Filtering in MongoDB

When performing CRUD operations, we can pass in **filters** to the operation's parameters. We'll learn a bit about the simple ones today.

Literal Value Queries

Literal value queries allow you to query for data that exactly matches a value you provide in the query document. A literal value query has two parts: a field name and a value.

```
const userQuery = {"username": "arvind6104"};
```

In this case, the field name is `"username"` and the value is `"arvind6104"`.

Documents returned from such a query must contain a field that has exactly the same name as the provided name and a value for that field that is exactly the same as the provided value. We will show how we can take advantage of this property in the following examples.

Example 1

Let's say we have some postQuery where we have the `"author"` of a post.

```
const postQuery = { "author": "arvind6104" };
```

Note: we can equivalently represent our postQuery in the following way (we'll learn more about comparison operators in recitation):

```
const postQuery = { author: {$eq: "arvind6104"}};
```

We can create a function within our `concepts/post.ts` to get the information for a post by its particular author.

```
async getByAuthor(postQuery: Filter<PostDoc>){
      const post = await this.posts.readOne({postQuery});
      return post;
}
```

If we call our function with `postQuery` defined as above, we could return the following result (given the database is already populated).

```
{ "_id": 15, "author": "arvind6104", "content": "software design is
cool!"}
```