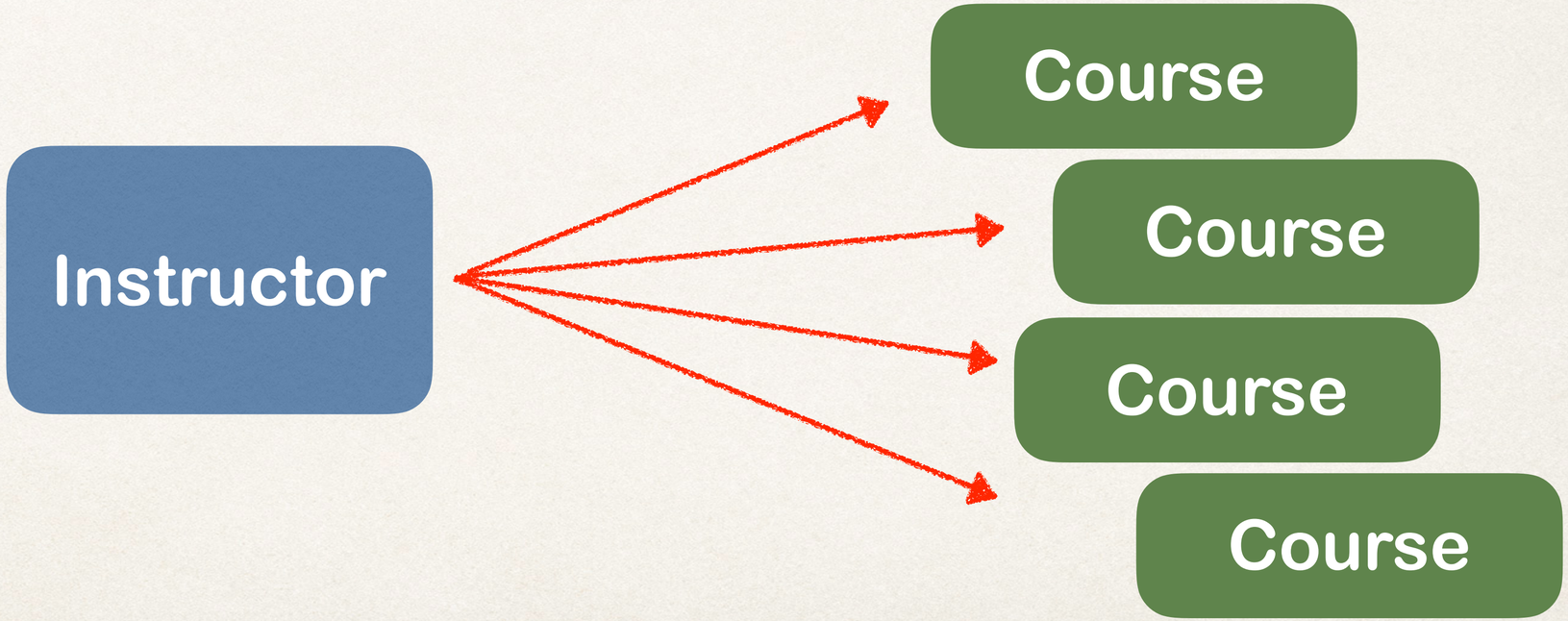
luv2code

Fetch Types: Eager vs Lazy

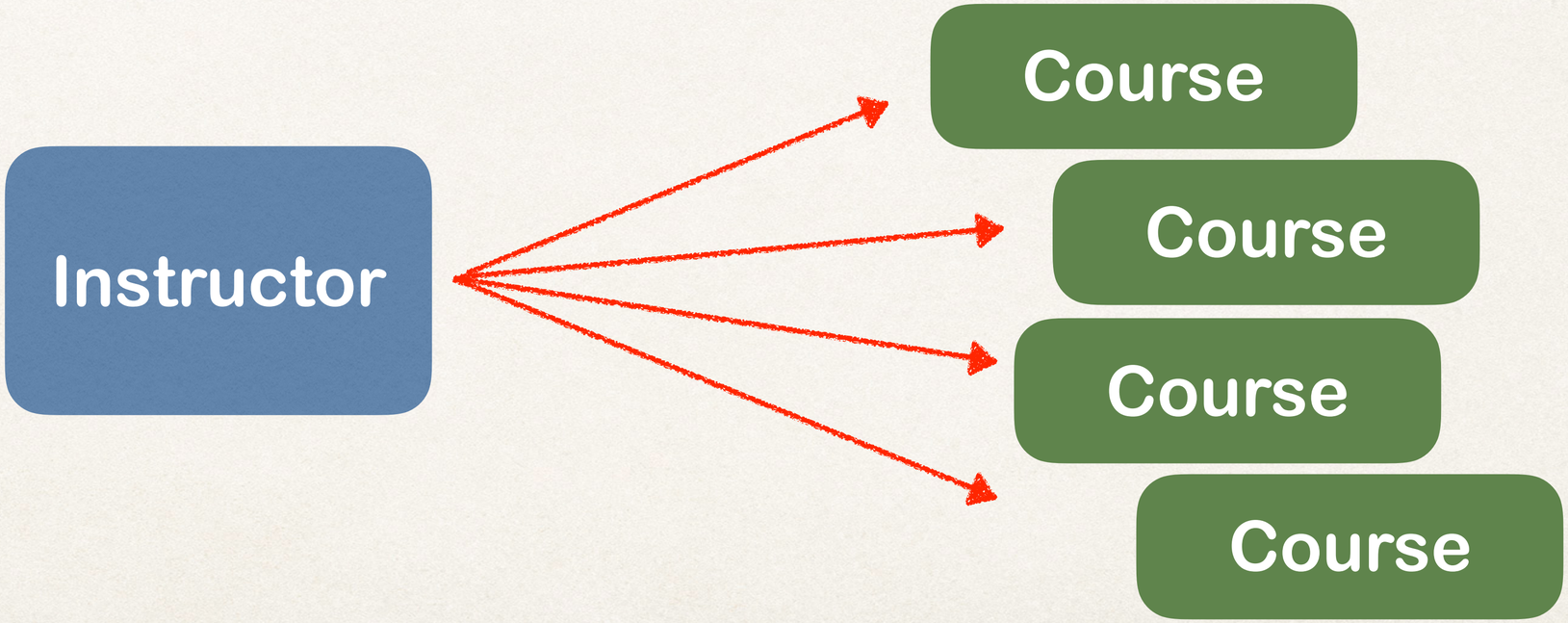
Fetch Types: Eager vs Lazy Loading

- When we fetch / retrieve data, should we retrieve EVERYTHING?
  - **Eager** will retrieve everything
  - **Lazy** will retrieve on request



