ReStore: relational storage made simple

John Aspinall

Pharo Days 2022

john.aspinall@gmail.com

Bits of History

- 2000 – Development started
  - Originally for Dolphin Smalltalk
  - Initially for own projects
- 2001 – Commercial product
- 2002-2004 – Further small systems
- 2005 – Hiatus

Back to the Future - Part 2

- 2015 – Semi-retired
- 2016 – ESUG Prague
- 2017 – New Project
- 2018 – ESUG Cagliari
- 2019 – ReStore for Pharo

The Day Today

- Shared code base for Pharo and Dolphin
- Regular updates
- > 2000 unit tests
- Full documentation
- Native Connectors - SQLite, MySQL, PostgreSQL
- ODBC Connector – SQLServer, Access…

Design Philosophy

- Smalltalk-first
- Simplicity over flexibility
- Highly transparent
- No SQL
- Database agnostic

Class Definition

```
Object subclass: #Customer
   instanceVariableNames: 'firstName surname emailAddress dateOfBirth address orders'
   classVariableNames: ''
   package: 'SSW ReStore Examples'
```

```
reStoreDefinition

   ^super reStoreDefinition
      define: #surname as: (String maxSize: 100);
      define: #firstName as: (String maxSize: 100);
      define: #emailAddress as: (String maxSize: 100);
      define: #dateOfBirth as: Date;
      define: #address as: Address dependent;
      define: #orders as: (OrderedCollection of: CustomerOrder dependent owner: #customer);
      yourself.
```

Creating the Database

```
ReStore
   connection: (SSWSQLite3Connection on: (Smalltalk imageDirectory / 'test.db') fullName);
   connect;
   addClasses: {Customer. Address. CustomerOrder. CustomerOrderItem. Product};
   synchronizeAllClasses.
```

```
CREATE TABLE `customer` (`id_` INTEGER PRIMARY KEY, `surname` VARCHAR(100), `first_name`
VARCHAR(100), `email_address` VARCHAR(100), `date_of_birth` DATE,

CREATE TABLE `address` (`id_` INTEGER PRIMARY KEY, `line1` VARCHAR(100), `postcode`
VARCHAR(16), `country` VARCHAR(100), `version_` INTEGER);

CREATE TABLE `customer_order` (`id_` INTEGER PRIMARY KEY, `order_date` DATE, `customer`
INTEGER, `total_price` NUMERIC, `version_` INTEGER);

CREATE TABLE `customer_order_item` (`id_` INTEGER PRIMARY KEY, `order` INTEGER,
`product` INTEGER, `quantity` INTEGER, `version_` INTEGER);

CREATE TABLE `product` (`id_` INTEGER PRIMARY KEY, `name` VARCHAR(100), `description`
TEXT, `price` NUMERIC, `version_` INTEGER);
```

Storing Objects

```
Customer new
   firstName: 'John';
   surname: 'Smith';
   address: (Address new country: 'UK'; yourself);
   store.
```

```
INSERT INTO `customer` (`id_`, `surname`, `first_name`, `email_address`,
`date_of_birth`, `address`, `version_`) VALUES (1, 'Smith', 'John', null, null, 1, 1);

INSERT INTO `address` (`id_`, `line1`, `postcode`, `country`, `version_`) VALUES (1,
null, null, 'UK', 1);
```

Reading Objects

```
"All Smiths"
Customer storedInstances select: [ :each | each surname = 'Smith'].
```

```
SELECT * FROM `customer` WHERE `customer`.`surname` = 'Smith';
```

```
"Customers in France"
Customer storedInstances select: [ :each | each address country = 'France'].
```

```
SELECT `customer`.* FROM (`customer` LEFT JOIN `address` ON `address`.`id_` =
`customer`.`address`) WHERE `address`.`country` = 'France';
```

Updating Objects

```
"Updating a Customer"
johnSmith := Customer storedInstances detect: [ :each | each fullName = 'John Smith'].
johnSmith dateOfBirth: (Date newDay: 1 monthIndex: 2 year: 1983).
johnSmith address postcode: 'W1 1AA'.
johnSmith store.
```

```
UPDATE `customer` SET `date_of_birth` = '1983-02-01', `version_` = 2 WHERE
(`customer`.`id_` = 1 AND `customer`.`version_` = 1);

UPDATE `address` SET `postcode` = 'W1 1AA', `version_` = 2 WHERE (`address`.`id_` = 1
AND `address`.`version_` = 1);
```

Try it yourself

rko281/
ReStoreForPharo

Relational database persistence for Pharo objects

https://github.com/rko281/ReStoreForPharo