

**17 SEPTEMBER 2019** 

# Rails find\_in\_batches vs find\_each

Often, we come across querying in batches to fetch records through ActiveRecord in Rails. This article discusses how we can use find\_in\_batches and find\_each to query records in batches with ActiveRecord.

Fetching all records with, causes performance issue. It loads all records in memory at once. To resolve this, use <u>find\_each instead</u> of all.each.

Let's consider User schema as given below.



### 1. find\_in\_batches

To query records in batches, we can use <u>find\_in\_batches</u>. It returns a group of records being queried. Then, we can iterate over the group to get individual record to process.

```
User.where('age > 19').find_in_batches do |users|
  users.each do |user|
   puts "User: #{user.name}, Age: #{user.age}"
  end
end
```

This fetches users in batches from database and gives us a group of users in variable users. We have iterated over users variable, to print user name and age in the loop.

#### Supported options

find\_in\_batches support multiple options to control number of records queried and fetched in a batch. It also supports, start and finish options to indicate where to start from and where to end quering.

Below are the supported options for find\_in\_batches:

batch\_size - Number of records to be queried and fetched

- start specifies the minimum value of primary key to start querying from. inclusive of value provided.
- finish specifies the maximum value of primary key to start querying from. inclusive of value provided.

These options are useful, if we want to query records and the order in which we query and process them is not a concern. In such case, we can parallelize to improve performance of such operation by using start and finish option.

## 2. find\_each

find\_each is an extension to ActiveRecord based on the way people use find\_in\_batches.

find\_each calls find\_in\_batches internally.

Usually, we need to

- query in batches
- process individual records from batch

As discussed earlier, find\_in\_batches does the exact same thing.

We can use find\_each without worrying about the batch queries. It takes care of it internally. The code described above can be written as given below with find\_each.

```
User.where('age > 19').find_each do |user|
  puts "User: #{user.name}, Age: #{user.age}"
end
```



```
def find_each(start: nil, finish: nil, batch_size: 1000, err
  if block_given?
    find_in_batches(start: start, finish: finish, batch_size
       records.each { |record| yield record }
    end
  else
    enum_for(:find_each, start: start, finish: finish, batch
       relation = self
       apply_limits(relation, start, finish).size
    end
  end
end
```

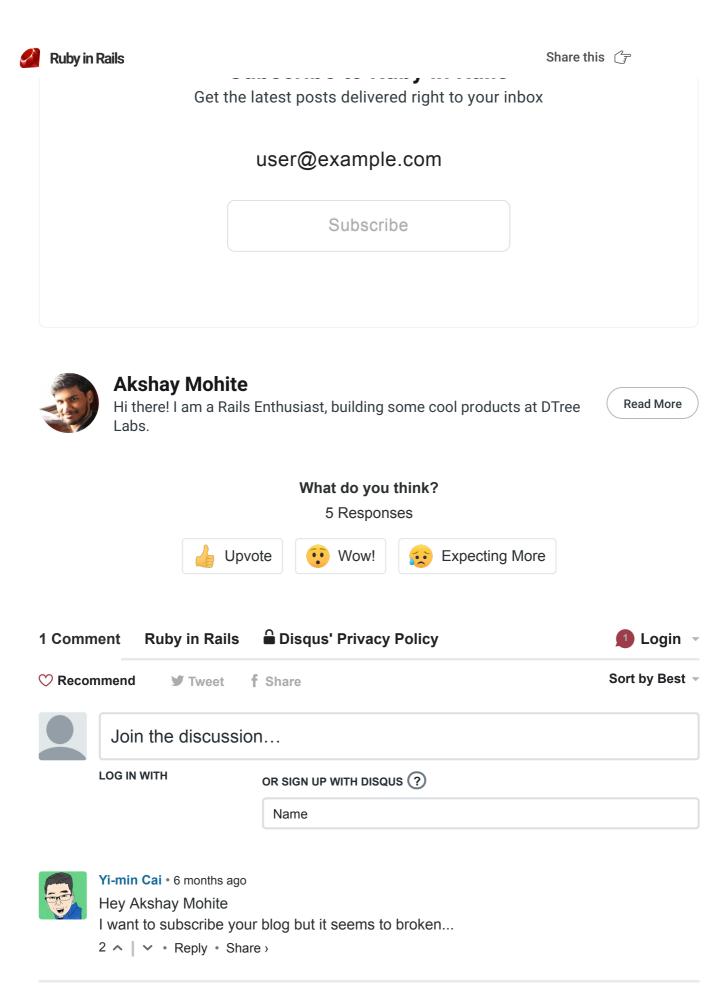
#### **Supported options**

Options supported by find\_each are exactly same as options supported by find\_in\_batches.

Below are the supported options for find\_each:

- batch\_size Number of records to be queried and fetched in the batch. Default value is 1000
- start specifies the minimum value of primary key to start querying from. inclusive of value provided.
- finish specifies the maximum value of primary key to start querying from. inclusive of value provided.

Thus, unless and until you need explicity control over start and finish option on the batches or you need batches based on custom start and finish options, you can use find\_each to avoid an overhead of managing iteration of batches.





# Rails Routes expanded view option



AKSHAY MOHITE

1 MIN READ

## Override Devise Mailer in Rails



AKSHAY MOHITE

1 MIN READ

Ruby in Rails © 2020

Latest Posts Github Twitter LinkedIn Instagram