我们在每一个循环的整个过程中保留10,000个对象。这增加了总内存的消耗和GC的时间。它也增加了如果数据集太大，内存会耗尽的风险（记住，活动记录需要3.5倍的空间来储存数据）。

**在不示例模型的情况下，用查询**

如果你所需要的只是运行数据库查询或更新表中的列，考虑使用以下不实例化的查询函数模型模型。

* 查询：:Base.connection.execute("select \* from things")

这个函数执行查询并返回未解析的结果。

* 查询：:Base.connection.select\_values("select col5 from things")

类似于前一个函数，但是它的返回值只返回数组查询结果的第一列

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

前两个函数的变化，返回一个包含要提取的参数中指定的整行货列的数组的值

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

更新表中的列。

它们不仅可节省内存，而且运行速度更快，因为它们都不能实例化模型或者再过滤器之前/之后执行。他们所有能做的是简单地跑SQL查询，在某些情况下，返回的数组作为结果。。

**让 ActionView 更快**

模板呈现花费的时间比控制器代码长并不罕见。但是你可能认为你不能做什么来加快它。大多数模板只是一个调用集合，用来呈现您没有编写的helper函数。而且它不能真正优化，除非在循环中调用它们。

呈现基本上是一个字符串的操作。我们已经知道了，这需要CPU时间和内存。在一个循环中，我们把已经存在的很慢的效果相乘。因此，每次在模板中迭代大型数据集时，请查看是否你可以优化它。