

# 数据库 Lab 2 实验报告

## 旅行预订系统

PB13011079

杨智

### 题目要求：

使用VC/C++等程序设计语言以及与SQL（ORACLE）的接口编写一个简单的旅行预订系统。该系统涉及的信息有航班，出租车，宾馆房间和客户的数据信息。他们的关系模式如下：

FLIGHTS (String flightNum, int price, int numSeats, int numAvail, String FromCity, String ArivCity);

HOTELS(String location, int price, int numRooms, int numAvail);

CARS(String location, int price, int numCars, int numAvail);

CUSTOMERS(String custName);

RESERVATIONS(String custName, int resvType, String resvKey)

该应用系统完成如下基本功能：

1. 航班，出租车，宾馆房间和客户基础数据的入库，更新（表中的属性也可以根据你的需要添加）。
2. 预定航班，出租车，宾馆房间。
3. 查询航班，出租车，宾馆房间，客户和预订信息。
4. 查询某个客户的旅行线路。
5. 检查预定线路的完整性。
6. 其他任意你愿意加上的功能。

---

---

### 实验环境与工具：

- 操作系统： OS X 10.10.5
- 数据库： Mysql
- 编程语言： Ruby 2.3.1
- Web框架： Ruby on Rails 4.2.6
- 接口： Mysql2

---

---

## 功能介绍：

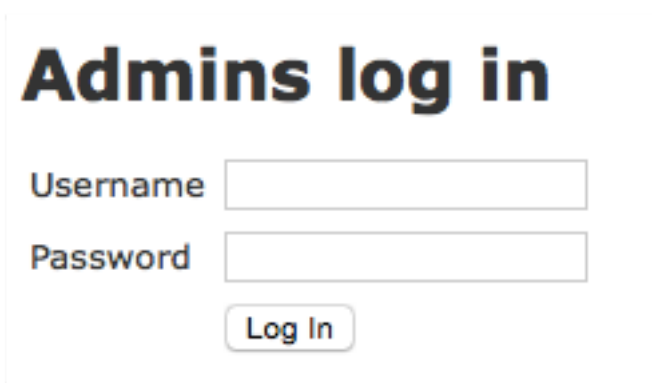
首先，启动服务器：

```
# yangzhi @ yangzhideMacBook-Air in ~/work/database/lab2_travel/trip/trip on git:master x [11:48:42]
$ rake server
rake aborted!
Don't know how to build task 'server' (see --tasks)

(See full trace by running task with --trace)

# yangzhi @ yangzhideMacBook-Air in ~/work/database/lab2_travel/trip/trip on git:master x [11:48:53] C:1
$ rails server
=> Booting WEBrick
=> Rails 4.2.6 application starting in development on http://localhost:3000
=> Run 'rails server -h' for more startup options
=> Ctrl-C to shutdown server
[2016-06-23 11:48:58] INFO  WEBrick 1.3.1
[2016-06-23 11:48:58] INFO  ruby 2.2.2 (2015-04-13) [x86_64-darwin14]
[2016-06-23 11:48:58] INFO  WEBrick::HTTPServer#start: pid=14381 port=3000
```

然后，输入本地网址 <http://localhost:3000/admins>，进入管理员登录界面



The image shows a web form titled "Admins log in". It contains two input fields: "Username" and "Password". Below these fields is a "Log In" button. The form is styled with a light blue border and a white background.

登陆后，进入管理员管理系统

Login successful.

# Administer Management System

[logout](#)

## Administer List

Adminname	Password	Option
root		<a href="#">Destroy</a>

[New Admin](#)

## Cars list

Location	Price	Numcars	Numavails	Option
beijing	10	100	99	<a href="#">Destroy</a> <a href="#">Edit</a>
shanghai	9	200	197	<a href="#">Destroy</a> <a href="#">Edit</a>
guangzhou	8	150	150	<a href="#">Destroy</a> <a href="#">Edit</a>
shenzhen	8	150	150	<a href="#">Destroy</a> <a href="#">Edit</a>

[New car information](#)

## Hotels list

Location	Price	Numcars	Numavails	Option
beijing	1000	50	50	<a href="#">Destroy</a> <a href="#">Edit</a>
shanghai	900	100	99	<a href="#">Destroy</a> <a href="#">Edit</a>

在这个区域删除已有管理员或者创建新的管理员

# Administer List

Adminname	Password	Option
root		<a href="#">Destroy</a>

[New Admin](#)

