A Quick Note On Performance

Now let's play a little with numbers. First, let me make some assumptions:

- Our application will have around 100 000 users
- Users have an average list size (subscriber count) of 5 000
- They send an average of 1 e-mail per week

100 000 users * 5 000 e-mails * 52 weeks = 26 000 000 000 rows in `sent_mails` in one year. It seems incredibly high, right? Let's make some other assumptions:

- Acquiring 100 000 users will probably take YEARS. It depends on many factors, of course, but it's a very high number. ConvertKit has about 500 000 users (the number comes from their site), and it's an ~8-year-old business in a niche where the CEO is very well-known.
- 5 000 is an entirely made-up number; I couldn't find a reliable number online.

At this stage, I wouldn't worry about this kind of problem. Instead, I want to be ready for the first 100 or 1000 users. What can we do to make the DB effective?

- Indexing the `sent_at` column.
- Also the `opened_at` and `clicked_at` columns

The indexing itself can speed up your application. The next step is to write queries that can use these indexes, so 'never' write something like this:

```
select count(*)
from sent_mails
where year(sent_at) = "2022";
```

This query **won't** use your index on the `sent_at` column. Simply because MySQL cannot use indexes if you're using the `year` function. So instead of using a function, write queries like that: