

Trabalho 2 - Introdução à Banco de Dados

1. Introdução

O trabalho consiste na modelagem de um banco de dados que permite realizar as seguintes funcionalidades:

- Consulta de livros por código, título, preço, gênero, ano de lançamento, participantes.
- Consulta de dados dos participantes como nome, data de nascimento, bibliografia, cidade natal, nacionalidade, etc.
- Consulta de todos os livros em que um participante trabalhou como ilustrador, editor, autor, etc.
- Ranking dos livros mais vendidos.
- Ranking dos autores mais vendidos.
- Lista dos autores que tem maior número de livros publicados.
- Lista dos livros que possuem versão e-book.
- Ranking dos livros de acordo com faturamento.
- Ranking dos autores de acordo com faturamento.
- Informações extras sobre os livros : Trechos, imagens, fotos dos participantes, etc
- Ranking dos livros mais baratos.
- O sistema web para as livrarias.

De acordo com a proposta apresentada as funcionalidades acima seriam implementadas, mas houve uma simplificação uma vez que a parte que envolve pagamentos e vendas depende justamente de alguma ferramenta adicional para implementação completa. Dessa forma foram feitas tabelas simples para representar o setor de vendas enquanto os outros dados foram apresentados em um sistema.

2. Implementação

Para implementação das queries a linguagem SQL foi que é também “gerada” (por código equivalente em ruby) pelo “ruby on rails”. **A geração de código é apenas uma sintaxe diferente e mais legível da forma SQL e não uma automatização.** O aplicativo foi implementado em “ruby on rails”.

O modelo rails de é baseado no Active Record:

```
class Book < ActiveRecord::Base
  validates_uniqueness_of :ISBN

  has_one :additional_info
  belongs_to :genre
  has_many :authorships
  has_many :authors, :through => :authorships, :source => :person
```

Código de declaração da classe Book(início).

O código acima (da definição da classe) é capaz de relacionar a classe book com outras classes (no caso todas as classes apresentadas são tabelas criadas). O Active Record permite que você recupere um único objeto de três formas diferentes. A maneira utilizada para recuperação de dados corresponde ao código abaixo:

```
def self.search book_info
  •
  •
  •
  books = books.where('titulo LIKE ?', "%#{book_info['name']}%") if book_info['name'].present?
  •
  •
  •
end
```

Código do método da search da classe Book.

Como pode ser observado o código acima nada mais é que uma maneira diferente de escrever SQL, uma vez que ela recebe um parâmetro (book_info) que corresponde a pesquisa que deseja ser feita e depois usa a condição de where junto com o comparativo determinado (no caso o Like). Importante saber que como o ruby on rails mapeia as relações de classes na declaração da mesma, ele acaba por entender quais tabelas estão relacionadas à um pesquisa.

