

Dancer and DBIx::Class

Stefan Hornburg (Racke)
`racke@linuxia.de`

Dutch Perl Workshop, Utrecht, 25th April 2014

Database Administration

- ▶ phpmyadmin
- ▶ phppgadmin
- ▶ TableEditor

TableEditor Features

- ▶ Different database systems
MySQL, PostgreSQL, ...
- ▶ higher level of abstraction
- ▶ modern frontend
- ▶ concise source code
- ▶ “simple” installation

Input Database Parameters

Database configuration

Database Driver

Pg

Database Name

interchange6

Username

racke

Password

Schema class

interchange6::Schema

Optional. If you don't specify existing DBIx schema class one will be generated for you.

DSN suffix

Optional. Extra options for DB connection.

Submit

View Products

Configuration
Status
Tables
Address
Attribute
Attribute Value
Cart
Country
Group Pricing
Inventory
Media
Media Display
Media Product
Media Type
Merchandising
Attribute
Merchandising
Product
Navigation
Navigation Attribute
Navigation Attribute Value
Order
- - -

Product - List of items

[New Product](#)

Sku	Name	Uri	Priority	Gtin	Canonical sku	Active	Inventory exempt	Created	Last modified	2
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	Filter
F0001	One Dozen Roses	one-dozen-roses	0			1	0	2014-03-26T14:15:14	2014-03-26T14:15:14	✕ 🔗
F0001-PINK	One Dozen Pink Roses	one-dozen-pink-roses	0		F0001	1	0	2014-03-26T14:15:14	2014-03-26T14:15:14	✕ 🔗

15 items found, Page 1 / 8

View Product

Configuration
Status
Tables
Address
Attribute
Attribute Value
Cart
Country
Group Pricing
Inventory
Media
Media Display
Media Product
Media Type
Merchandising Attribute
Merchandising Product
Navigation
Navigation Attribute
Navigation Attribute Value
Order
Orderline
Payment Order

Product

F0002 - Product

General

Orderline

Canonical

Product Attribute

Merchandising Product Related

Merchandising Product

Media Products

Inventory

Group Pricings

Media Display

Variant

Sku

F0002

Name

One Dozen Roses & Calla Lily

Short description

What says I love you better than 1 dozen fresh roses with calla lily?

Description

Surprise the one who makes you smile, or express yourself perfectly with this stunning bouquet of one dozen fresh red roses. This elegant

Price

49.35

Uri

one-dozen-red-roses-calla-lily

Weight



Relationship Orderline

Configuration

Status

Tables

- Address
- Attribute
- Attribute Value
- Cart
- Country
- Group Pricing
- Inventory
- Media
- Media Display
- Media Product
- Media Type
- Merchandising Attribute
- Merchandising Product
- Navigation
- Navigation Attribute
- Navigation Attribute Value
- Order
- Orderline
- Payment Order

Product

F0002 - Product

[General](#) [Group Pricings](#) [Media Display](#) [Variant](#) [Canonical](#) [Product Attribute](#) [Merchandising Product Related](#)

Orderline [Merchandising Product](#) [Inventory](#) [Media Products](#)

Related Orderline

[+ New Orderline](#)

Orderlines id	Order	Order position	Product	Name	Weight	Quantity	Status	10
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>
2	1 - Order	2	F0002 - Product	One Dozen Roses & Calla Lily	5	1	✕ 🔗	<input type="text"/>

1 items found. Page 1 / 1

Overview Dancer::Plugin::DBIC

- ▶ Usage
- ▶ Configuration
- ▶ UTF-8
- ▶ Create schema dynamically

DBIx::Class without Dancer Plugin

```
use Interchange6 :: Schema;  
  
$schema = Interchange6 :: Schema->connect (...);  
  
$schema->resultset('User')->search({..});
```

DBIx::Class with Dancer Plugin

```
use Dancer::Plugin::DBIC;  
  
schema->resultset('User')->search({...});  
  
resultset('User')->search({...});  
  
rset('User')->search({...});
```

Configuration

```
plugins :  
  DBIC :  
    default :  
      dsn: dbi:mysql:interchange6  
      user: racke  
      pass: nevairbe  
      schema_class: Interchange6::Schema
```

