**Rails includes and joins:**

Rails includes is eager loading and joins is lazy loading.

And includes uses left outer join and joins uses inner join.

Includes:

With includes, Active records ensures that all of the specified associations are loaded using the minimum possible number of queries.

@companies = Company.includes(:persons).where(:persons => {:active => true}) .all

@companies.each do |company|

Company.person.name

end

When executing includes query it loads the company and persons table in memory which in turn reduce the queries required to retrieve associated data.

While executing the each loop, person name should be obtained by results in a memory no database queries should be fired.

Hence, includes method eagerly loaded the associated person table. So this block requires a single query.

If we don’t want to use person table information, we can use joins.

Company.joins(:persons).where(:persons => {:active => true}).all

Joins is lazy loading, if we want to retrieve any person associated table information it will executes a query.

**Find vs Where:**

Find return an object.

If record not found, it raises Activerecord:Notfound exception and renders 404 error.

Find\_by:

It return an object.

If record not found,its results nil.

If we want to raise Activerecord:Notfound exception, use find\_by!

Where:

This will return active record relation with two or more records if records found.

It will return empty array if no records found.

**Procs vs Lambda:**

In Ruby everything is object except blocks.

In order to convert blocks to an object we used procs and lambdas.