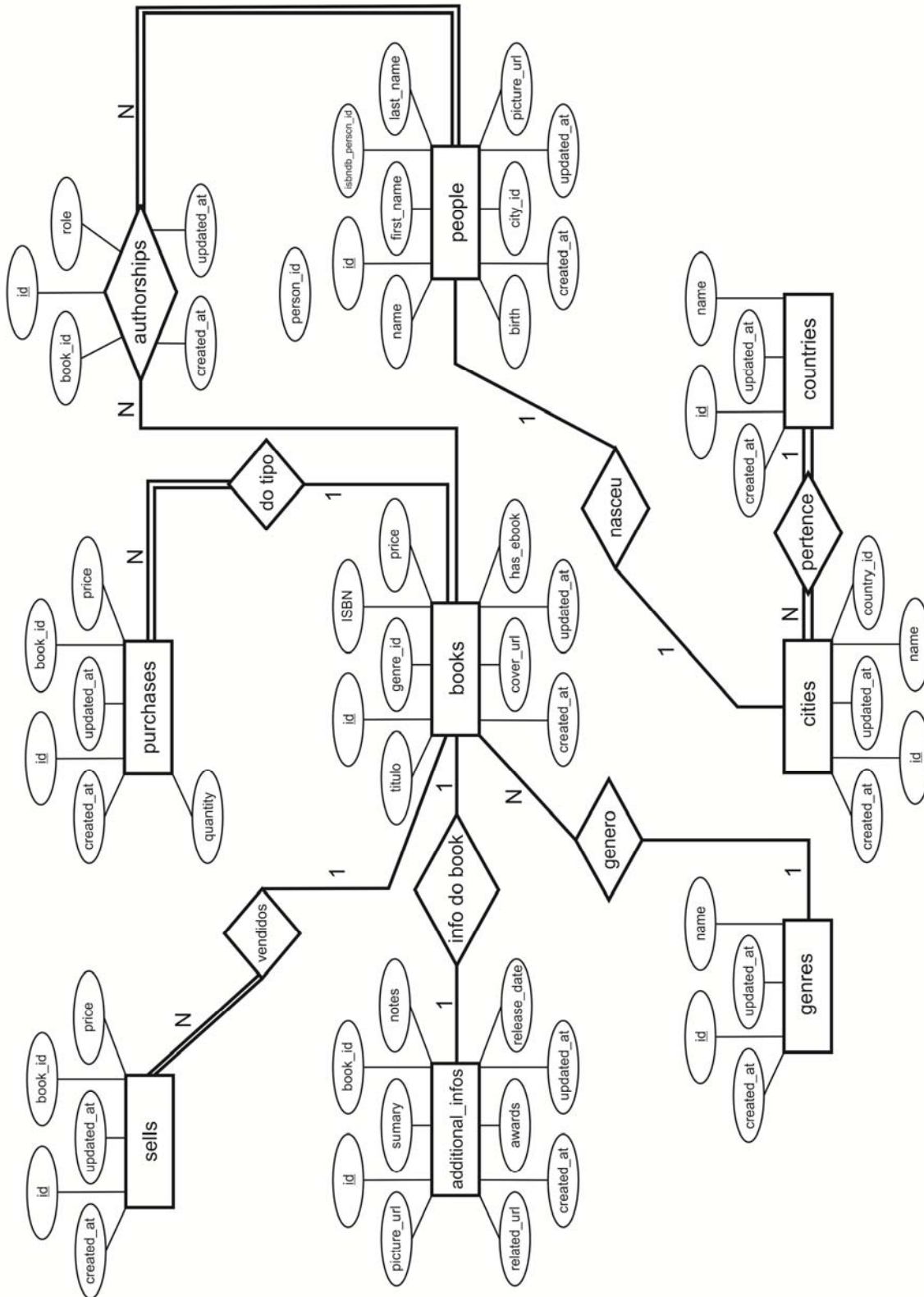
3. Sobre o Rails

Ruby on Rails é um framework livre que aumenta a velocidade e a facilidade no desenvolvimento de sites orientados a banco de dados (database-driven web sites), uma vez que é possível criar aplicações com base em estruturas pré-definidas. Frequentemente referenciado como Rails ou RoR, o Ruby on Rails é um projeto de código aberto escrito na linguagem de programação Ruby. As aplicações criadas utilizando o framework Rails são desenvolvidas com base no padrão de arquitetura MVC (Model-View-Controller).