在这个区域创建新的出租车信息，或者删除、修改已有的出租车信息（flights list、hotels list也是和这个差不多的，不赘述）

## Cars list

Location	Price	Numcars	Numavails	Option
beijing	10	100	99	<a href="#">Destroy</a> <a href="#">Edit</a>
shanghai	9	200	197	<a href="#">Destroy</a> <a href="#">Edit</a>
guangzhou	8	150	150	<a href="#">Destroy</a> <a href="#">Edit</a>
shenzhen	8	150	150	<a href="#">Destroy</a> <a href="#">Edit</a>

[New car infomation](#)

点击Cars list中的edit按钮之后，会跳转到如下的页面，可以对已有的出租车纪录进行修改

# Editing Car

Location

Price



Numcars



Numavail



Update Car

[<< Back to List](#)

在这里可以查看用户的预定信息，可以对用户的订单进行管理

## user1 reservation

### flight reservation

Flightnum	Price	numseats	numavail	Fromcity	Arivcity	Option
CA001	1000	50	48	guangzhou	beijing	<a href="#">unbook</a>
CA003	1000	50	49	shenzhen	guangzhou	<a href="#">unbook</a>
CA001	1000	50	48	guangzhou	beijing	<a href="#">unbook</a>
CA006	1000	50	49	shanxi	shenzhen	<a href="#">unbook</a>

### hotel reservation

Location	Price	Numcars	Numavails	Option
shanghai	900	100	99	<a href="#">unbook</a>
shenzhen	800	70	69	<a href="#">unbook</a>

### car reservation

Location	Price	Numcars	Numavails	Option
shanghai	9	200	197	<a href="#">unbook</a>
beijing	10	100	99	<a href="#">unbook</a>
shanghai	9	200	197	<a href="#">unbook</a>

## user2 reservation

### car reservation

Location	Price	Numcars	Numavails	Option
shanghai	9	200	197	<a href="#">unbook</a>

最后这部分用来对用户进行管理

---

## Users list

user1 [Destroy](#)

user2 [Destroy](#)

点击logout，登出当前的管理员账号

Login successful.

## Administer Management System

[logout](#)

然后，输入本地网址 localhost:3000/users ，进入用户登录界面

# Users log in

Username

Password

Log In

[sign up as a new user](#)

按下“sign up as a new user”之后，可以注册成为新用户

[<< Back to List](#)

## User Signup

Username

Password

Create User

输入用户的账号密码之后，进入用户预订系统



Login successful.

# User Management System

[logout](#)

## check travel route

### Cars list

Location	Price	Numcars	Numavails	Option
beijing	10	100	99	<a href="#">book</a>
shanghai	9	200	197	<a href="#">book</a>
guangzhou	8	150	150	<a href="#">book</a>
shenzhen	8	150	150	<a href="#">book</a>

### Hotels list

Location	Price	Numcars	Numavails	Option
beijing	1000	50	50	<a href="#">book</a>
shanghai	900	100	99	<a href="#">book</a>
guangzhou	800	70	70	<a href="#">book</a>
shenzhen	800	70	69	<a href="#">book</a>

点击“book”按钮，可以预定相应的航班／旅馆／出租车

## Flights list

Flightnum	Price	numseats	numavail	Fromcity	Arivcity	Option
CA001	1000	50	48	guangzhou	beijing	<a href="#">book</a>
CA002	1000	50	50	shanghai	guangzhou	<a href="#">book</a>
CA003	1000	50	49	shenzhen	guangzhou	<a href="#">book</a>
CA004	1000	52	50	beijing	wuhan	<a href="#">book</a>
CA005	1000	50	50	wuhan	beijing	<a href="#">book</a>
CA006	1000	50	49	shanxi	shenzhen	<a href="#">book</a>

这个区域会显示用户所有的订单信息。可以在此处退订

## flight reservation

Flightnum	Price	numseats	numavail	Fromcity	Arivcity	Option
CA001	1000	50	48	guangzhou	beijing	<u>unbook</u>
CA003	1000	50	49	shenzhen	guangzhou	<u>unbook</u>
CA001	1000	50	48	guangzhou	beijing	<u>unbook</u>
CA006	1000	50	49	shanxi	shenzhen	<u>unbook</u>

## hotel reservation

Location	Price	Numcars	Numavails	Option
shanghai	900	100	99	<u>unbook</u>
shenzhen	800	70	69	<u>unbook</u>

## car reservation

Location	Price	Numcars	Numavails	Option
shanghai	9	200	197	<u>unbook</u>
beijing	10	100	99	<u>unbook</u>
shanghai	9	200	197	<u>unbook</u>

在这个地方，“check travel route”，可以检查预订路线的完整性

# User Management System

[logout](#)

