Use find_in_batches or find_each to deal with many records efficiently

Occasionally you need to do something directly on the server – like having all records recalculate something that cannot be done in a migration because it takes a long time.

Let's say you do something like this:

```
Project.all.each(&:recalculate_statistics!)
```

Even though you may have been successful with this on your development machine or the staging server, keep in mind that production machines often hold a lot more records. Using `all` may just work, even with lots of records, but when you iterate over such records and fetch associations for every object, those will be attached to them and kept in memory as well – which causes more and more memory to be consumed over time.

When you need to process many records prefer `find_in_batches` (http://apidock.com/rails/ActiveRecord/Batches/ClassMethods/find_in_batches):

```
Project.find_in_batches do |projects|
  projects.each do |project|
    project.recalculate_statistics!
  end
end
```

If the default of 1000 records per batch is still too much just set your own size like `find_in_batches(:batch_size => 50)`.

When you only want to iterate in batches (and don't need to do anything on the scope `find_in_batches` gives you), you can use `find_each` (http://apidock.com/rails/ActiveRecord/Batches/ClassMethods/find_each) (it does the `each` from above for you) as a shortcut:

```
Project.find_each do |project|
  project.recalculate_statistics!
end
```

Both `find_in_batches` and `find_each` return records ordered by ID because they need to be able to iterate in batches. Modern Rails will raise an error if you try order yourself.

If you are on Rails 2.3, be aware that `find` calls inside the block are implicitly scoped (http://apidock.com/rails/ActiveRecord/Batches/ClassMethods/find_in_batches#771-Careful-with-scopes). This is fixed in Rails 3+.

Posted by Arne Hartherz to makandra dev