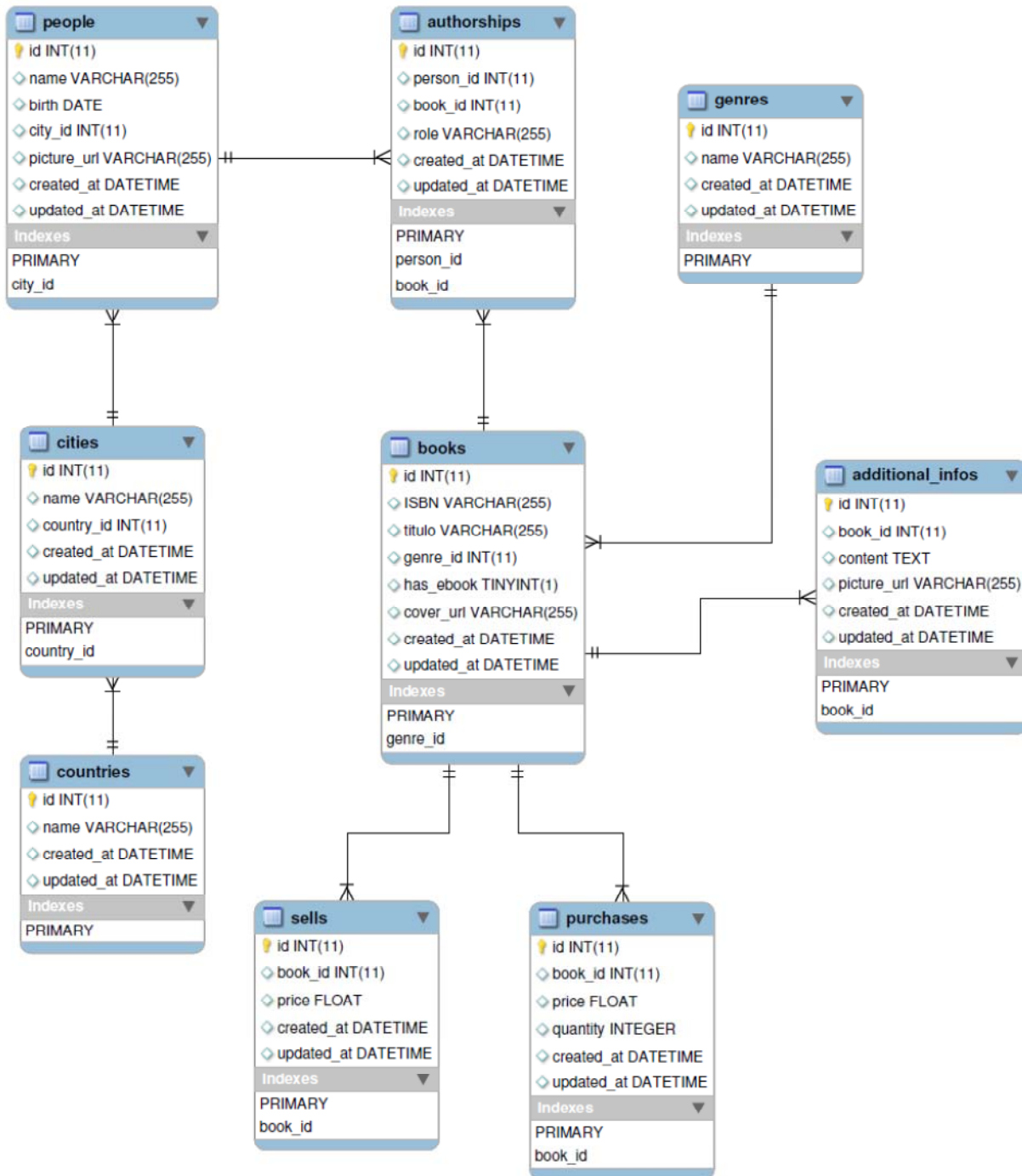
O Rails é um "meta-framework" (ou seja, um framework de frameworks), composto pelos seguintes frameworks:

- Active Record – O Active Record é uma camada de mapeamento objeto-relacional (object-relational mapping layer), responsável pela interoperabilidade entre a aplicação e o banco de dados e pela abstração dos dados.
- Action Pack – Compreende o Action View (geração de visualização de usuário, como HTML, XML, JavaScript, entre outros) e o Action Controller (controle de fluxo de negócio).
- Action Mailer – O Action Mailer é um framework responsável pelo serviço de entrega e até mesmo de recebimento de e-mails. É relativamente pequeno e simples, porém poderoso e capaz de realizar diversas operações apenas com chamadas de entrega de correspondência.
- Active Support – É uma coleção de várias classes úteis e extensões de bibliotecas padrões que foram considerados úteis para aplicações em Ruby on Rails.

4. Modelo ER



5. Modelo Relacional



6. Aplicação

Os modelos acima representam as relações e informações da estrutura implementada de banco de dados. Como já foi observado anteriormente o mapeamento das relações é feito pelo Active Record as queries geradas pelo código ruby são correspondentes aos SQLs abaixo:

Queries:

Query para buscar livro pelo título:

```
SELECT `books`.* FROM `books` WHERE `books`.`titulo` = 'asdf'
```

Query para buscar livro por ISBN:

```
SELECT `books`.* FROM `books` WHERE `books`.`ISBN` = 'asdf'
```

Query para buscar livro por preço:

```
SELECT `books`.* FROM `books` WHERE `books`.`price` = 1.0
```

Query para buscar livro pela data de lançamento

```
SELECT `books`.* FROM `books` INNER JOIN `additional_infos` ON `additional_infos`.`book_id` = `books`.`id` WHERE (additional_infos.release_date = '1900-01-01');
```

Query para buscar livro por gênero:

```
SELECT `books`.* FROM `books` WHERE `books`.`genre_id` = (SELECT `genres`.id FROM `genres` WHERE `genres`.`name` = 'Drama');
```

Query para buscar pessoas que participaram de um livro:

```
SELECT `people`.* FROM `people` INNER JOIN `authorships` ON `people`.`id` = `authorships`.`person_id` WHERE `authorships`.`book_id` = 1;
```

Query para buscar cidade de uma pessoa

```
SELECT `cities`.* FROM `cities` WHERE `cities`.`id` = 1 LIMIT 1
```