---

**check travel route**

---

## Cars list

Location	Price	Numcars	Numavails	Option
----------	-------	---------	-----------	--------

有两种类型的检查，第一种是检查出发地和目的地这两点的预定情况，第二种是检查从出发地到目的地的路线

# User Management System

[logout](#)

---

**check travel route**

---

## Cars list

Location	Price	Numcars	Numavails	Option
----------	-------	---------	-----------	--------

比如，查询从shanxi到beijing的飞行路线是否完整

check two points

Fromcity:

shanxi

Arivcity:

beijing

check path completeness

## your reservation flight reservation

Flightnum	Price	numseats	numavail	Fromcity	Arivcity	Option
CA001	1000	50	48	guangzhou	beijing	<a href="#">unbook</a>
CA003	1000	50	49	shenzhen	guangzhou	<a href="#">unbook</a>
CA001	1000	50	48	guangzhou	beijing	<a href="#">unbook</a>
CA006	1000	50	49	shanxi	shenzhen	<a href="#">unbook</a>

显示结果是这样的，结果为“路线完整”，页面上输出的是调试信息

[back](#) counting: 0 shanxi...shenzhen counting: 1 shenzhen...guangzhou counting: 2 guangzhou...beijing

**validate path completeness:**  
**True**

---

检查路线完整性的方法是用了一个深度限定为10的深度优先搜索

```

2
3 <% count = 0 %>
4 <% queue = [] %>
5 <% queue.push(@startnode) %>
6 <% while(queue.size != 0) %>
7     counting: <%= count%>
8     <% if count == 10 %>
9         <% result = false %>
10        <% break %>
11    <% end %>
12    <% now = queue.shift %>
13    <% if now == @endnode %>
14        <% result = true %>
15        <% break %>
16    <% end %>
17    <% @userfli.each do |chl| %>
18        <% if chl.fromcity == now %>
19            <%= chl.fromcity %>...<%= chl.arivcity %>
20            <% if chl.arivcity == @endnode %>
21                <% result = true %>
22                <% break %>
23            <% end %>
24            <% queue.push(chl.arivcity) %>
25        <% end %>
26    <% end %>
27    <% count = count + 1 %>
28 <% end %>
29
30 <h1> validate path completeness: </h1>
31 <% if result == true %>
32     <h1>True</h1>
33 <% else %>
34     <h1>False</h1>
35 <% end %>

```

## 实现细节

为了实现登录操作的安全性，首先要加密用户保留在数据库中的密码，

明文的密码是“xingzuan2”，加密后的密码如图所示：

```
mysql> select * from users;
+-----+-----+-----+-----+-----+
| id | username | created_at | updated_at | password_digest |
+-----+-----+-----+-----+-----+
| 1 | user1 | 2016-06-08 14:42:00 | 2016-06-08 14:42:00 | $2a$10$V/41eyPoZXI8f0w$XG1R109XWLVdYqpBfAxn4mC17IVCgtEvT4q/q |
| 2 | user2 | 2016-06-08 14:42:00 | 2016-06-08 14:42:00 | $2a$10$yLdfc7D.6EfZLhTwRyDmA0IS53Yf7zqWgbD6HSYgZmV9UtyVWYm. |
+-----+-----+-----+-----+-----+
2 rows in set (0.00 sec)

mysql>
```

另外，在每次进入管理界面之前，都要先确认用户已经登录（管理员的登录验证也是类似的）：

```
users_controller.rb
1 class UsersController < ApplicationController
2   before_action :confirm_logged_in_user, :except => [:login, :attempt_login, :logout, :destroy, :create, :new]
3
4   def login
```