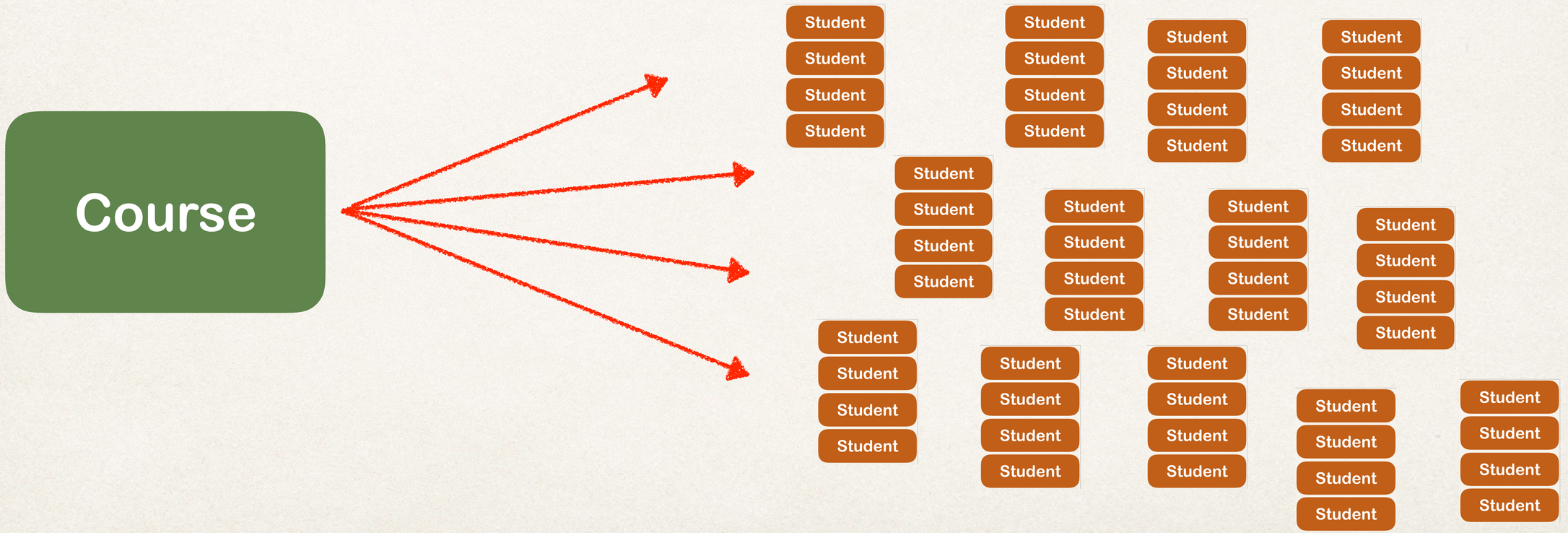
Eager Loading

- Eager loading will load all dependent entities
  - Load instructor and all of their courses at once



Eager Loading

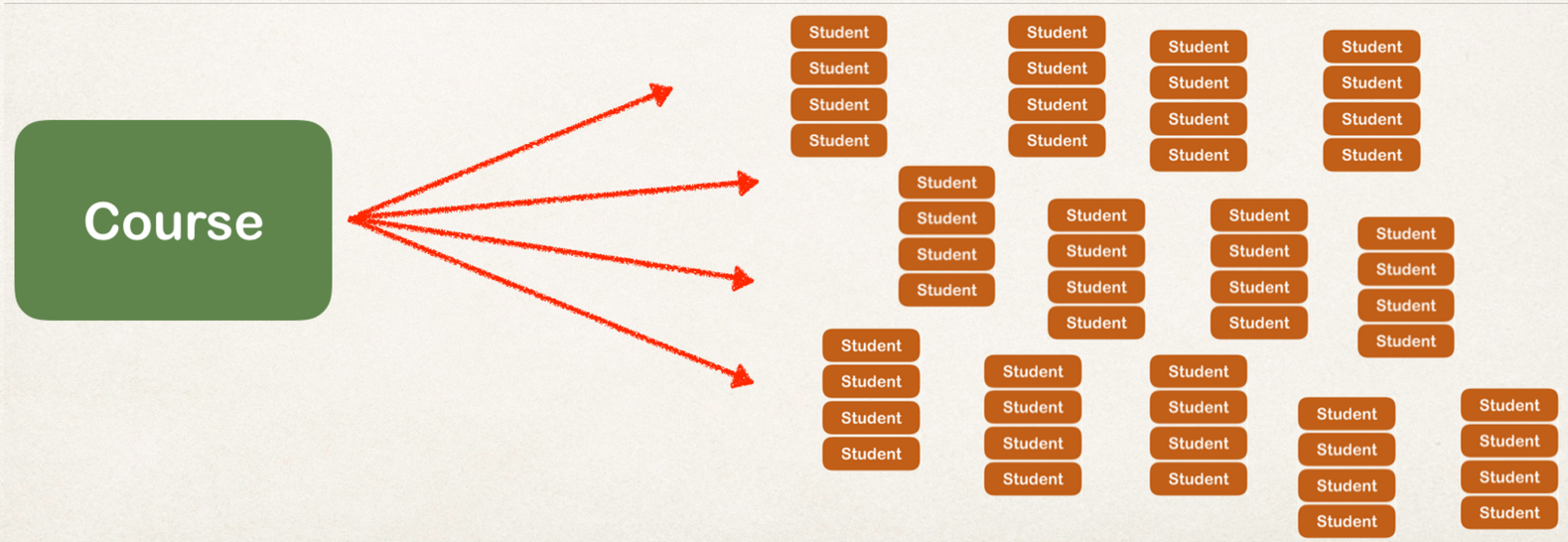
- What about course and students?
  - Could easily turn into a performance nightmare …



