

methods

add_column

Creates a new column on the specified table.

```
add_column :table_name, :column_name, :column_type, options ↓
:nullable => true or false - if false, the underlying column has a not null constraint added by the database engine
:limit => size - set a limit on the size of the field
:default => value - set a default value for the column
```

add_index

Creates an index for the specified table.

```
add_index :table_name, :column_name, :unique => true
```

change_column

Change the data type of the specified column

```
change_column :table_name, :column_name, :new_type,
options as add_column ↑
```

create_table

Creates a table on the database. Creates a table called `:table_name` and makes the table object available to a block that can then add columns to it, following the same format as `add_column`.

```
create_table :table_name, options ↓ do |t|
  t.column :column_name, :column_type, :options
end
```

`:force => true` - forces drop of an existing table of the same name before creation the new one

`:temporary => true` - creates a temporary table, one that goes away when the application disconnects from the database

`:id => false` - defines a table with no primary key, for example when you need to define a join table

`:primary_key => :new_primary_key_name` - overrides the default name of `:id` for the primary column, use this to specify the name of the column in the database that Rails will use to store the primary key

`:options => ""` - lets you specify options to your underlying database, e.g. `auto_increment = 10000` will lose default `ENGINE=InnoDB` statement

execute

Takes a single string identifying a valid SQL command to execute directly

```
execute "alter table line items add constraint fk_line_item_products foreign key (product_id) references products(id)"
```

IrreversibleMigration

Use in the `down` method of a migration file to raise an exception when the `up` methods of the same migration file can not be reversed, e.g. changing a column type from `:integer` to `:string`.

```
raise ActiveRecord::IrreversibleMigration
```

rename_table

Renames the specified table.

```
rename_table :new_table_name, :old_table_name
```

rename_column

Renames the specified column.

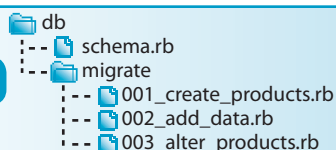
```
rename_table :old_column_name, :new_column_name
```

remove_index

Remove an index for the specified table.

```
remove_index :table_name, :column_name
```

directory



rake tasks

Generate migration

```
ruby script/generate migration
your_chosen_migration_name
```

run all unapplied migrations

```
rake db:migrate
```

migrate database to specific version

```
rake db:migrate VERSION=18
```

use migrations to recreate tables in the testing or production databases

```
rake db:migrate RAILS_ENV=production
```

Create a db/schema.rb file that can be portably used against any database supported by ActiveRecord

```
rake db:schema:dump
```

Load a schema.rb file into the database

```
rake db:schema:load
```

Dump database structure to SQL file

```
rake db:structure:dump
```

Load fixtures from test/fixtures into the current environment's database

```
rake db:fixtures:load
```

Create a sessions table for use with CGI::Sessions::ActiveRecordStore

```
rake db:sessions:create
```

Clear the sessions table

```
rake db:sessions:clear
```

clone your database structure into the test database

```
rake db:test:prepare
```

Empty the test database

```
rake db:test:purge
```

fixtures

Fixtures contain data which can be loaded into your database using migrations. For example, to load data into a table named `customers`

1. Create a directory, `db/migrate/data`

2. Create a file, `customers.yml`, inside `db/migrate/data`

customers.yml

```
melissa:
```

```
  name: Melissa Jayne
  age: 18
```

```
david:
```

```
  name: David Woodford
  age: 23
```

3. Generate a new migration

```
file: ruby script/generate
```

```
migration load_customers_data
```

4. Edit it to load data from the `customers.yml`

file into your `customers` table

xxx_load_customer_data.rb

```
require 'active_record/fixtures'
```

```
class LoadCustomerData
```

```
  def self.up
```

```
    down
```

```
    directory = File.join(File.dirname(__FILE__), "data")
```

```
    Fixtures.create_fixtures(directory, "customers")
```

```
  end
```

```
  def self.down
```

```
    Customer.delete_all
```

```
  end
```

```
end
```

db/migrate/example_001.rb

```
class CreateCustomers < ActiveRecord::Migration
```

```
  def self.up
```

```
    # Create "Customers" table
    create_table :customers, :primary_key => :customer_id, :options =>
      "auto_increment = 10000" do |t|
```

```
      # Add columns to "Customers" table
```

```
      t.column :customer_id, :integer
```

```
      t.column :name, :string, :limit => 30, :null => false
```

```
      t.column :age, :integer
```

```
      t.column :premium, :boolean, :default => 0
```

```
      t.column :photo, :binary, :limit => 2.megabytes
```

```
      t.column :thumbnail, :binary, :limit => 256.kilobytes
```

```
      t.column :dob, :date, :null => false
```

```
      t.column :created_at, :timestamp
```

```
      t.column :notes, :text, :default => "No notes recorded"
```

```
    end
```

```
    # Add "surname" column to "Customers" table
```

```
    add_column :customers, :surname, :string, :limit => 50
```

```
    # Add "price" column to "Orders" table
```

```
    add_column :orders, :price, :decimal, :precision => 8,
```

```
    :scale => 2
```

```
    # Create a record on the "Customers" table
```

```
    Customer.create :name => "David", :surname => "Smith", :age => "32",
```

```
    :premium => "1", :notes => "One of our top customers!"
```

```
  end
```

```
  def self.down
```

```
    # Delete the "Customers" table
```

```
    drop_table :customers
```

```
  end
```

Mapping

	db2	mysql	openbase	Oracle	postgresql	sqlite	sqlserver	Sybase
:binary	blob(32768)	blob	object	blob	bytea	blob	image	image
:boolean	decimal(1)	tinyint(1)	boolean	number(1)	boolean	boolean	bit	bit
:date	date	date	date	date	date	date	datetime	datetime
:datetime	timestamp	datetime	datetime	date	timestamp	datetime	datetime	datetime
:decimal	decimal	decimal	decimal	decimal	decimal	decimal	decimal	decimal
:float	float	float	float	number	float	float	float(8)	float(8)
:integer	int	int(11)	integer	number(38)	integer	integer	int	int
:string	varchar(255)	varchar(255)	char(4096)	varchar2(255)	*	varchar(255)	varchar(255)	varchar(255)
:text	clob(32768)	text	text	clob	text	text	text	text
:time	time	time	time	date	time	datetime	datetime	time
:timestamp	timestamp	datetime	timestamp	date	timestamp	datetime	datetime	timestamp