CS544 EA

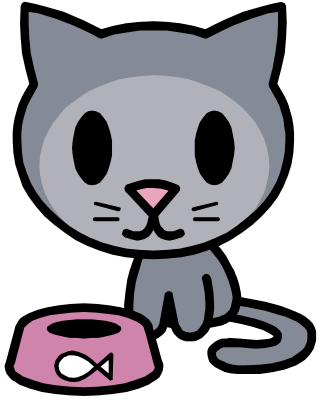
Hibernate

Collection: Map

Map

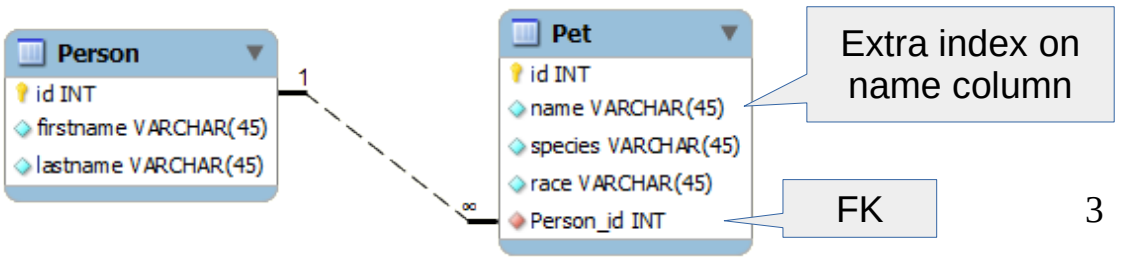
- A map ‘maps’ a set of Keys to a bag of values:
  - **Unique keys that lead to values** (not unique)
  - Given the key, map can quickly get value
  - No built-in order in keys or values (can use @OrderBy)

- Pet ownership can be modeled as a map
  - Each pet has a unique name*
  - To find a pet you use its name
  - No specific order in names or pets



Map Implementation

- Java has the **java.util.Map** interface
  - **java.util.HashMap** is a common implementation
- Maps are indexed collections
  - Need a **FK** and a **index** on another column
  - Can be column of entity, or additional column



Code for Map (no additional column)

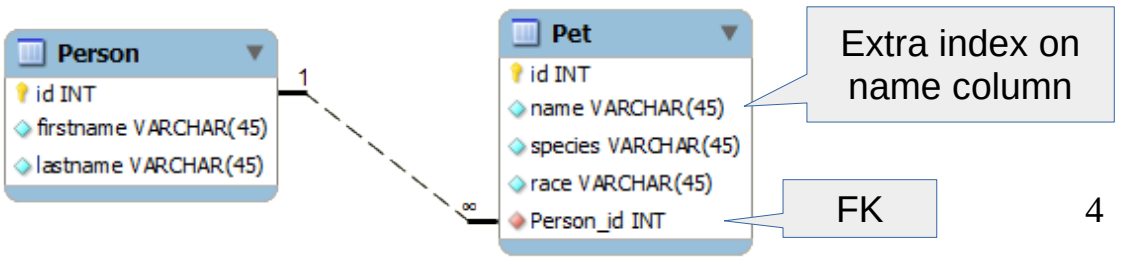
- **@MapKey** if the key is already part of the entity

```java
@Entity
public class Person {
	@Id
	@GeneratedValue
	private Long id;
	private String firstname;
	private String lastname;
	@OneToMany(mappedBy="owner")
	@MapKey(name="name")
	private Map<String, Pet> pets
		= new HashMap<>();
```

Specify property on other side

```java
@Entity
public class Pet {
	@Id
	@GeneratedValue
	private Long id;
	private String species;
	private String race;
	private String name;
	@ManyToOne
	private Person owner;
```

Property used as key



Additional Column & Bi-Directional

- When a collection needs an **additional column**
  - It **needs to be the owning side** of the association
  - Only it knows the value that needs to be inserted

- Bi-directional: collection side **defaults to not being owner**!
  - Side with collection normally gives up control with mappedBy
  - Other side does not even have mappedBy option

Uni-Direct & Additional Column

- **@MapKeyColumn** no problem for uni-directional

```java
@Entity
public class Person {
	@Id
	@GeneratedValue
	private Long id;
	private String firstname;
	private String lastname;
	@OneToMany
	@JoinColumn(name="Person_id")
	@MapKeyColumn(name="name")
	private Map<String, Pet> pets
		= new HashMap<>();
```

No mappedBy

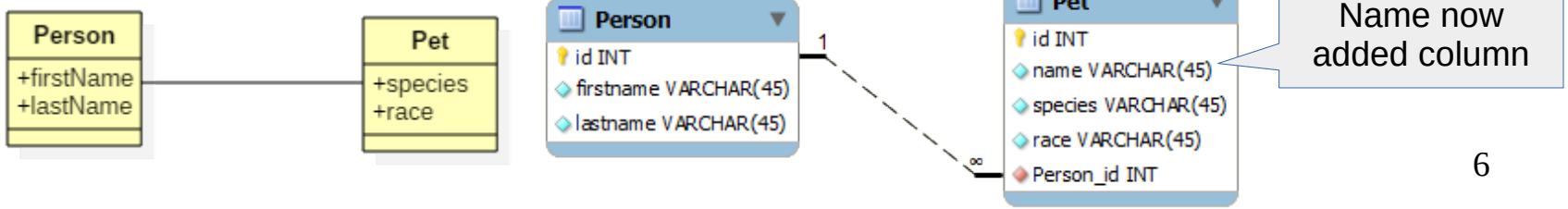
@JoinColumn to avoid join table!

@MaKeyColumn Specifies name of (indexed) column to add

```java
@Entity
public class Pet {
	@Id
	@GeneratedValue
	private Long id;
	private String species;
	private String race;
```

No name property

No reference back (uni)



Bi-Direct @MapKeyColumn

- MappedBy emulation on @JoinColumn for @ManyToOne
  - **insertable=false, updatable=false**

```java
@Entity
public class Person {
	@Id
	@GeneratedValue
	private Long id;
	private String firstname;
	private String lastname;
	@OneToMany
	@JoinColumn(name="Person_id")
	@MapKeyColumn(name="name")
	private Map<String, Pet> pets
		= new HashMap<>();
```

```java
@Entity
public class Pet {
	@Id
	@GeneratedValue
	private Long id;
	private String species;
	private String race;
	@ManyToOne
	@JoinColumn(name="Person_id", insertable=false, updatable=false)
	private Person owner;
```

Gives up control without mappedBy

Same as unidirectional

Both map to same join column.

Avoids race condition by using insertable=false, updatable=false