Eager Loading

- In our app, if we are searching for a course by keyword
  - Only want a list of matching courses

- Eager loading would still **load all students** for **each course** …. not good!



Best Practice

Only load data when absolutely needed

Prefer Lazy loading
instead of
Eager loading

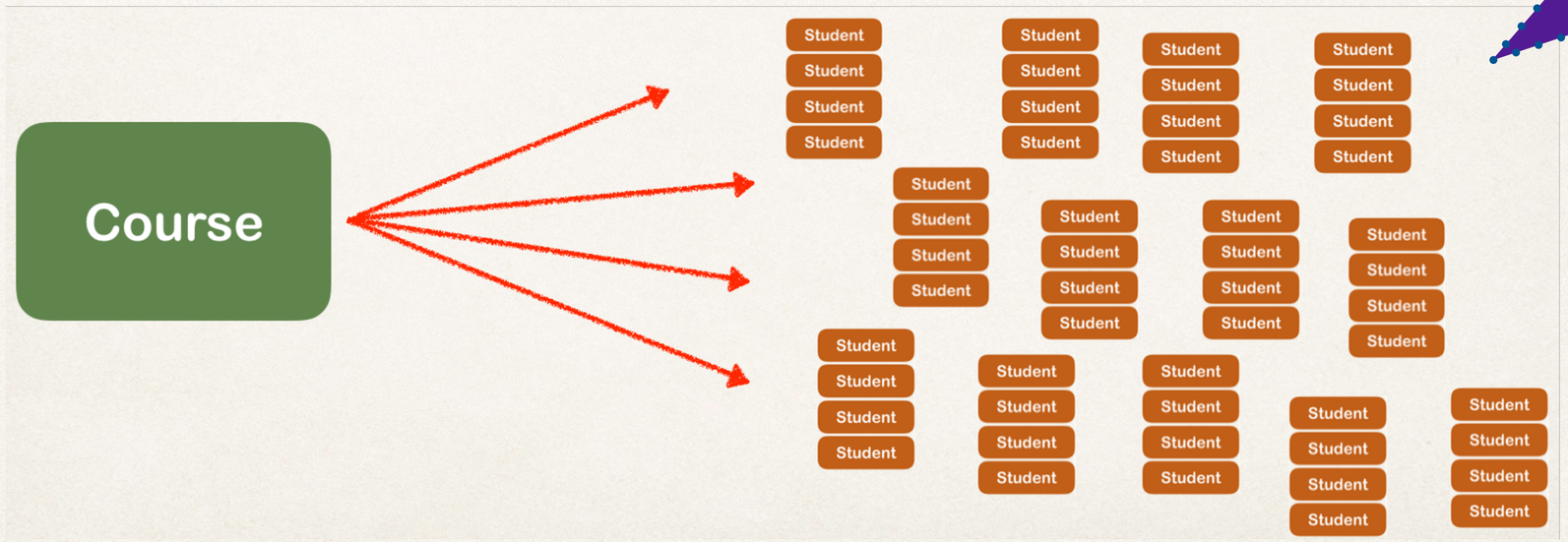
Lazy Loading

- Lazy loading will load the main entity first
  - Load dependent entities on demand (lazy)