Query para buscar país de uma pessoa

```
SELECT `countries`.* FROM `countries` INNER JOIN `cities` ON `countries`.`id` = `cities`.`country_id` WHERE `cities`.`id` = 1 LIMIT 1
```

Query para buscar pessoas a partir do nome

```
SELECT `people`.* FROM `people` WHERE `people`.`name` = 'asdf'
```

Query para buscar pessoas a partir do título de um livro:

```
SELECT `people`.* FROM `people`, `books` INNER JOIN `authorships` ON `people`.`id` =  
`authorships`.`person_id` WHERE `authorships`.`book_id` = (Select id from books where titulo = 'asdf')
```

Query para buscar informações adicionais dos livros (trechos, notas, prêmios...)

```
SELECT `additional_infos`.* FROM `additional_infos` WHERE `additional_infos`.`book_id` = 1 LIMIT 1
```

Query para buscar informações sobre compras (pela livraria) de livros

```
SELECT `purchases`.* FROM `purchases` WHERE `purchases`.`book_id` = 1
```

Query para buscar informações sobre vendas de livros

```
SELECT `sells`.* FROM `sells` WHERE `sells`.`book_id` = 1
```

Query para pegar o ranking dos livros mais vendidos

```
SELECT titulo, count(*) AS total_vendas FROM `books` INNER JOIN `sells` ON `sells`.`book_id` =  
`books`.`id` GROUP BY sells.book_id ORDER BY total_vendas DESC LIMIT 10
```

Query para pegar o ranking dos autores mais vendidos

```
SELECT `people`.name, count(*) AS total_vendas FROM `people` INNER JOIN `authorships` ON  
`people`.`id` = `authorships`.`person_id` JOIN books ON authorships.book_id = books.id JOIN `sells`  
ON `sells`.`book_id` = `books`.`id` GROUP BY sells.book_id ORDER BY total_vendas DESC LIMIT 10;
```

Query para pegar o ranking dos livros de acordo com o faturamento

```
SELECT titulo, SUM(sells.price) AS total_price FROM `books` INNER JOIN `sells` ON `sells`.`book_id` =  
`books`.`id` GROUP BY books.id ORDER BY total_price DESC LIMIT 10;
```

Query para pegar o ranking dos autores de acordo com o faturamento

```
SELECT `people`.name, SUM(sells.price) AS total_price FROM `people` INNER JOIN `authorships` ON  
`people`.`id` = `authorships`.`person_id` JOIN books ON authorships.book_id = books.id JOIN `sells`  
ON `sells`.`book_id` = `books`.`id` GROUP BY people.id ORDER BY total_price DESC LIMIT 10
```

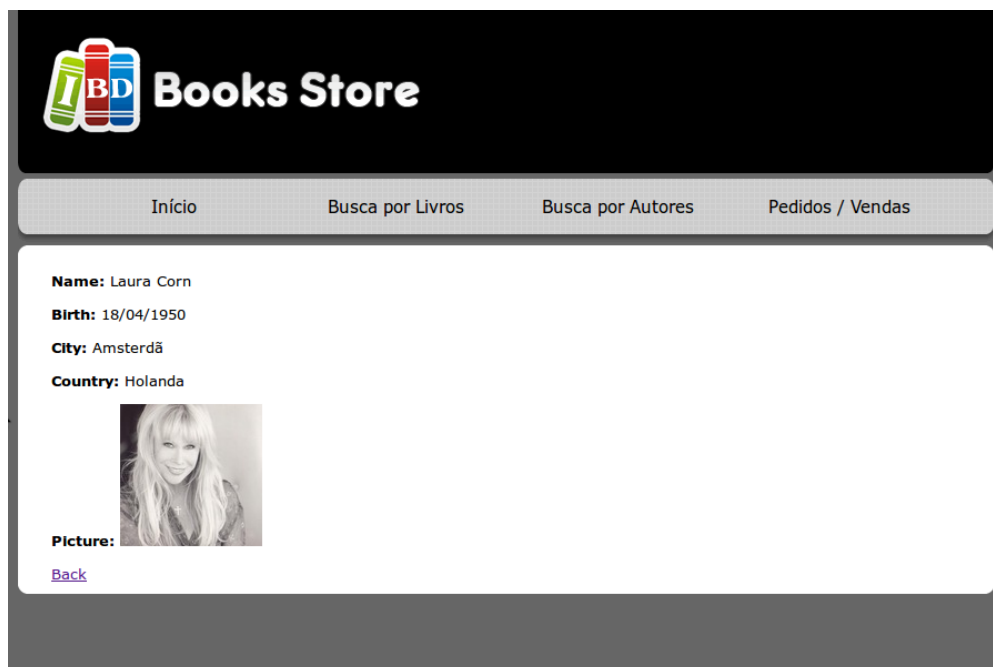
Query para pegar o ranking dos autores com maior número de livros publicados

```
SELECT name, COUNT(*) AS total_livros FROM `people` INNER JOIN `authorships` ON `people`.`id` =  
`authorships`.`person_id` INNER JOIN `books` ON `books`.`id` = `authorships`.`book_id` GROUP BY  
authorships.person_id ORDER BY total_livros DESC LIMIT 10
```

