在这里，我们在每个循环的整个持续时间内保留10,000个对象。这会增加总内存消耗和GC时间。如果数据集太大，它还会增加内存不足的风险（请记住，ActiveRecord需要3.5倍的空间来存储数据）。

在没有实例化模型的情况下使用ActiveRecord

如果你只需要运行数据库查询或更新表中的值，请考虑使用以下不实例化模型的ActiveRecord函数。

* ActiveRecord::Base.connection.execute("select\*fromthings")

此函数执行查询并返回其未解析的结果。

* ActiveRecord::Base.connection.select\_values("selectcol5fromthings")

与上一个函数类似，但仅从查询结果的第一列返回值数组。

* Thing.all.pluck(:col1,:col5)

与前两个函数不同。该函数返回一个值数组，其中包含你在pluzer参数中指定的整行或列的值。

* Thing.where("id<10").update\_all(col1:'something')

更新表中的列值。

这些操作不仅可以节省内存，而且使运行速度更快，因为它们既不实例化模型，也不在过滤器之前/之后执行。他们所做的只是运行纯SQL查询，在某些情况下，返回数组。

使ActionView更快

模板渲染花费的时间比控制器代码长，这并不罕见。但是你可能认为你不能做太多事情来使它加速。大多数模板只是对渲染辅助函数的调用的集合，这些函数是你没有编写的，并且无法真正被优化 - 除非它们在循环中被调用。

渲染基本上是字符串操作。我们已经知道，这需要占用CPU时间和内存。在循环中的速度需要对每次已经很慢的迭代进行乘方。因此，每次迭代模板中的大型数据集时，请查看是否可以对其进行优化。