Lazy Loading is preferred

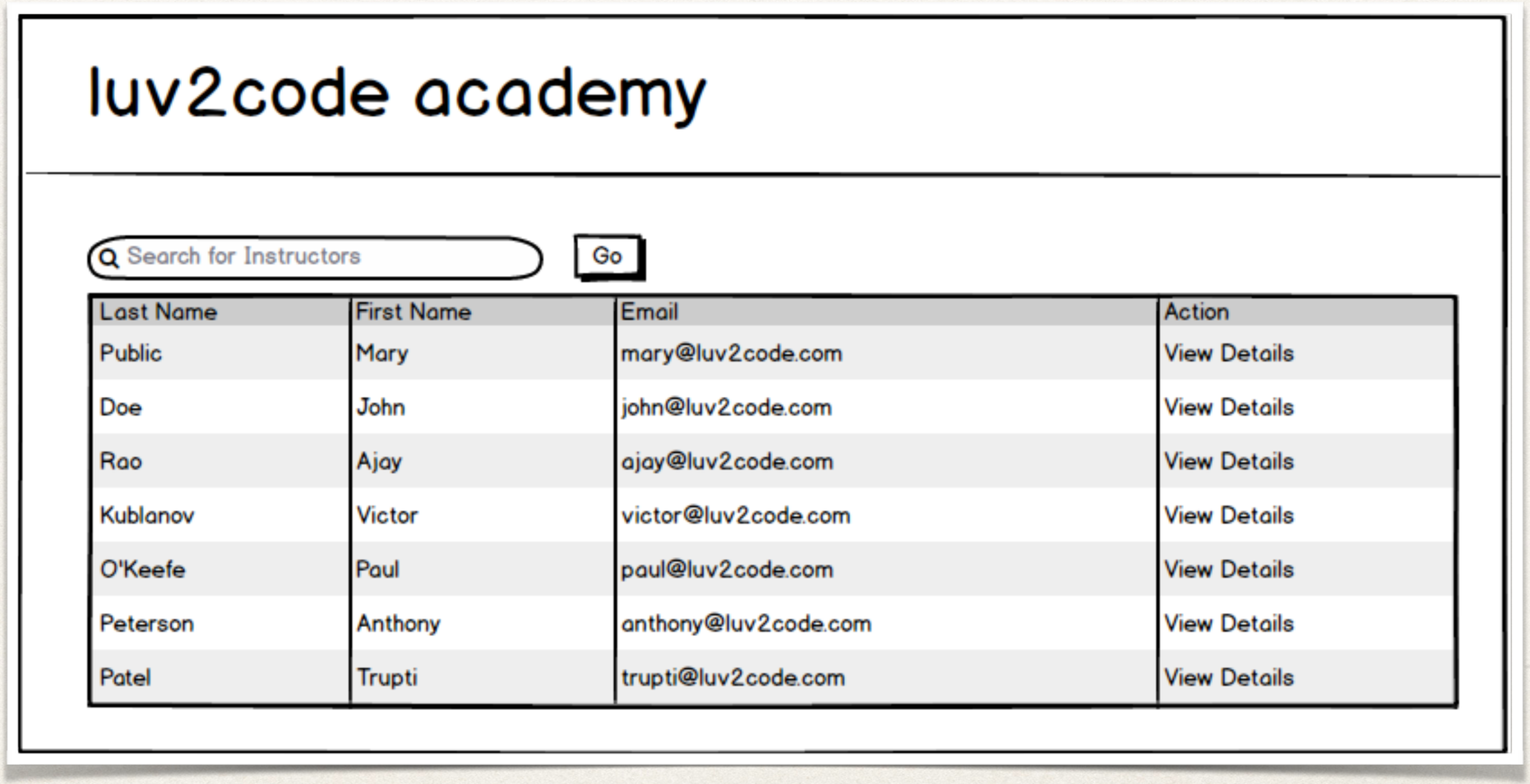
Load on demand (lazy)