Multiple Schemas

plugins :

DBIC:

default :

dsn: dbi:mysql:interchange6

user: racke

pass: nevairbe

schema_class: Interchange6::Schema

legacy :

dsn: dbi:mysql:interchange5

user: racke

pass: nevairbe

schema_class: Interchange5::Schema

Multiple Schemas

```
use Dancer::Plugin::DBIC;  
  
schema('legacy')->resultset('UserDb')->search({..});
```

UTF-8 for MySQL

```
plugins :  
  DBIC :  
    default :  
      dsn: dbi:mysql:interchange6  
      user: racke  
      pass: nevairbe  
      schema_class: Interchange6::Schema  
      options :  
        mysql_enable_utf8: 1
```

Create schema dynamically

- ▶ `schema_class` missing in configuration
- ▶ `DBIx::Class::Schema::Loader`
- ▶ test and development
- ▶ `TableEditor`

Overview Dancer::Session::DBIC

- ▶ engines
- ▶ configuration
- ▶ serialization
- ▶ session expiration

Engines

- ▶ Templates
TT, Xslate, Flute, ...
- ▶ Sessions
Storable, Database, DBIC
- ▶ Logger
File, Syslog
- ▶ Serializer
JSON, YAML, XML

Configuration

`session` name of session engine (DBIC)

`session_options` options

`session_expires` expiration date

Configuration

```
session: "DBIC"  
session_options:  
  dsn: dbi:mysql:interchange6  
  user: racke  
  pass: nevairbe  
  schema_class: Interchange6::Schema  
  resultset: Session  
  id_column: sessions_id  
  data_column: session_data  
session_expires: 12 hours
```

Configuration

```
set session => 'DBIC';  
set session_options => {schema => schema};
```

Example table

```
CREATE TABLE 'sessions' (  
  'sessions_id' varchar(255) NOT NULL,  
  'session_data' text NOT NULL,  
  'created' datetime NOT NULL,  
  'last_modified' datetime NOT NULL,  
  PRIMARY KEY ('sessions_id')  
) ENGINE=InnoDB;
```

Serializer

```
set 'session_options' => {  
  schema      => schema,  
  serializer  => sub { YAML::Dump(@_); },  
  deserializer => sub { YAML::Load(@_); },  
};
```

Session expiration

- ▶ remove old sessions from database
- ▶ `Interchange6::Schema::ResultSet::Session`

```
$schema->resultset('Session')->expire('12 hours');
```

Overview TableEditor

- ▶ Installation
- ▶ Frontend
- ▶ Routes
- ▶ Login
- ▶ Relationships
- ▶ Limitations
- ▶ Configuration

Installation

```
git clone https://github.com/interchange/TableEditor
cd TableEditor
cpanm .
./bin/app.pl
```

Driver

- ▶ DBD::mysql
- ▶ DBD::Pg
- ▶ ...

Frontend

- ▶ Angular
 - ▶ Routes for the frontend
 - ▶ XHR requests to REST API
 - ▶ JSON
- ▶ Bootstrap
- ▶ Theme

Routes

```
get('/:class/:id' => require_login sub {  
  # retrieve database record and add relationships  
  ...  
  
  return to_json($data, {allow_unknown => 1});  
};
```

Login

- ▶ Dancer::Plugin::Auth::Extensible
- ▶ Provider
 - ▶ Unix
 - ▶ DBIC
- ▶ Database (*planned*)

Relationships

- ▶ `might_have`
- ▶ `has_many`
- ▶ `belongs_to`
- ▶ `many_to_many`

Limitations

- ▶ Primary key for **one** column only
- ▶ Speed (complex schemas)
- ▶ Error handling

Configuration

- ▶ Auth::Extensible
- ▶ DBIC
 - ▶ default

Planned Features

- ▶ Search (Solr)
- ▶ Select schema
- ▶ Debian packages

Development

`https://github.com/interchange/TableEditor`

Dancer2

`Plugin::DBIC` <https://metacpan.org/pod/Dancer2::Plugin::DBIC>

`Session::DBIC` not yet ported

`TableEditor` not yet ported