Status	Varchar(50)		NO		Статус заказа: собирается (in progress), в пути (on the way), доставлен (delivered).
Operation_ids	Tinytext		YES		Массив (по факту строка) из номеров операций из таблицы `operations` (см. примечание)

Price	Decimal(19,2)		YES		Стоимость заказа (вычисляется)
Created_at	Datetime	Current_timestamp	YES	Default_generated	
Updated_at	datetime	Current_timestamp	YES	Default_generated	

Примечание:

Поскольку наш предполагаемый сайт продаёт и оптом и в розницу, у нас должна быть возможность как продавать целыми коробками, так и маленькими порциями (по 100 грамм, например). Кроме того, если один заказчик покупает и яблоки, и груши, нам надо уметь это как-то записывать. (Нам так же важно помнить из какой коробки брали товары, чтобы уметь отсеивать брак по жалобам.) Скорее всего, для таких целей лучше всего подходит monogodb с его возможностью хранить массивы в формате json. Но у нас mysql. Поэтому будем писать процедуру (см. файл deliveries_proc.sql), в которой заказ будем разбивать на операции взятия некоторого количества товара из коробки с возможностью забрать целую коробку или дробное число коробок. Каждая операции соответствует взаимодействию с одной определённой коробкой. Информация об операциях в таблице `operations`. Одном заказу может соответствовать несколько операций.

`operations` – в некотором смысле, вспомогательная таблица, необходимая для формирования заказа и запоминания из каких коробок брали товары

column	type	Def. value	nullable	extra	comments
Id	Bigint		NO	Auto_increment	
Box_id	Bigint		NO		Номер коробки, из которой взяли продукт – id из таблицы `box`
Delta_M	Decimal(5,3)		YES		Какую массу взяли из коробки; обязательно должна быть меньше массы исходной коробки; вычисляется.

Хранимые процедуры и триггеры

Самая важная процедура хранится в файле deliveries_proc.sql. Она заполняет таблицы `deliveries` и `operation`. Допустим нам из вне пришёл запрос, записать в таблицу заказ от юзера с данным id (@utd). В этом заказе есть два сорта с разными весами (@sort1, @weight1, @sort2, @weight2). (Сделана реализация только для заказа из двух сортов. Но можно усложнить и сделать для произвольного.)

Вызываем процедуру deliveries(). В неё проверяем что на безе вообще есть в нужном количестве запрашиваемые продукты и если чего-то не хватает выводим сообщение. Если всего хватает записываем запрос с помощью процедуры fill_deliveries().

Процедура fill_deliveries() запрашивает стоимость заказа из функции f_proce_num() и список операций по изъятию товаров из коробок из функции f_opIds().

Функция f_opIds() возвращает строку (TINYTEXT) – список id операций из таблицы `operations`. Она же эту таблицу и заполняет. Подробное описание по строчкам можно найти в комментариях в самом файле с функцией (deliveries_proc.sql). В общем говоря, она проверяет можно ли взять вес ь необходимый вес из первой подходящей коробки, если да, то берём вес из неё, вносим изменения в таблицу box (вес и, возможно, статус) и вносим запись в `operations`. Если нет, то опустошаем первую подходящую коробку и ищем следующую (делая соответствующие записи в таблицы operations, box). Список id совершенных операций записывается в строковую переменную operations_list с помощью concate().

Сделаны два триггеры – файл triggers.sql.

Триггер для тех, кто при заполнении будет путать, где столбик для минимальной температуры, а где для максимальной – trigger_temp. Если заполняющий перепутает два столбика, то база данных сама поменяет значения местами.

Триггер на добавление пользователя чтобы была корректная дата рождения – trigger_users. Если дата рождения больше сегодняшнего числа, то вернётся ошибка и заполнения не произойдёт.

Характерные выборки

Все содержится в файле выборки.sql

1) Найти все коробки данного заказа
   Это по большей части не выборка, а ещё одна процедура. Но её задача заключается в выборке. По номеру заказа мы можем получить строчку, в которой записаны id операций из таблицы operations. В таблице operations содержится информация о коробке, с которой совершалась операция. Но мы не можем делать select напрямую, поскольку мы не сможем делать поиск по id операций так как они записаны в строчку такого вида ‘ , 1, 2’. Поэтому сначала нам нужно “разобрать” эту строчку на составляющие используя substring() и length(), а затем уже искать коробку по номеру операции. Процедура proc_op(*строка номеров операций из deliveries, длина этой строки*) возвращает записанные через запятую номера коробок.

2) Сколько килограмм какого продукта есть в наличии (суммарный вес, название сорта, название категории)
3) Самые любимые пользователями продукты и кем они доставляются (число покупателей, выбравших этот продукт; категория продукта; сорт продукта; имя поставщика)

Представления

Файл представления.sql

1) Представление weight_sort даёт информацию о том, сколько килограмм какого сорта есть в наличии. Это представление пригодилось при проверке можем ли мы выполнить заказ или нет в процедуре deliveries(). (Два столбца – вес, id сорта)
2) Какие сорта при какой температуре хранить temperature_of_storage. Три столбца – перечисленные через запятую названия сортов (с названием соответствующей категории), минимальная температура, максимальная температуры.