Здравствуйте. Сегодня решил сделать первую сессию информационной системы «Лопушок». Начну сразу с критериев оценивания и с того, что вообще надо сделать. Читая предметную область и понимаю, что информационная система «Лопушок» это про тебя, понимаешь, что надо переходить сразу во введение. Там сказано, что надо всего лишь создать подсистему для работы с продукцией: просмотр списка продукции, добавление/удаление/редактирование данных о продукции, управление списком материалов, необходимых для производства продукции. То е

**Дизайн базы данных ERD**

Название диаграммы ER соответствует требованиям 0.1

Основные сущности определены 1.2

Минус 20% за каждую ошибку

Отношения определены правильно с учетом отсутствующих объектов 1

Минус 20% за каждую ошибку

Все атрибуты (поля) рассмотрены и определены 2

Минус 20% за каждую ошибку

Созданы ограничения на связи между сущностей, отражающие характер предметной области 0.5

Минус 30% за каждую ошибку

Идентификатор в таблице присутствует 0.3

Минус 20% за каждую ошибку

Разработанная база данных находится в 3НФ (при наличии минимального необходимого набора) 0.9

Минус 0.3 балла за НФ

**Дизайн базы данных**

Таблицы и поля названы в соответствии с индустриальными стандартами

**Импорт данных**

Все данные загружены верно и в правильном формате 2

*Замечание: не сказано каким образом загружены*

Минус 10% за каждые 10% незагруженных данных

Все данные о материалах для изготовления продукции загружены верно и в правильном формате 2

Минус 10% за каждые 10% незагруженных данных

Все данные о материалах загружены верно и в правильном формате 1

Минус 10% за каждые 10% незагруженных данных

*Вопрос: как загрузить не в правильном формате?*

**Разработка базы данных**

Созданы таблицы необходимые для описанной подсистемы и реализуемого функционала 0.5

Минус 20% за каждую ошибку

Созданы поля в таблицах 0.6

Минус 20% за каждую ошибку

Созданы типы данных полей 0.7

Минус 20% за каждую ошибку

Созданы связи между таблицами 0.5

Минус 20% за каждую ошибку

Созданы первичные ключи

Минус 50% за каждую ошибку

То есть анализируя папку импорт можно прийти к выводу, что и какие таблицы нам нужны, а также какие связи существуют в подсистеме.

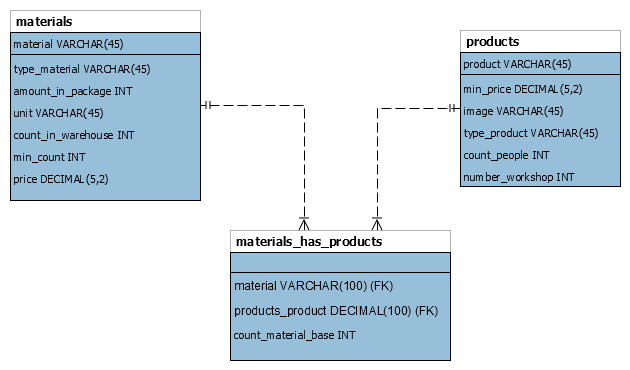


Рис. 1. Диаграмма

На рисунке 1 показана ERD – диаграмма, которая построена по анализу импорта исходных данных. Сразу видно, что тут нет стандартного идентификатора id, который обычно принято делать для удобства. Фрагмент исходных данных для связи таблиц materials и products говорит о том, что эти таблицы находятся в связи многие ко многим и целесообразней использовать в таких случаях автоикрементный id вместо полного названия первичных ключей продуктов и материалов. Но тут у нас возникает проблема, что таблицы не находится в 3 нормальной форме. Она даже не находится в первой нормальной форме (значения в первичном ключе не атомарные). Если мы будет уходить в такой модели построения таблицы (например, декомпозицией или сделаем составной первичный ключ), то нам придется менять структуру исходных данных, которые надо импортировать. По этой же причине мы не может сделать соединяющую таблицу из id, так как придется самим делать исходные данные по тем исходным данным, которые уже у нас есть.

