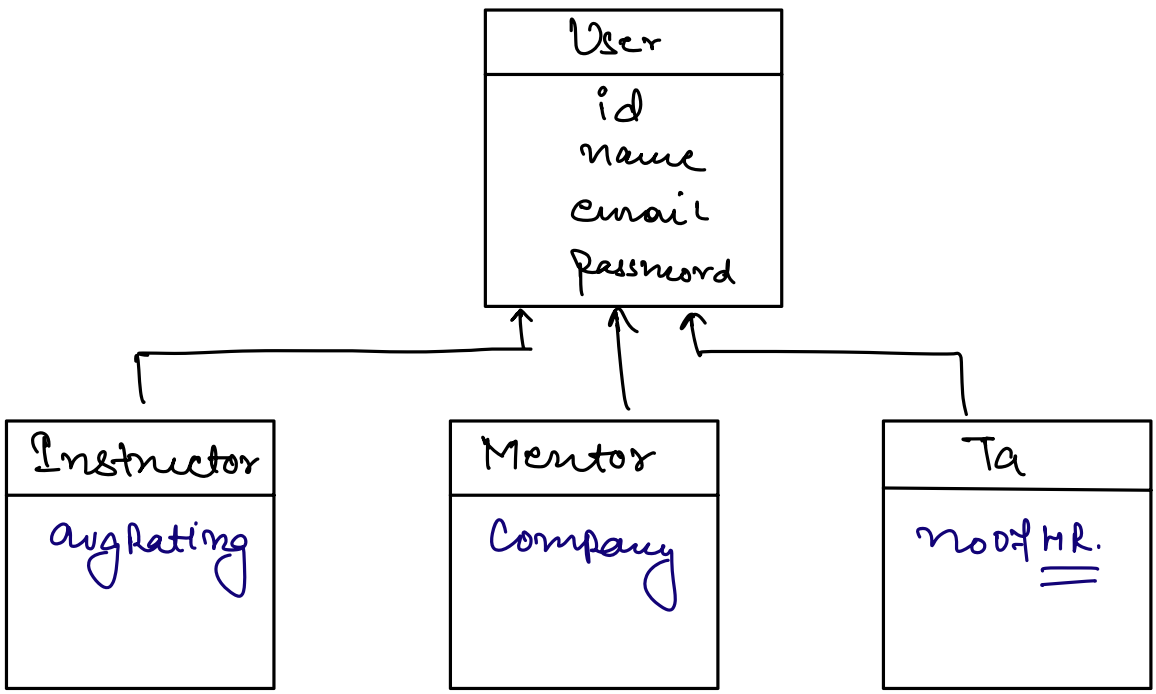
Agenda.

→ Representing Inheritance in DB.

→ Implementing APIs using DB.

→ JPA Queries.

Representing Inheritance in DB.



1) Mapped Super Class.

instructors

id	name	email	password	avgRating

mentors

id	name	email	password	Company

tas

id	name	email	password	noOfHR

2) Joined Table.

Users

id	name	email	password

instructors

user_id	avgRating

mentors

user_id	company

tas

user_id	noOfHR

3) Table Per Class.

Users

id	name	email	password

instructors

id	name	email	password	avgRating

mentors

id	name	email	password	Company

tas

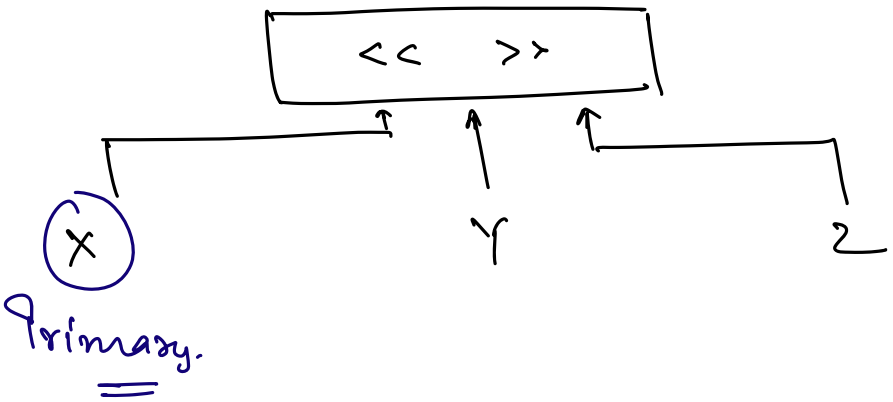
id	name	email	password	noOfHR

4) Single Table. ✗

id	name	email	password	avgRating	Company	. . . .	type

⇒ When there are multiple beans to inject, which one to choose ?

- 1) @Primary
- 2) @Qualifier.

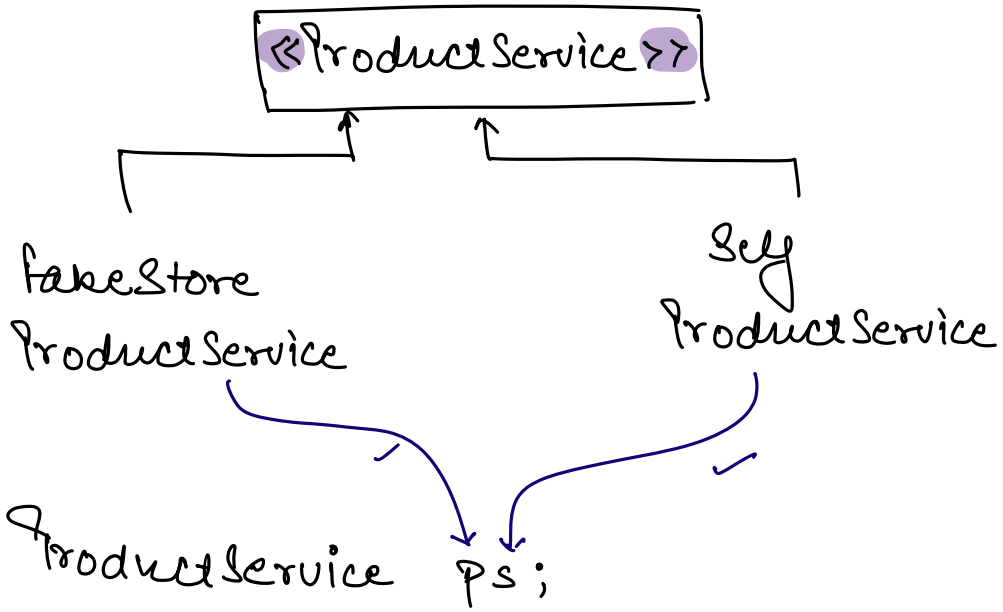

<< >>
X
Y
Z
Primary.

A {

}

B {

}


«ProductService»
FakeStore ProductService
Self ProductService
ProductService ps;

Queries.

⇒ Hibernate (ORM) will write the queries on our behalf based on the function name.

```
findByEmail(——);

Select * from user
where email = ?
```

Declared Queries.

⇒ No need to write queries on our own.

⇒ Just give a method name & ORM will create a query based on the method name.