As imagens abaixo ilustram a aplicação implementada:



The screenshot shows the 'Books Store' application interface. At the top, there is a navigation bar with four links: 'Início', 'Busca por Livros', 'Busca por Autores', and 'Pedidos / Vendas'. Below the navigation bar, the main content area is titled 'Pesquisa por Autores'. It contains a form with four input fields: 'Nome', 'Cidade', 'País', and 'Nome de livro que participou'. A 'Pesquisar' button is located at the bottom right of the form.



The screenshot shows the 'Books Store' application interface displaying the profile of an author. The navigation bar is the same as in the previous screenshot. The main content area shows the following information:

- Name:** Laura Corn
- Birth:** 18/04/1950
- City:** Amsterdã
- Country:** Holanda

Below the text information is a small portrait photo of Laura Corn. At the bottom left, there is a 'Picture:' label and a 'Back' link.


Books Store

[Início](#)
[Busca por Livros](#)
[Busca por Autores](#)
[Pedidos / Vendas](#)

Pesquisa por Livros

Título

Isbn

Preço

de até

Ano de lançamento

 ▼

Gênero


 ▼

Participantes

Tem e-book?

 S/N ▼

Pesquisar


Books Store

[Início](#)
[Busca por Livros](#)
[Busca por Autores](#)
[Pedidos / Vendas](#)

Resultados

01 Lesson: Beautiful Women Prefer Nerds! A Real Man's Guide on How to F	Ross Quigley	R\$ 43.00
Laura Corn's 101 nights of grrreat romance	Laura Corn	R\$ 20.00
101 Nights of Grrreat Romance	Laura Corn	R\$ 10.00
101 Ways to Romance	BARBARA DE ANGELIS	R\$ 30.00
120 Great Paintings of Love and Romance CD-ROM and Book	Belanger Grafton, Car	R\$ 870.00
2000s Romance Films (Study Guide)	Não Cadastrado	R\$ 71.00
2012 Harlequin Romance Wall Calendar	Dream Publishing Day	R\$ 5.00
21 Romance Way	Merri Hiatt	R\$ 3.00
35 Frank Sinatra Romance	Various Artists-Soun	R\$ 17.00
365 Glorious Nights of Love and Romance	Patrika Darbo	R\$ 11.00
365 Glorious Nights of Love and Romance	Patrika Darbo	R\$ 817.00
Access Romance Pb	Kaye, Marilyn	R\$ 146.00
Addicted to Filet Crochet-Romance	Ingrid Malik-Connor	R\$ 0.00
Addicted to romance	Hardwick, Joan	R\$ 36.00
ADD & romance	Halverstadt, Jonathan	R\$ 98.00
Adolescent romance novels	Bereska, Tami M.	R\$ 23.00

7. Avaliação dos Integrantes

Pedro Araujo Pires

O Fabiano e o Pedro Duffemond não possuíam muita experiência com SQL, e por isso tiveram um pouco mais de dificuldade no desenvolvimento do TP. Porém, essa dificuldade foi superada, de forma que o desenvolvimento do trabalho foi de grande aprendizado para ele. Muito da linguagem ruby e da framework rails foi aprendido durante esse trabalho. A implementação de interfaces gráficas foi fundamental para entendimento da relação banco de dados e interface.

Fabiano Ishiy Zaparoli

O Pedro Araújo demonstrou bastante facilidade para lidar com SQL, pois já trabalhou nessa área. Participou ativamente do trabalho, inclusive me ajudando a entender alguns conceitos e maneiras mais eficientes de fazer as consultas. Além disso aprendi bastante sobre rails e seus conceitos de implementação de interfaces web.

Pedro Henrique Doffemond Costa

O Pedro Araújo demonstrou bastante facilidade para lidar com SQL, Ruby e Rails. O trabalho não só teve foco no entendimento de SQL como também da framework Rails. Foi interessante aprender um modelo tão rápido e conciso de implementação de sistemas web. Gostaria de poder aprender mais sobre o ruby on rails no futuro.