Load course first



Real-World Use Case

- Search for instructors

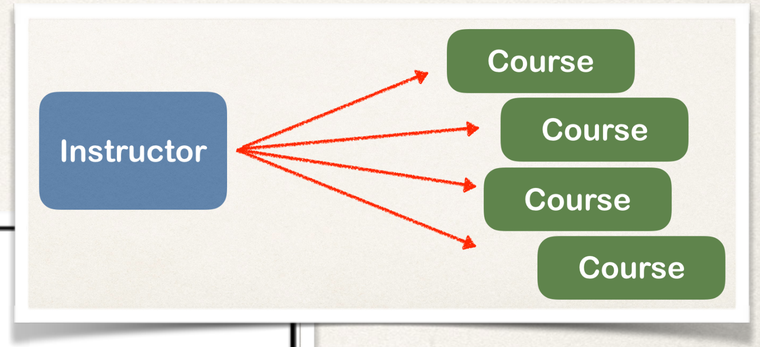


Real-World Use Case

- In Master view, use lazy loading
- In Detail view, retrieve the entity and necessary dependent entities

Real-World Use Case - Master View

- In Master view, use lazy loading for search results
- Only load instructors … not their courses



luv2code academy

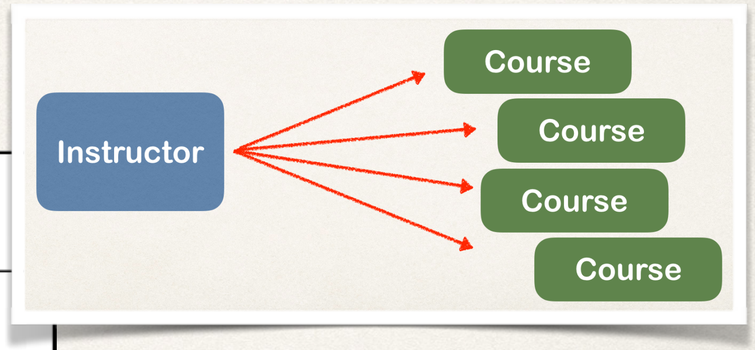
Search for Instructors

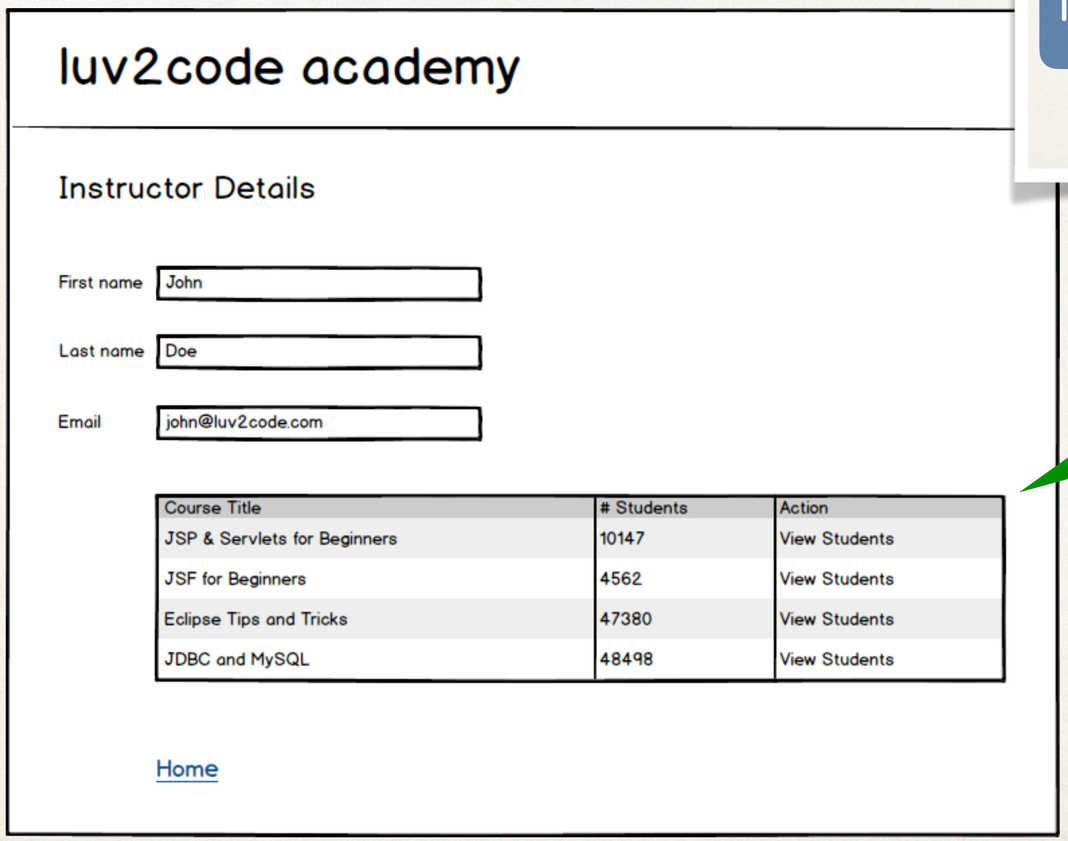
Go

Last Name	First Name	Email	Action
Public	Mary	mary@luv2code.com	View Details
Doe	John	john@luv2code.com	View Details
Rao	Ajay	ajay@luv2code.com	View Details
Kublanov	Victor	victor@luv2code.com	View Details
O'Keefe	Paul	paul@luv2code.com	View Details
Peterson	Anthony	anthony@luv2code.com	View Details
Patel	Trupti	trupti@luv2code.com	View Details

Real-World Use Case - Detail View

- In Detail view, retrieve the entity and necessary dependent entities
- Load instructor AND their courses

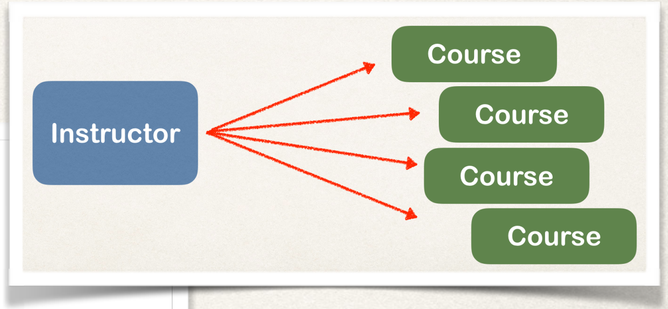




List of courses

Fetch Type

- When you define the mapping relationship
  - You can specify the fetch type: EAGER or LAZY




```
@Entity
@Table(name="instructor")
public class Instructor {
    ...

    @OneToMany(fetch=FetchType.LAZY, mappedBy="instructor")
    private List<Course> courses;

    ...

}
```

Default Fetch Types

Mapping	Default Fetch Type
@OneToOne	FetchType.EAGER
@OneToMany	FetchType.LAZY
@ManyToOne	FetchType.EAGER
@ManyToMany	FetchType.LAZY

Overriding Default Fetch Type

- Specifying the fetch type, overrides the defaults

Mapping	Default Fetch Type
@ManyToOne	FetchType.EAGER

```
@ManyToOne(fetch=FetchType.LAZY)
@JoinColumn(name="instructor_id")
private Instructor instructor;
```

More about Lazy Loading

- When you lazy load, the data is only retrieved on demand
- However, this requires an open Hibernate session
  - need an connection to database to retrieve data

More about Lazy Loading

- If the Hibernate session is closed
  - And you attempt to retrieve lazy data
  - Hibernate will throw an exception

Watch out for this!

More about Lazy Loading

- To retrieve lazy data, you will need to open a Hibernate session

- Retrieve lazy data using
  - Option 1: session.get and call appropriate getter method(s)
  - Option 2: Hibernate query with HQL

- Many other techniques available but the two above are most common