In this lecture, we will discuss…

- ✧ Including SQL fragments in Active Record queries
- ✧ The dangers of SQL injection

Exact Searches

- We already know about basic retrieves:
  - `find(id)` or `find(id1, id2)`
  - `find_by(hash)`
  - `where(hash)`

These are nice if you know EXACTLY what you are looking for…

```
~/advanced_ar$ rails c
Loading development environment (Rails 4.2.3)
irb(main):001:0> bron = Person.find_by last_name: "James"
  Person Load (0.3ms)  SELECT  "people".* FROM "people" WHERE "people"."last_name" = ? LIMIT 1  [["last_name", "James"]]
=> #<Person id: 7, first_name: "LeBron", age: 30, last_name: "James", created_at: "2015-09-08 21:55:15", updated_at: "20
15-09-08 21:55:15">
```

Including SQL fragments

- Can specify SQL fragment (as opposed to hash) inside the `where` and `find_by`

```
Loading development environment (Rails 4.2.3)
irb(main):001:0> Person.where("age BETWEEN 30 and 33").to_a
  Person Load (1.1ms)  SELECT "people".* FROM "people" WHERE (age BETWEEN 30 and 33)
=> [#<Person id: 1, first_name: "Kalman", age: 33, last_name: "Smith", created_at: "2015-09-08 21:55:15", updated_at: "2015-09
-08 21:55:15">, #<Person id: 7, first_name: "LeBron", age: 30, last_name: "James", created_at: "2015-09-08 21:55:15", updated_
at: "2015-09-08 21:55:15">]
irb(main):002:0> Person.find_by ("first_name LIKE '%man'")
  Person Load (0.2ms)  SELECT  "people".* FROM "people" WHERE (first_name LIKE '%man') LIMIT 1
=> #<Person id: 1, first_name: "Kalman", age: 33, last_name: "Smith", created_at: "2015-09-08 21:55:15", updated_at: "2015-09-
08 21:55:15">
```

- Very powerful, but beware of SQL injection!

What is a SQL injection?

- Manipulating raw SQL to hack into a database
- This includes maliciously dropping/deleting tables or gaining access to confidential information
- https://en.wikipedia.org/wiki/SQL_injection
- Quick demo…

Modify People Table

- Add login and pass fields to people table

```
~/advanced_ar$ rails g migration add_login_pass_to_people login pass
      invoke  active_record
      create    db/migrate/20150908221446_add_login_pass_to_people.rb
~/advanced_ar$ rake db:migrate
== 20150908221446 AddLoginPassToPeople: migrating ============================
-- add_column(:people, :login, :string)
   -> 0.0016s
-- add_column(:people, :pass, :string)
   -> 0.0002s
== 20150908221446 AddLoginPassToPeople: migrated (0.0020s) ===================
```

```
class AddLoginPassToPeople < ActiveRecord::Migration
  def change
    add_column :people, :login, :string
    add_column :people, :pass, :string
  end
end
```

Reload Data

seeds.rb

```ruby
Person.destroy_all

Person.create! [
  { first_name: "Kalman", last_name: "Smith", age: 33, login: "kman", pass: "abc123" },
  { first_name: "John", last_name: "Whatever", age: 27, login: "john1", pass: "123abc" },
  { first_name: "Michael", last_name: "Smith", age: 15, login: "mike", pass: "not_telling" },
  { first_name: "Josh", last_name: "Oreck", age: 57, login: "josh", pass: "password1" },
  { first_name: "John", last_name: "Smith", age: 27, login: "john2", pass: "no_idea" },
  { first_name: "Bill", last_name: "Gates", age: 75, login: "bill", pass: "windows3.1" },
  { first_name: "LeBron", last_name: "James", age: 30, login: "bron", pass: "need more rings" }
]
```

```
~/advanced_ar$ rake db:seed
~/advanced_ar$
```

