## 1 Rails migrate

### redo migrate

rakedb:migrate:redo VERSION=20131017111433 –trace

$ # use the STEP parameter if you need to go more than one version back

$ rakedb:migrate:redo STEP=3

or

rakedb:rollback

rakedb:migrate:up VERSION=20090408054532

or

delete migrate db

### rails generate migration AddPartNumberToProducts

## 2.rails undo generator :

rails destroy controller lalala

rails destroy model yadayada

rails destroy scaffold hohoho

Rails 3.2 adds a new d shortcut to the command, so now you can write:

rails d controller lalala

rails d model yadayada

rails d scaffold hohoho

## 3 Rails devise

<https://github.com/plataformatec/devise>

Devise 3.0 works with Rails 3.2 onwards. You can add it to your Gemfile with:

gem 'devise'

Run the bundle command to install it.

After you install Devise and add it to your Gemfile, you need to run the generator:

rails generate devise:install

The generator will install an initializer which describes ALL Devise's configuration options and you MUST take a look at it. When you are done, you are ready to add Devise to any of your models using the generator:

rails generate devise MODEL

Replace MODEL by the class name used for the applications users, it's frequently User but could also be Admin. This will create a model (if one does not exist) and configure it with default Devise modules. Next, you'll usually run rake db:migrate as the generator will have created a migration file (if your ORM supports them). This generator also configures your config/routes.rb file to point to the Devise controller.

Note that you should re-start your app here if you've already started it. Otherwise you'll run into strange errors like users being unable to login and the route helpers being undefine

### Controller filters and helpers

Devise will create some helpers to use inside your controllers and views. To set up a controller with user authentication, just add this before\_filter:

before\_filter:authenticate\_user!

To verify if a user is signed in, use the following helper:

user\_signed\_in?

For the current signed-in user, this helper is available:

current\_user

You can access the session for this scope:

user\_session

After signing in a user, confirming the account or updating the password, Devise will look for a scoped root path to redirect. Example: For a :user resource, it will use user\_root\_path if it exists, otherwise default root\_path will be used. This means that you need to set the root inside your routes:

rootto:"home#index"

You can also overwrite after\_sign\_in\_path\_for and after\_sign\_out\_path\_for to customize your redirect hooks.

Finally, you need to set up default url options for the mailer in each environment. Here is the configuration for "config/environments/development.rb":

config.action\_mailer.default\_url\_options={:host=>'localhost:3000'}

Notice that if your devise model is not called "user" but "member", then the helpers you should use are:

before\_filter:authenticate\_member!

member\_signed\_in?

current\_member

member\_session

### Configuring Models

The devise method in your models also accepts some options to configure its modules. For example, you can choose the cost of the encryption algorithm with:

devise:database\_authenticatable,:registerable,:confirmable,:recoverable,:stretches=>20

Besides :stretches, you can define :pepper, :encryptor, :confirm\_within, :remember\_for, :timeout\_in, :unlock\_in and other values. For details, see the initializer file that was created when you invoked the "devise:install" generator described above.

### Started with devise

#### [mika@mppdevdevise\_test]$ rails generate devise:install

create config/initializers/devise.rb

create config/locales/devise.en.yml

==================================================================

Some setup you must do manually if you haven't yet:

1. Ensure you have defined default url options in your environments files. Here

is an example of default\_url\_options appropriate for a development environment

inconfig/environments/development.rb:

config.action\_mailer.default\_url\_options = { :host => 'localhost:3000' }

In production, :host should be set to the actual host of your application.

2. Ensure you have defined root\_url to \*something\* in your config/routes.rb.

For example:

root :to => "home#index"

3. Ensure you have flash messages in app/views/layouts/application.html.erb.

For example:

<p class="notice"><%= notice %></p>

<p class="alert"><%= alert %></p>

4. If you are deploying on Heroku with Rails 3.2 only, you may want to set:

config.assets.initialize\_on\_precompile = false

On config/application.rb forcing your application to not access the DB

or load models when precompiling your assets.

5. You can copy Devise views (for customization) to your app by running:

rails g devise:views

#### [mika@mppdevdevise\_test]$ rails generate devise user

invoke active\_record

createdb/migrate/20131018123208\_devise\_create\_users.rb

create app/models/user.rb

invoketest\_unit

create test/unit/user\_test.rb

create test/fixtures/users.yml

insert app/models/user.rb

route devise\_for :users

## 3 simple form

[hejun@com mysubscription]$ rails generate simple\_form:install --bootstrap

create config/initializers/simple\_form.rb

create config/initializers/simple\_form\_bootstrap.rb

exist config/locales

create config/locales/simple\_form.en.yml

create lib/templates/erb/scaffold/\_form.html.erb

===============================================================================

Be sure to have a copy of the Bootstrap stylesheet available on your

application, you can get it on http://twitter.github.com/bootstrap.