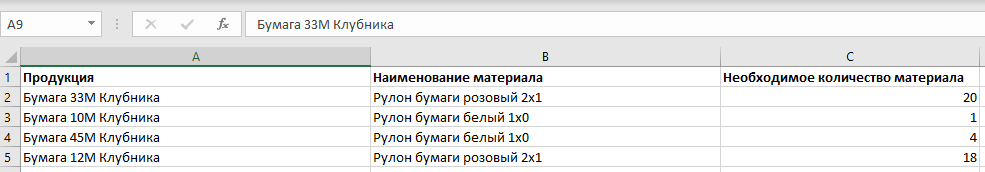


Рисунок 2 – Фрагмент исходных данных для соединяющей таблицы

Таким образом мы либо значительно усложним себе задачу, чтобы довести таблицы и исходные данные до 3НФ, либо потеряем специально 0.9 балла. Напомню, что критерий выглядит так: разработанная база данных находится в 3НФ (при наличии минимального необходимого набора). Минус 0.3 балла за НФ. Минимальный набор – это и есть те таблицы, исходные данные для которых представлены в первой сессии.

Далее мы может экспортировать ERD – диаграмму в git.

Теперь перейдем к разработке базы данных. Тут у нас есть два пути. Первый путь – это движение от модели ERD – диаграммы с помощью forward eingeneering формируем sql – скрипт, который можно выполнить и получить в базе данных таблицы. Естественно, Workbench сгенерирует много вспомогательного кода. Далее представлен скрипт

-- MySQL Workbench Forward Engineering

SET @OLD\_UNIQUE\_CHECKS=@@UNIQUE\_CHECKS, UNIQUE\_CHECKS=0;

SET @OLD\_FOREIGN\_KEY\_CHECKS=@@FOREIGN\_KEY\_CHECKS, FOREIGN\_KEY\_CHECKS=0;

SET @OLD\_SQL\_MODE=@@SQL\_MODE, SQL\_MODE='ONLY\_FULL\_GROUP\_BY,STRICT\_TRANS\_TABLES,NO\_ZERO\_IN\_DATE,NO\_ZERO\_DATE,ERROR\_FOR\_DIVISION\_BY\_ZERO,NO\_ENGINE\_SUBSTITUTION';

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

-- Schema test\_data

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

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

-- Schema test\_data

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

CREATE SCHEMA IF NOT EXISTS `test\_data` DEFAULT CHARACTER SET utf8 ;

USE `test\_data` ;

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

-- Table `test\_data`.`materials`

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

CREATE TABLE IF NOT EXISTS `test\_data`.`materials` (

`material` VARCHAR(45) NOT NULL,

`type\_material` VARCHAR(45) NOT NULL,

`amount\_in\_package` INT UNSIGNED NOT NULL,

`unit` VARCHAR(45) NOT NULL,

`count\_in\_warehouse` INT NOT NULL,

`min\_count` INT NOT NULL,

`price` DECIMAL(5,2) UNSIGNED NOT NULL,

PRIMARY KEY (`material`))

ENGINE = InnoDB;

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

-- Table `test\_data`.`products`

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

CREATE TABLE IF NOT EXISTS `test\_data`.`products` (

`product` VARCHAR(45) NOT NULL,

`min\_price` DECIMAL(5,2) UNSIGNED NOT NULL,

`image` VARCHAR(45) NOT NULL DEFAULT '/default/image.jpg',

`type\_product` VARCHAR(45) NOT NULL,

`count\_people` INT NOT NULL,

`number\_workshop` INT UNSIGNED NOT NULL,

PRIMARY KEY (`product`))

ENGINE = InnoDB;

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

-- Table `test\_data`.`materials\_has\_products`

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

CREATE TABLE IF NOT EXISTS `test\_data`.`materials\_has\_products` (

`material` VARCHAR(100) NOT NULL,

`products\_product` DECIMAL(100) NOT NULL,

`count\_material\_base` INT NOT NULL,

INDEX `fk\_materials\_has\_products\_products1\_idx` (`products\_product` ASC) VISIBLE,

INDEX `fk\_materials\_has\_products\_materials\_idx` (`material` ASC) VISIBLE,

CONSTRAINT `fk\_materials\_has\_products\_materials`

FOREIGN KEY (`material`)

REFERENCES `test\_data`.`materials` (`material`)

ON DELETE NO ACTION

ON UPDATE NO ACTION,

CONSTRAINT `fk\_materials\_has\_products\_products1`

FOREIGN KEY (`products\_product`)

REFERENCES `test\_data`.`products` (`product`)

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = InnoDB;

SET SQL\_MODE=@OLD\_SQL\_MODE;

SET FOREIGN\_KEY\_CHECKS=@OLD\_FOREIGN\_KEY\_CHECKS;

SET UNIQUE\_CHECKS=@OLD\_UNIQUE\_CHECKS;

Можно отредактировать данный скрипт под свои нужды. То есть нам надо создать каркас для импорта данных. Готовая база данных с таблицами представлена на рисунке 3.