confirm\_logged\_in\_user部分的代码如下：



```

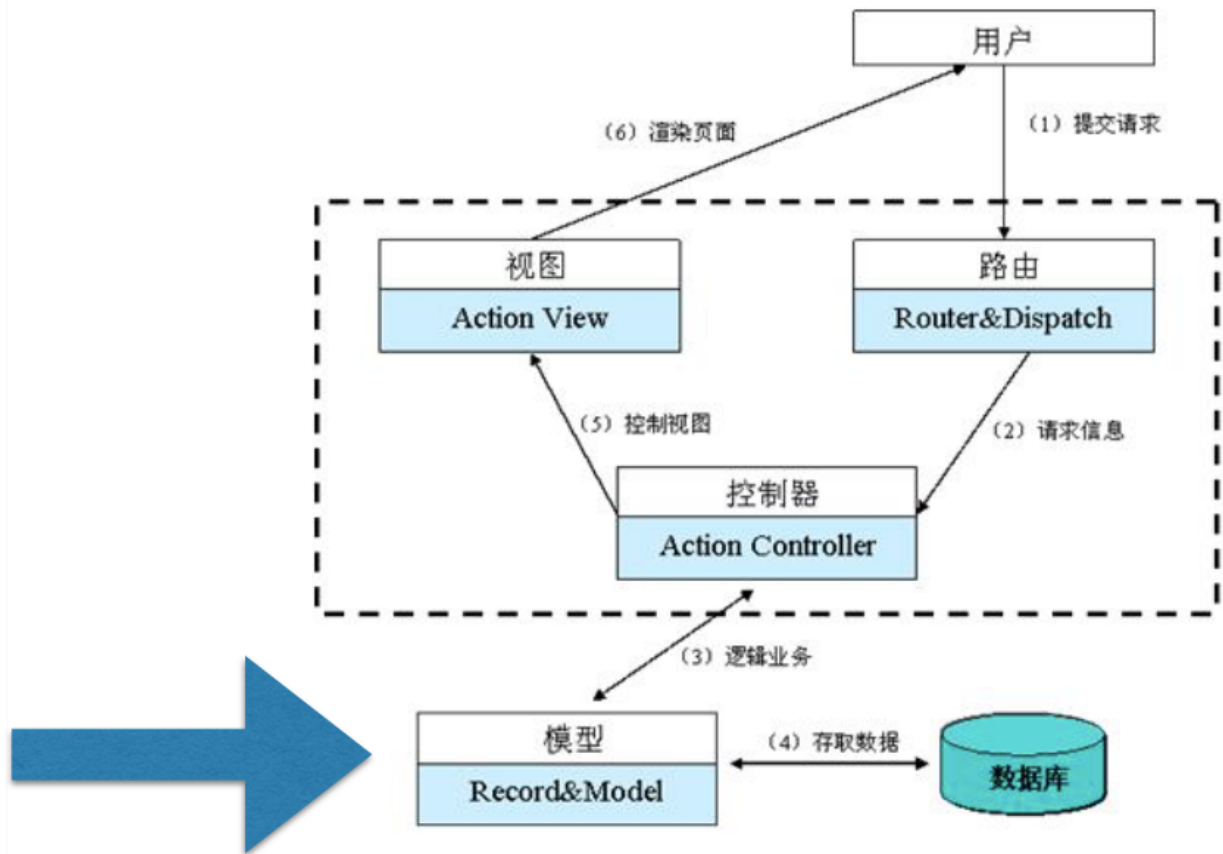
class ApplicationController < ActionController::Base
  # Prevent CSRF attacks by raising an exception.
  # For APIs, you may want to use :null_session instead.
  protect_from_forgery with: :exception
  private
  def confirm_logged_in_admin
    unless session[:user_id] && session[:priv] == 'admin'
      flash[:notice] = "Please log in."
      redirect_to(:controller => 'admins', :action => 'login')
      return false
    else
      return true
    end
  end

  def confirm_logged_in_user
    unless session[:user_id] && session[:priv] == 'user'
      flash[:notice] = "Please log in."
      redirect_to(:controller => 'users', :action => 'login')
      return false
    else
      return true
    end
  end

  def confirm_logged_in
    unless session[:user_id]
      flash[:notice] = "Please log in."
      redirect_to(:controller => 'users', :action => 'login')
      return false
    else
      return true
    end
  end
end

```

另外，Ruby使用的是MVC模型，图示如下：



一般而言，在使用ruby on rails的时候，我们可以在“model”部分做验证（保证数据库的一致性），下面给出一个例子：  
在Flights关系中，我们要验证flightnum不为空，而且必须是unique的，而且numavail（即空座位数）必须大于等于0。

```
class Flight < ActiveRecord::Base
  validates :flightnum, presence: true, uniqueness: true
  validates_numericality_of :numavail, :greater_than_or_equal_to => 0
end
```

其他几个model中的验证也是类似的，这里不赘述了。

---

---

## 实验总结：

做完这个实验之后，感觉自己学到了很多新的知识。

首先，我学会了ruby和ruby on rails。

另外，学会了怎么在一个实际的应用中使用数据库。我感觉理论和实际的应用还是有很多区别的。在rails里面，一般都不会直接在数据库上进行操作，所有的逻辑业务都要通过“model & record”来间接地访问数据库。另外，因为用了mvc的模型，所以我们可以更方便地对数据库进行管理，例如可以应用migrations来新增一个关系、在表中新增一列等的操作。同时，我们也可以在“model”层验证输入到数据库中的数据，保证数据库的一致性。还有很多我没说的好处。