Inside your views, use the 'simple\_form\_for' with one of the Bootstrap form

classes, '.form-horizontal', '.form-inline', '.form-search' or

'.form-vertical', as the following:

= simple\_form\_for(@user, :html => {:class => 'form-horizontal' }) do |form|

## i18n locale

<http://rails-weekly.group.iteye.com/group/wiki/1818-rails-questions-weekly-10-i18n>

<http://guides.rubyonrails.org/i18n.html>

<http://ihower.tw/rails3/i18n.html>

### 使用Yml文件配置，相应的语言：

新建文件，写入如下内容：

zh:

admin:

hello: "test new lang"

login:

hello: "login hello"

Rails Console作相应的测试。

irb(main):007:0> I18n.locale="zh"

=> "zh"

irb(main):013:0> I18n.t 'admin.hello'

=> "test new lang"

## Rails rolify

[mika@mppdev mysubscription]$ rails g rolify:role Role User

create app/models/role.rb

insert app/models/user.rb

create config/initializers/rolify.rb

create db/migrate/20131104131413\_rolify\_create\_roles.rb

===============================================================================

An initializer file has been created here: config/initializers/rolify.rb, you

can change rolify settings to match your needs.

Defaults values are commented out.

A Role class has been been created in app/models (with the name you gave as

argument otherwise the default is role.rb), you can add your own business logic

inside.

Inside your User class (or the name you gave as argument otherwise the default

is user.rb), rolify method has been inserted to provide rolify methods.

Now, if you just have to run the migration using rake command:

rake db:migrate

and you will be able to add the resourcify method inside all models you want

scoped by a role.

===============================================================================

[mika@mppdev mysubscription]

## Rails Environment Variables & Figaro gem

<http://railsapps.github.io/rails-environment-variables.html>

Figaro is for setting rails unix shell environment

## DB Design

### Adding Foreign Key Relationships Using an EER Diagram

The vertical toolbar on the left side of an EER Diagram has six foreign key tools:

* **one-to-one non-identifying relationship**
* **one-to-many non-identifying relationship**
* **one-to-one identifying relationship**
* **one-to-many identifying relationship**
* **many-to-many identifying relationship**
* **Place a Relationship Using Existing Columns**

An identifying relationship is one where the child table cannot be uniquely identified without its parent. Typically this occurs where an intermediary table is created to resolve a many-to-many relationship. In such cases, the primary key is usually a composite key made up of the primary keys from the two original tables. An identifying relationship is indicated by a solid line between the tables and a nonidentifying relationship is indicated by a broken line.

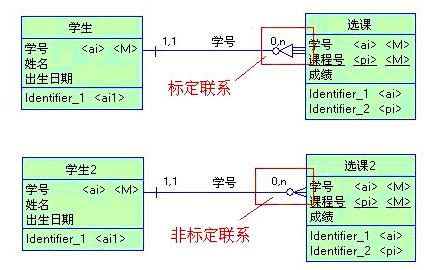
Create or drag and drop the tables that you wish to connect. Ensure that there is a primary key in the table that will be on the “one” side of the relationship. Click on the appropriate tool for the type of relationship you wish to create. If you are creating a one-to-many relationship, first click the table that is on the “many” side of the relationship, then on the table containing the referenced key. This creates a column in the table on the many side of the relationship. The default name of this column is *table\_name\_key\_name* where the table name and the key name both refer to the table containing the referenced key.

When the many-to-many tool is active, double-clicking a table

### 标定实体与非标定实体：

**标定联系：**每个实体类型都有自己的标识符，如果两个实体集之间发生联系，其中一个实体类型的标识符进入另一个实体类型并与该实体类型中的标识符共同组成其标识符时，这种联系则称为标定联系，也叫依赖联系。反之称为非标定联系，也叫非依赖联系。  
 注意：  
在 非标定联系中，一个实体集中的部分实例依赖于另一个实例集中的实例，在这种依赖联系中，每个实体必须至少有一个标识符。而在标定联系中，一个实体集中的全 部实例完全依赖于另个实体集中的实例，在这种依赖联系中一个实体必须至少有一个标识符，而另一个实体却可以没有自己的标识符。没有标识符的实体用它所依赖 的实体的标识符作为自己的标识符。

