Базы данных □□

Шаблон для описания структуры данных

https://docs.google.com/document/d/1HFmzIuHap1sfu1RYq-Hu81rl3auIXiYjnnlVV6ebBjE/edit?usp=sharing

Видео-ролиики: теория и практика по базам данных

https://www.youtube.com/embed/YdCGGBUCoDQ

https://www.youtube.com/embed/8cP6x9pTIy0

Типы полей

Типы полей в Laravel Nova https://nova.laravel.com/docs/3.0/resources/fields.html#fields

По сути Nova определяет какие типы полей мы можем использовать, работая с Laravel. Вот их список.

1. **Чекбокс (Boolean)** — булево значение (true/false)
2. **Целое число (Integer)** — натуральное целое число
3. **Дробное числа (Float)** — число с плавающей точкой (не целое)
4. **Дата время (DateTime)** — дата время. Подходит и для времени, и для даты. Учитывает таймзоны и тп.
5. **Выпадающий список (Select)** — выбор из списка захардкоженных значений. Может быть с поиском и множественным выбором. Например, выбор пола: М, Ж, ХЗ
6. **Текст (Text)** — текстовая строка

7. **Текстовое поле (Textarea)** — большое (многострочное) текстовое поле
8. **Визуальный редактор (WYSIWYG, пакет TinyMCE)** — визуальный редактор
9. **Файл** — можно приложить один файл
10. **Массив файлов (пакет Nova Medialibrary)** — можно приложить несколько файлов
11. **Изображение** — одно изображение. Можно делать с пакетом или без. В админке разницы не много
12. **Видео** — одно видео. Можно делать с пакетом или без. В админке разницы не много
13. **Медиа галерея (пакет Nova Medialibrary)** — группа изображений и видео (можно настроить)
14. **Поле-повторитель, конструктор (JSON, пакет Nova Flexible Content)** — очень мощный инструмент для контентных страниц, который мы активно использовали в WordPress. Это поле в котором можно быть массив любых других полей. Самое класнное, что повторитель может содержать сам себя. Но не стоит злоупотреблять этим полем, так как на нем очень трудно сделать какую то логику. То есть оно хорошо подходит, например, для создания лонг-ридов (LP). Но если вы попытаетесь, например, засунуть в повторитель характеристики товара, то не сможете сделать ни филтрацию, ни сортировку по этим характеристикам
15. **Геолокация** — выбор точки на карте с возможностью ввода адреса текстом (уточню, как делается. сделано на мдате)
16. **Ключ-значение (JSON, KeyValue)** — набор пар ключ-значение. Очень удобная фича, добавляющая NoSQL в реляционную СУБД

Типы связей

Типы связей в Laravel https://laravel.com/docs/master/eloquent-relationships

1 к 1

Связь one-to-one — самая редкоиспользуемая. Может пригодиться, например, если Лид превратился в Проект, чтобы связать эти сущности.

1 ко многим

Часто используемая связь. Реулизуется через foreign key. Выглядит таким образом.

Таблица Товары

id	title	category_id
1	Банан	2

id	title	category_id
2	Отбойный молоток	3

Таблица Категории

id	title
1	Канцтовары
2	Фрукты
3	Инструменты

Как видите, товар относится к категории как "одна категория ко многгим товарам"

Многие ко многим

Хороший пример для связи "многие-ко-многим" это записи блога и тэги. Один тэг может быть связан со многими статьями, многие статьи могут быть связаны с одним тэгом. Реализуется через промежуточную таблицу, которую мы называем pivot'ной, а записи в ней pivot'ами.

Таблица Статьи блога (articles)

id	title
1	Статья 1
2	Статья 2

Таблица тэги (tags)

id	title
2	Тэг 1
3	Тэг 2

Промежуточная pivot таблица (article_tag)

article_id	tag_id
2	3
1	3

В нашем примере мы связали обе статьи с тэгом "Тэг 2".

Данные в пивоте

Важно понимать, что в самой связи "Многие ко многим" можно хранить данные. Например, если есть Юзер и Товар, и Юзер может добавить Товар в избранное, при этом добавив комментарий, то реальная таблица связей будет выглядеть так.

product_id	user_id	description
2	3	Дайте две!
1	3	С пивом потянет

Timestamps

По умолчанию, для любой таблицы Laravel генерирует два столбца "created_at" и "updated_at". Программисты не всегда выводят их в адмнку, но зачастую эти данные у нас есть и иногда могут выручить. Например, если нужно отсортировать в админке что нибудь по дате создания или дате последнего редактирования — это очень просто сделать.