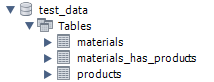


Рисунок 3 – База данных

Теперь остается правильно импортировать данные. Следует учесть, что предварительно нужно отредактировать данные для того, чтобы соотнести их с правильным форматом типа столбца, иначе можно просто не «переварить» данные. С помощью проб и ошибок, боли и муки, слабоумии и отваги удалось выяснить три пути импорта данных. Даже четыре пути. Четвертый путь – это для тех, кто не сдается.

Первый путь – использовать возможности оболочки MySQL Workbench. Здесь нас ждем множество приключений с кодировками, версиями MySQL сервера, версиями MySQL Workbench, а также ограничение на форматы импортируемых данных (csv и json). Это прекрасный путь у вас дома (это не точно), но не на экзамене. Я не доверяю этому пути.

Второй путь использовать чистый SQL – код, но в рамках Workbench. Это команда LOAD DATA. Здесь нам потребуется наша память, потому что команда не короткая. Важную роль здесь играет гибкость настройки, а также возможность игнорирования первой строки (вспоминаем импортируемые файлы). Также данная команда принимает больше форматов на импорт. Опять же здесь ключевое слово LOCAL (разрешает загружать данные локально). Если она не сработает, то надо прописывать в настройках учетной записи Workbench OPT\_LOCAL\_1.

LOAD DATA LOCAL INFILE 'data.txt' INTO TABLE *tbl\_name*

FIELDS TERMINATED BY ',' ENCLOSED BY '"'

LINES TERMINATED BY '\r\n'

IGNORE 1 LINES;

Пусть номер 3. Это как путь номер 2, только из-под командной строки. То есть вам вообще без разницы на все оболочки. Вы работаете чисто с mysql, как настоящий программист. Нюанс заключается в том, что надо при соединении с базой данных иметь исполняемый файл (здесь нам на помощь придут доблестный работники вычислительного центра и создадут где-нибудь тайный репозиторий с нужными файлами, в том числе mysql.exe). Но надо будет немного потрудиться и прописать все руками, начиная от соединения

mysql –local-infile -u root -h localhost -p password

Естественно, что параметры свои. Тут ключевая роль отдана опции –local-infile

Четвертый путь самый долгий и самый надежный. Просто использовать команду INSERT INTO построчно. Да, долго. Все равно вам редактировать файл. В конце концов, главное результат, а каким способом вы загрузили данные не важно. Главное не сколько ты падал, а сколько ты поднимался. Ну вы поняли).

Применение техники номер два представлено на рисунке 3.

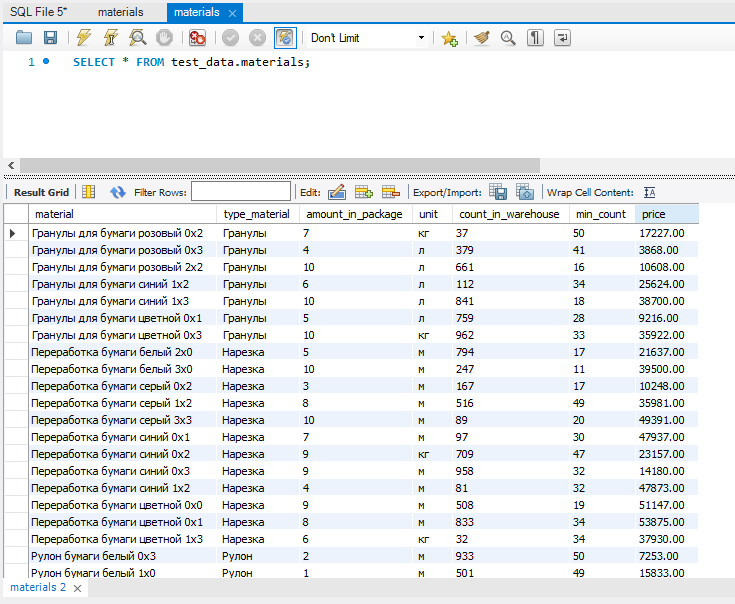


Рисунок 3. Импорт материалов

Здесь пришлось отредактировать поле price, чтобы сделать его атомарным, а также сделать тип decimal(10,2). В любом случае ошибки подскажут вам, что у вас не так при импорте. В данном случае, предупреждение.