换句话来理解，在标定联系中，一个实体（选课）依赖 一个实体（学生），那么（学生）实体必须至少有一个标识符，而（选课）实体可以没有自己的标识符，没有标标识符的实体可以用实体（学生）的标识符作为自己的标识符

。

### Foreign Key

<http://stackoverflow.com/questions/8334602/need-to-create-a-foreign-key-when-creating-a-table-on-rails>

Deprecated。

### 数据库索引：

为什么要创建索引呢？这是因为，创建索引可以大大提高系统的性能。   
第一，通过创建唯一性索引，可以保证数据库表中每一行数据的唯一性。   
第二，可以大大加快 数据的检索速度，这也是创建索引的最主要的原因。   
第三，可以加速表和表之间的连接，特别是在实现数据的参考完整性方面特别有意义。   
第四，在使用分组和排序 子句进行数据检索时，同样可以显著减少查询中分组和排序的时间。   
第五，通过使用索引，可以在查询的过程中，使用优化隐藏器，提高系统的性能。

也许会有人要问：增加索引有如此多的优点，为什么不对表中的每一个列创建一个索引呢？这种想法固然有其合理性，然而也有其片面性。虽然，索引有许多优点， 但是，为表中的每一个列都增加索引，是非常不明智的。这是因为，增加索引也有许多不利的一个方面。

第一，创建索引和维护索引要耗费时间，这种时间随着数据 量的增加而增加。   
第二，索引需要占物理空间，除了数据表占数据空间之外，每一个索引还要占一定的物理空间，如果要建立聚簇索引，那么需要的空间就会更大。   
第三，当对表中的数据进行增加、删除和修改的时候，索引也要动态的维护，这样就降低了数据的维护速度。

索引是建立在数据库表中的某些列的上面。因此，在创建索引的时候，应该仔细考虑在哪些列上可以创建索引，在哪些列上不能创建索引。一般来说，应该在这些列 上创建索引，例如：

在经常需要搜索的列上，可以加快搜索的速度；   
在作为主键的列上，强制该列的唯一性和组织表中数据的排列结构；   
在经常用在连接的列上，这 些列主要是一些外键，可以加快连接的速度；   
在经常需要根据范围进行搜索的列上创建索引，因为索引已经排序，其指定的范围是连续的；   
在经常需要排序的列上创 建索引，因为索引已经排序，这样查询可以利用索引的排序，加快排序查询时间；   
在经常使用在WHERE子句中的列上面创建索引，加快条件的判断速度。

同样，对于有些列不应该创建索引。一般来说，不应该创建索引的的这些列具有下列特点：

第一，对于那些在查询中很少使用或者参考的列不应该创建索引。这是因 为，既然这些列很少使用到，因此有索引或者无索引，并不能提高查询速度。相反，由于增加了索引，反而降低了系统的维护速度和增大了空间需求。   
第二，对于那 些只有很少数据值的列也不应该增加索引。这是因为，由于这些列的取值很少，例如人事表的性别列，在查询的结果中，结果集的数据行占了表中数据行的很大比 例，即需要在表中搜索的数据行的比例很大。增加索引，并不能明显加快检索速度。   
第三，对于那些定义为text, image和bit数据类型的列不应该增加索引。这是因为，这些列的数据量要么相当大，要么取值很少。   
第四，当修改性能远远大于检索性能时，不应该创建索 引。这是因为，修改性能和检索性能是互相矛盾的。当增加索引时，会提高检索性能，但是会降低修改性能。当减少索引时，会提高修改性能，降低检索性能。因 此，当修改性能远远大于检索性能时，不应该创建索引。

创建索引的方法和索引的特征   
创建索引的方法   
创建索引有多种方法，这些方法包括直接创建索引的方法和间接创建索引的方法。直接创 建索引，例如使用CREATE INDEX语句或者使用创建索引向导，间接创建索引，例如在表中定义主键约束或者唯一性键约束时，同时也创建了索引。虽然，这两种方法都可以创建索引，但 是，它们创建索引的具体内容是有区别的。   
使用CREATE INDEX语句或者使用创建索引向导来创建索引，这是最基本的索引创建方式，并且这种方法最具有柔性，可以定制创建出符合自己需要的索引。在使用这种方式 创建索引时，可以使用许多选项，例如指定数据页的充满度、进行排序、整理统计信息等，这样可以优化索引。使用这种方法，可以指定索引的类型、唯一性和复合 性，也就是说，既可以创建聚簇索引，也可以创建非聚簇索引，既可以在一个列上创建索引，也可以在两个或者两个以上的列上创建索引。