Soft deletes

Другая фича, которую часто используют в Laravel. Обратите внимание, что она не идет из коробки, требует небольшого времени на настройку и имеет небольшие ограничения в использовании.

Суть — при удалении данных, они не удаляются, а кладутся в архив. Реализация: в таблице с данными добавляется колонка "deleted_at" и при удалении в нее записывается дата удаления. То есть ко всему прочему мы можем узнать дату удаления сущности. Ну и очевидно, что все это делается, чтобы данные не удалялись и их можно было "спасти".

Как определить тип связи

Для того, чтобы определить, какой тип связи нужно использовать, используйте такой подход. Допустим, у нас есть заказы и пользователи. Задайте вопрос, "может ли быть у юзера несколько заказов?" Затем спросите себя "может ли быть у заказа несколько юзеров?". Ответы: да, нет. Значит это связь "один юзер ко многим заказам". One-to-many.

Каскадные удаления

Если у вас есть связь "один ко многим", то есть ребенок и родитель. Например, если у юзера могут быть заказы, то заказ — это ребенок юзера.

Возникает дилемма, как базе данных поступить с детьми, если вы удаляете их родителя. Звучит грустно, но надо принять решение. Поведение базы при удалении родителя легко изменить для каждой связи в любой момент времени. Однако нужно настроить это поведение до того, как у произойдет удаление.

Варианты поведения базы при удалении родителя

1. `CASCADE` — то самое каскадное удаление. При удалении родителя удалятся и все его дети по этой связи
2. `SET NULL` — связь сбрасывается и в поле **id родителя** записывается NULL. Это возможно только, если ребенок может быть без родителя, иначе база база будет ругаться и ничего не удалит
3. `RESTRICT` – запрет на удаление родителя, если у него есть дети. самый гуманный вариант, но не всегда подходящий
4. `SET DEFAULT` — установит некое дефолтное (по умолчанию) значение ребенку. Например, какого-то дефолтного родителя

Миграции (migrations)

Доставка кода

На каждом нашем проекте есть несколько разных площадок: локальные у разработчиков на компьютерах, тестовая (одна или две) у нас на тестовом сервере и продакшен (боевая площадка, куда ходят реальные посетители). Постоянная задача при этом — доставлять новые фичи сперва на тестовую площадку с локальной, затем на прод. Доставкой кода занимается система контроля версий Git и наш CI/CD инструмент Pullkins. CI/CD — это это автоматизация доставки кода и выполнения рутинных операций над ним, например, установки зависимостей, тестирования или выполнения миграций.

Доставка структуры БД

С кодом все довольно просто, так как есть Git, но что делать с базой данных. Когда прогер изменил структуру какой-то таблицы, как ему перенести это изменение на тестовую площадку? Для этой задачи используются миграции. Это специальные файлы, которые содержат инструкции по изменению структуры БД, например, добавление столбца в таблице. Каждая миграция — отдельный файл, в котором содержится одно или несколько изменений. Таким образом, каждый программист, выполнив все миграции, может полностью воссоздать структуру БД. Надо помнить, что миграции, как правильно, не содержат данных (контента). Обычно при деплое (публикации) на прд площадку мы не меняем данные в БД, только меняем структуру БД. Но при деплое на тестовые площадки, мы обычно стираем все

данные и заполняем базу тестовой информацией. Это происходит автоматически с помощью Pullkins. Тетосвые данные еще называются сидами.

Сиды (seed)

Сиды — это специальный механизм в Laravel, который позволяет заполнить базу тестовыми данными. Это позволяет всегда вести разработку на наполненной базе. Это очень удобно, когда программист разворачивает новый проект. Еще это удобно для тестирования нового функционала, программист сразу делает для вас наполнение. Сиды не существуют без фабрик. Фабрики (factory) — это программная абстракция, которая генерирует тестовые данные. В Фабриках описано, как создавать данные. А в сидах написано сколько и каких сущностей надо создать. Просто знайте эти понятия, когда нибудь пригодится.

Версия #1
Admin создал 4 December 2022 18:37:06
Admin обновил 4 April 2023 16:22:37