Verifying The New Data

```
sqlite> select * from people;
id          first_name  age         last_name   created_at                 updated_at                 login       pass
----------  ----------  ----------  ----------  -------------------------  -------------------------  ----------  ----------
8           Kalman      33          Smith       2015-09-08 22:22:51.990586 2015-09-08 22:22:51.990586 kman        abc123
9           John        27          Whatever    2015-09-08 22:22:51.992746 2015-09-08 22:22:51.992746 john1       123abc
10          Michael     15          Smith       2015-09-08 22:22:51.994324 2015-09-08 22:22:51.994324 mike        not_tellin
11          Josh        57          Oreck       2015-09-08 22:22:51.995846 2015-09-08 22:22:51.995846 josh        password1
12          John        27          Smith       2015-09-08 22:22:51.997415 2015-09-08 22:22:51.997415 john2       no_idea
13          Bill        75          Gates       2015-09-08 22:22:51.999069 2015-09-08 22:22:51.999069 bill        windows3.1
14          LeBron      30          James       2015-09-08 22:22:52.000502 2015-09-08 22:22:52.000502 bron        need more
```

SQL Injection Example

- **Our app wants:** Pull out the information for a particular user based on his/her credentials
- **Hacker wants:** ALL users/passwords!

```
~/advanced_ar$ rails c
Loading development environment (Rails 4.2.3)
irb(main):001:0> login = "john2"; pass = "no_idea"
=> "no_idea"
irb(main):002:0> Person.where("login = '#{login}' AND pass = '#{pass}'")
  Person Load (1.1ms)  SELECT "people".* FROM "people" WHERE (login = 'john2' AND pass = 'no_idea')
=> #<ActiveRecord::Relation [#<Person id: 12, first_name: "John", age: 27, last_name: "Smith", created_at: "2015-09-08 22:22:51",
updated_at: "2015-09-08 22:22:51", login: "john2", pass: "no_idea">]>
irb(main):003:0> pass = "got you' OR 'x' = 'x"
=> "got you' OR 'x' = 'x"
irb(main):004:0> Person.where("login = '#{login}' AND pass = '#{pass}'")
  Person Load (0.3ms)  SELECT "people".* FROM "people" WHERE (login = 'john2' AND pass = 'got you' OR 'x' = 'x')
=> #<ActiveRecord::Relation [#<Person id: 8, first_name: "Kalman", age: 33, last_name: "Smith", created_at: "2015-09-08 22:22:51",
 updated_at: "2015-09-08 22:22:51", login: "kman", pass: "abc123">, #<Person id: 9, first_name: "John", age: 27, last_name: "Whate
ver", created_at: "2015-09-08 22:22:51", updated_at: "2015-09-08 22:22:51", login: "john1", pass: "123abc">, #<Person id: 10, firs
t_name: "Michael", age: 15, last_name: "Smith", created_at: "2015-09-08 22:22:51", updated_at: "2015-09-08 22:22:51", login: "mike
", pass: "not_telling">, #<Person id: 11, first_name: "Josh", age: 57, last_name: "Oreck", created_at: "2015-09-08 22:22:51", upda
ted_at: "2015-09-08 22:22:51", login: "josh", pass: "password1">, #<Person id: 12, first_name: "John", age: 27, last_name: "Smith"
, created_at: "2015-09-08 22:22:51", updated_at: "2015-09-08 22:22:51", login: "john2", pass: "no_idea">, #<Person id: 13, first_n
ame: "Bill", age: 75, last_name: "Gates", created_at: "2015-09-08 22:22:51", updated_at: "2015-09-08 22:22:51", login: "bill", pas
s: "windows3.1">, #<Person id: 14, first_name: "LeBron", age: 30, last_name: "James", created_at: "2015-09-08 22:22:52", updated_a
t: "2015-09-08 22:22:52", login: "bron", pass: "need more rings">]>
```

Summary

- Can easily include SQL fragments in queries
- Unfortunately, this approach can leave one susceptible to SQL injection

What's Next?

- Array and Hash parameters to avoid SQL injection