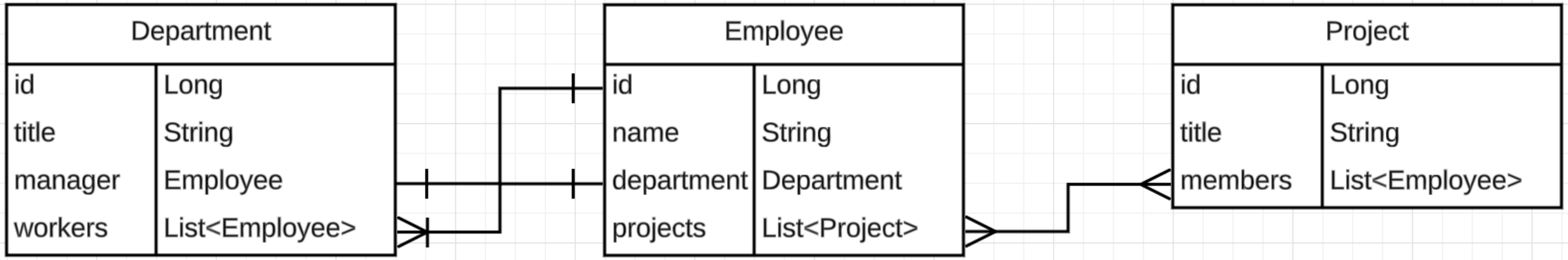
# 3.6

# Mappings - 2



| Department | |
|---|---|
| id | Long |
| title | String |
| manager | Employee |
| workers | List<Employee> |

| Employee | |
|---|---|
| id | Long |
| name | String |
| department | Department |
| projects | List<Project> |

| Project | |
|---|---|
| id | Long |
| title | String |
| members | List<Employee> |

- One Department can have One Manager
- One Department can have many workers

- One Employee can be a worker in One Department
- One Employee can be a manager in One Department
- One Employee can work on multiple Projects.

- One Project can have many Employees working on it.

# One To Many Mappings

- @OneToMany
- @ManyToOne

- @JoinColumn
- @JoinTable

# Cascade

To Propagate the operations from Parent to child

- CascadeType.ALL: Propagates all operations (e.g., persist, merge, remove, refresh, detach) from the parent to the child entity.
- CascadeType.PERSIST: Propagates the persist (save) operation
- CascadeType.MERGE: Propagates the merge operation
- CascadeType.REMOVE: Propagates the remove (delete) operation
- CascadeType.REFRESH: Propagates the refresh
- CascadeType.DETACH: Propagates the detach operation

# Fetch

Loading related entities:

- FetchType.LAZY: Defers the loading of the related entity until it is accessed, optimizing performance by reducing initial data retrieval.
- FetchType.EAGER: Loads the related entity immediately with the initial query, ensuring it is readily available.

# Many To Many Mappings

```
@ManyToMany
@JoinTable(
        name = "freelancer_department",
        joinColumns = @JoinColumn(name = "department_id"),
        inverseJoinColumns = @JoinColumn(name="employee_id")
)
```