Импорт материалов показан на рисунке 4

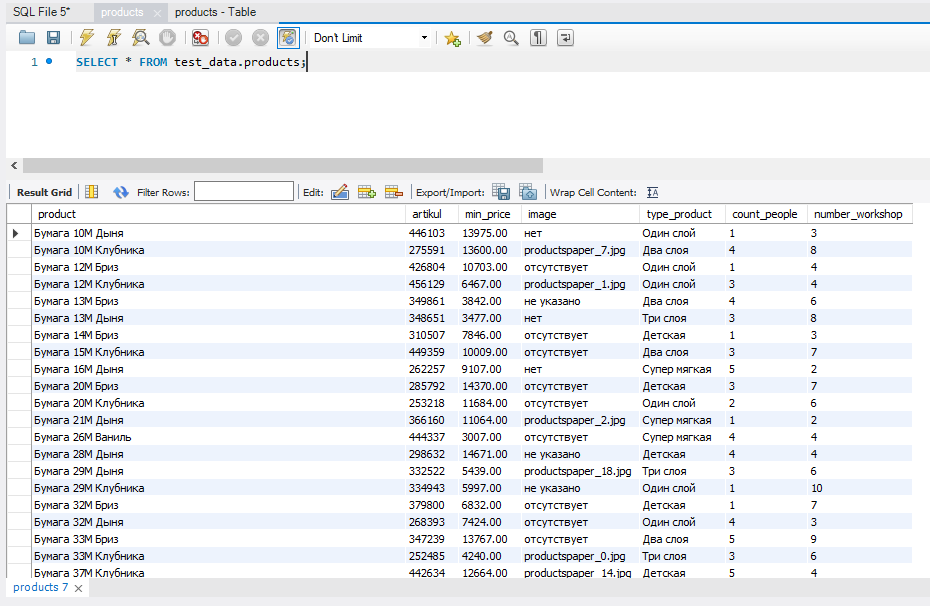


Рисунок 4. Импорт материалов

Изначально исходный файл был форматом csv. Стоит учесть тот факт, что разделитель в таком случае в команде LOAD DATA стоит поменять на ‘ ; ’.

Далее на рисунке 5 показа импорт связующей таблицы.

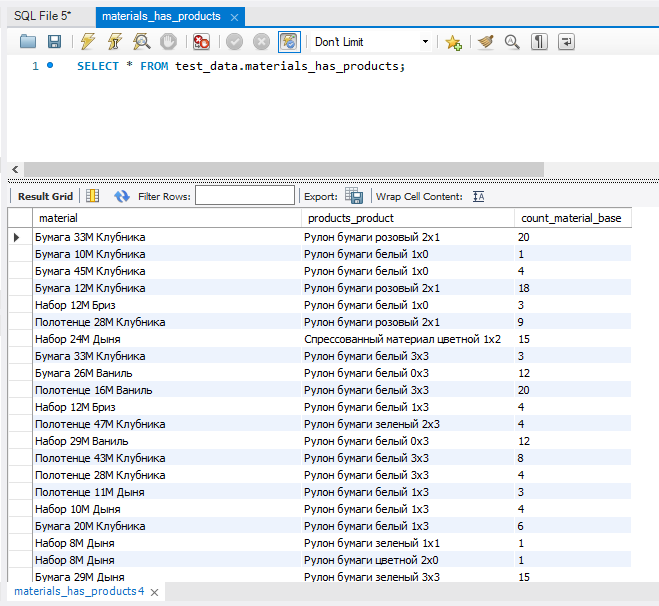


Рисунок 5 – Связующая таблица

В данном случает потребовалось прописать временное разрешение на игнорирование ограничений внешнего ключа

SET FOREIGN\_KEY\_CHECKS=0;

Замечание: скорее всего вам не с первого раза удастся импортировать верно данные в полном объеме и правильном формате. Поэтому не лишней будет команда

Delete from table1

SET SQL\_SAFE\_UPDATES = 0;

Вывод: проделав первую сессию много времени тратиться только на редактирование данных. Можно считать, что первая сессия является очень легкой сессией. Здесь можно набрать максимальное количество баллов. В данном случает таблицы не находятся в 3НФ, поэтому 0.9 балов мы жертвуем на дополнительное время