Rails Relationships\Associations Lab

In this lab we will create a simple rails application with two models:

- Order
- Items

Conditions:

The application will store many Orders in the database.

Each Order may have 1 or many Items.

Step 1: Create a Rails application named AssociationsExample

rails new AssociationsExample

Step 2: Create a scaffold for Order

rails generate scaffold Order order_name:string

Step 3: Create a model for Items

We will setup our Items manually, ie we will not use a scaffold for them!!

Create an Item model (create the model only)

rails generate model Item item_name:string order:references

"order:references" creates a foreign key column (order_id) in "items" referencing "orders"

Step 4: Generate the controller and views for Items

rails generate controller Items index show new edit

This creates the Items controller and the following views (index show new edit), all blank!!

Step 5: Create the database and the Orders and Items tables

rake db:migrate

Step 6: Edit Order.rb

Model: Order.rb

```
class Order < ActiveRecord::Base

 attr_accessible :order_name

end
```

Update the Model as follows:

```
class Order < ActiveRecord::Base

 attr_accessible :order_name

 has_many   :items

end
```

Adding the has_many :items reference to the Order model specifies there is a one to many relationship between them. Ie one Order can have many Items.

Step 6: Update the routes.rb file

Update the resources :orders to look as follows (ie replace resources :orders with the following)

```
resources :orders do

   resources :items

end
```

Step 7: Items Controller

Add the following code to the Items Controller. We will discuss in details the exact operation of these commands in our next lab!!

```
# GET /orders/1/items
  def index
    # For URL like /orders/1/items
    # Get the order with id=1
    @order = Order.find(params[:order_id])

    # Access all items for that order
    @items = @order.items
  end

  # GET /orders/1/items/2
  def show
    @order = Order.find(params[:order_id])

    # For URL like /orders/1/items/2
    # Find an item in orders 1 that has id=2
    @item = @order.items.find(params[:id])
  end

  # GET /orders/1/items/new
  def new
    @order = Order.find(params[:order_id])


    # Associate an item object with order 1
    @item = @order.items.build
  end

  # POST /orders/1/items
  def create
    @order = Order.find(params[:order_id])

    # For URL like /orders/1/items
    # Populate an item associate with order 1 with form data
    # Order will be associated with the item
    @item = @order.items.build(params[:item])
    if @item.save
      # Save the item successfully
      redirect_to order_item_url(@order, @item)
    else
      render :action => "new"
    end
  end

  # GET /orders/1/items/2/edit
  def edit
    @order = Order.find(params[:order_id])

    # For URL like /orders/1/items/2/edit
    # Get item id=2 for order 1
    @item = @order.items.find(params[:id])
  end

  # PUT /orders/1/items/2
  def update
    @order = Order.find(params[:order_id])
    @item = Item.find(params[:id])
    if @item.update_attributes(params[:item])
      # Save the item successfully
      redirect_to order_item_url(@order, @item)
    else
      render :action => "edit"
    end
  end

  # DELETE /orders/1/items/2
  def destroy
    @order = Order.find(params[:order_id])
    @item = Item.find(params[:id])
    @item.destroy

    respond_to do |format|
      format.html { redirect_to order_items_path(@order) }
      format.xml  { head :ok }
    end
  end
```

Be careful with the end tag for the controller class!!

Step 8: Adding the content for the Views

app/views/items/index.html.erb

```
<h1>Items in <%= @order.order_name %></h1>


<table>
  <tr>
    <th>Item name</th>
  </tr>


<% for item in @items %>
  <tr>
    <td><%= item.item_name %></td>
    <td><%= link_to 'Show', order_item_path(@order, item) %></td>
    <td><%= link_to 'Edit', edit_order_item_path(@order, item) %></td>
    <td><%= link_to 'Destroy', order_item_path(@order, item), :confirm =>
'Are you sure?', :method => :delete %></td>
  </tr>
<% end %>
</table>


<br />


<%= link_to 'New item', new_order_item_path(@order) %>
<%= link_to 'Back to Order', @order %>
```

app/views/items/show.html.erb

```
<h1>Item in <%= @order.order_name %></h1>
<p>
  <b>Item name:</b>
  <%= @item.item_name %>
</p>
<%= link_to 'Edit', edit_order_item_path(@order, @item) %> |
<%= link_to 'Back', order_items_path(@order) %>
```

app/views/items/new.html.erb

```
<h1>New item</h1>

<%= render 'form' %>

<%= link_to 'Back', order_items_path(@order) %>
```

app/views/items/edit.html.erb

```
<h1>Editing item</h1>


<%= render 'form' %>


<%= link_to 'Show', order_item_path(@order, @item) %> |
<%= link_to 'Back', order_items_path(@order) %>
```

app/views/orders/show.html.erb

```
<p>
  <b>Order name:</b>
  <%= @order.order_name %>
</p>


<!-- Display items of an order -->
<h2>Items</h2>
<% @order.items.each do |item| %>
  <p>
    <b>Item name:</b>
    <%= item.item_name %>
  </p>
<% end %>



<%= link_to 'Edit', edit_order_path(@order) %> |
<%= link_to 'Back', orders_path %> |
<%= link_to 'Show Items', order_items_path(@order) %>
```

Step 9: Create Form

Create the file named _form.html.erb in the following location

app/views/items/_form.html.erb

```
<%= form_for([@order, @item]) do |f| %>
 <% if @item.errors.any? %>
  <div id="error_explanation">
   <h2><%= pluralize(@item.errors.count, "error") %> prohibited this order from being saved:</h2>


   <ul>
   <% @item.errors.full_messages.each do |msg| %>
    <li><%= msg %></li>
   <% end %>
   </ul>
  </div>
 <% end %>


 <div class="field">
        <p>
  <%= f.label :item_name %><br />
  <%= f.text_field :item_name %>
 </p>
 </div>
 <div class="actions">
  <%= f.submit %>
 </div>
<% end %>
```

Step 10: Run the server

Go to http://127.0.0.1:3000/orders

Create a new order and add a number of items to this order.

Repeat this for a number of Orders.

Use SQLite Manager in Firefox to view the content of both database tables.

Homework:

Can you re-create this lab by using two scaffolds, Order and Item

(hint we did the Item manually in this lab!!)