通过定义主键约束或者唯一性键约束，也可以间接创建索引。主键约束是一种保持数据完整性的逻辑，它限制表中的记录有相同的主键记录。在创建主键约束 时，系 统自动创建了一个唯一性的聚簇索引。虽然，在逻辑上，主键约束是一种重要的结构，但是，在物理结构上，与主键约束相对应的结构是唯一性的聚簇索引。换句话 说，在物理实现上，不存在主键约束，而只存在唯一性的聚簇索引。同样，在创建唯一性键约束时，也同时创建了索引，这种索引则是唯一性的非聚簇索引。因此， 当使用约束创建索引时，索引的类型和特征基本上都已经确定了，由用户定制的余地比较小。

当在表上定义主键或者唯一性键约束时，如果表中已经有了使用CREATE INDEX语句创建的标准索引时，那么主键约束或者唯一性键约束创建的索引覆盖以前创建的标准索引。也就是说，主键约束或者唯一性键约束创建的索引的优先 级高于使用CREATE INDEX语句创建的索引。

索引的特征   
索引有两个特征，即唯一性索引和复合索引。   
唯一性索引保证在索引列中的全部数据是唯一的，不会包含冗余数据。如果表 中已经有一个主键约束或者唯一性键约束，那么当创建表或者修改表时，SQL Server自动创建一个唯一性索引。然而，如果必须保证唯一性，那么应该创建主键约束或者唯一性键约束，而不是创建一个唯一性索引。当创建唯一性索引 时，应该认真考虑这些规则：当在表中创建主键约束或者唯一性键约束时，SQL Server自动创建一个唯一性索引；如果表中已经包含有数据，那么当创建索引时，SQL Server检查表中已有数据的冗余性；每当使用插入语句插入数据或者使用修改语句修改数据时，SQL Server检查数据的冗余性：如果有冗余值，那么SQL Server取消该语句的执行，并且返回一个错误消息；确保表中的每一行数据都有一个唯一值，这样可以确保每一个实体都可以唯一确认；只能在可以保证实体 完整性的列上创建唯一性索引，例如，不能在人事表中的姓名列上创建唯一性索引，因为人们可以有相同的姓名。

复合索引就是一个索引创建在两个列或者多个列上。在搜索时，当两个或者多个列作为一个关键值时，最好在这些列上创建复合索引。当创建复合索引时，应 该考虑 这些规则：最多可以把16个列合并成一个单独的复合索引，构成复合索引的列的总长度不能超过900字节，也就是说复合列的长度不能太长；在复合索引中，所 有的列必须来自同一个表中，不能跨表建立复合列；在复合索引中，列的排列顺序是非常重要的，因此要认真排列列的顺序，原则上，应该首先定义最唯一的列，例 如在（COL1，COL2）上的索引与在（COL2，COL1）上的索引是不相同的，因为两个索引的列的顺序不同；为了使查询优化器使用复合索引，查询语 句中的WHERE子句必须参考复合索引中第一个列；当表中有多个关键列时，复合索引是非常有用的；使用复合索引可以提高查询性能，减少在一个表中所创建的 索引数量。

## Ajax submit form:

|  |  |
| --- | --- |
|  | You want to:   1. Stop the normal behaviour of submit. 2. Send it through ajax to the server. 3. Get a reply back and change things accordingly.   The code below should do that:  $('form').submit(function() {  var valuesToSubmit = $(this).serialize();  $.ajax({  url: $(this).attr('action'), //sumbits it to the given url of the form  data: valuesToSubmit,  dataType: "JSON" // you want a difference between normal and ajax-calls, and json is standard  }).success(function(json){  //act on result.  });  return false; // prevents normal behaviour  }); Rails ajax work way |

<http://keithschacht.com/getting-ajax-to-work-in-rails-3-with-jquery/>

|  |  |
| --- | --- |
|  | try by rendering the partial from the controller as,  def method\_name  render :partial => 'some\_partial'  end  and in js,  success: function(result){  jQuery("#image\_center").html(result);  }  OR ELSE  create a js.erb file corresponding to that action and inside that paste the following:  jQuery("#image\_center").html("<%= escape\_javascript(render(:partial => 'pages/top |

## [SQL JOIN 的用法](http://www.cnblogs.com/yiki/archive/2007/01/04/611452.html)

关于sql语句中的连接（join）关键字，是较为常用而又不太容易理解的关键字，下面这个例子给出了一个简单的解释 --建表table1,table2：  
create table table1(id int,name varchar(10))  
create table table2(id int,score int)  
insert into table1 select 1,'lee'  
insert into table1 select 2,'zhang'  
insert into table1 select 4,'wang'  
insert into table2 select 1,90  
insert into table2 select 2,100  
insert into table2 select 3,70  
如表  
-------------------------------------------------  
 table1  | table2  |  
-------------------------------------------------  
id  name |id  score |  
1  lee |1  90 |  
2  zhang |2  100 |  
4  wang |3  70 |  
-------------------------------------------------

以下均在查询分析器中执行

一、外连接  
1.概念：包括左向外联接、右向外联接或完整外部联接

2.左连接：left join 或 left outer join  
(1)左向外联接的结果集包括 LEFT OUTER 子句中指定的左表的所有行，而不仅仅是联接列所匹配的行。如果左表的某行在右表中没有匹配行，则在相关联的结果集行中右表的所有选择列表列均为空值(null)。  
(2)sql语句  
select \* from table1 left join table2 on table1.id=table2.id  
-------------结果-------------  
id name id score  
------------------------------  
1 lee 1 90  
2 zhang 2 100  
4 wang NULL NULL  
------------------------------  
注释：包含table1的所有子句，根据指定条件返回table2相应的字段，不符合的以null显示

3.右连接：right join 或 right outer join  
(1)右向外联接是左向外联接的反向联接。将返回右表的所有行。如果右表的某行在左表中没有匹配行，则将为左表返回空值。  
(2)sql语句  
select \* from table1 right join table2 on table1.id=table2.id  
-------------结果-------------  
id name id score  
------------------------------  
1 lee 1 90  
2 zhang 2 100  
NULL NULL 3 70  
------------------------------  
注释：包含table2的所有子句，根据指定条件返回table1相应的字段，不符合的以null显示

4.完整外部联接:full join 或 full outer join   
(1)完整外部联接返回左表和右表中的所有行。当某行在另一个表中没有匹配行时，则另一个表的选择列表列包含空值。如果表之间有匹配行，则整个结果集行包含基表的数据值。  
(2)sql语句  
select \* from table1 full join table2 on table1.id=table2.id  
-------------结果-------------  
id name id score  
------------------------------  
1 lee 1 90  
2 zhang 2 100  
4 wang NULL NULL  
NULL NULL 3 70  
------------------------------  
注释：返回左右连接的和（见上左、右连接）

二、内连接  
1.概念：内联接是用比较运算符比较要联接列的值的联接

2.内连接：join 或 inner join

3.sql语句  
select \* from table1 join table2 on table1.id=table2.id  
-------------结果-------------  
id name id score  
------------------------------  
1 lee 1 90  
2 zhang 2 100  
------------------------------  
注释：只返回符合条件的table1和table2的列

4.等价（与下列执行效果相同）  
A:select a.\*,b.\* from table1 a,table2 b where a.id=b.id  
B:select \* from table1 cross join table2 where table1.id=table2.id  (注：cross join后加条件只能用where,不能用on)

三、交叉连接(完全)

1.概念：没有 WHERE 子句的交叉联接将产生联接所涉及的表的笛卡尔积。第一个表的行数乘以第二个表的行数等于笛卡尔积结果集的大小。（table1和table2交叉连接产生3\*3=9条记录）

2.交叉连接：cross join (不带条件where...)

3.sql语句  
select \* from table1 cross join table2  
-------------结果-------------  
id name id score  
------------------------------  
1 lee 1 90  
2 zhang 1 90  
4 wang 1 90  
1 lee 2 100  
2 zhang 2 100  
4 wang 2 100  
1 lee 3 70  
2 zhang 3 70  
4 wang 3 70  
------------------------------  
注释：返回3\*3=9条记录，即笛卡尔积

4.等价（与下列执行效果相同）  
A:select \* from table1,table2

10.git

reset last commit

git reset --soft "HEAD^"

## Image css conflict with bootstrap

当在页面div用百度地图时，出现不显示地图内容的问题。网上就是有牛人。给出了解决办法  
  
http://www.dewen.org/q/5566

bootstrap.css的样式里有放map的样式，用地图的时候使用id="map\_canvas" 就行了

1. #map\_canvas img {
2. max-width: none;
3. }

## mysql export data

### 11.1 export only data

mysqldump -u root -p sdt\_development --no-create-info > mydb2.sql

1qaz0okmZJU

### 12.render javascripts in erb:

<%= render :partial => 'scenic\_spots/add\_makers', :handlers => [:erb], :formats => [:js]%>