FIAP GRADUAÇÃO

FIAP

ENTERPRISE APPLICATION DEVELOPMENT

Prof. THIAGO T. I. YAMAMOTO

#07 – MAPEAMENTO AVANÇADO

 thiagoyama  thiagoyama@gmail.com

AGENDA

- Chaves compostas
- Múltiplas tabelas
- Herança

CHAVE COMPOSTA

CHAVE COMPOSTA

FIAP

Tomemos o modelo de reserva de pacotes abaixo;

Nele, podemos verificar a existência de uma tabela associativa **TAB_RESERVA** que possui uma chave primária composta por: **COD_RESERVA**, **COD_CLIENTE** e **COD_PACOTE**;

Além, disso, existe um campo **NUM_DIAS**, próprio da tabela e, portanto, neste caso não podemos utilizar a anotação **@ManyToMany**;

A especificação da **JPA 2.0** facilitou o mapeamento de tabelas com chaves compostas conforme veremos adiante...

CHAVE COMPOSTA

FIAP

Para mapear chaves compostas você deve:

1. Criar uma classe que conterá apenas os atributos correspondentes aos atributos da chave estrangeira (**manter conformidade com os tipos de dados!**);

2. Na entidade em si utilizar a anotação **@IdClass** para indicar a classe criada em 1;

3. Criar na entidade os mesmos atributos que a classe de id (criada em 1) com anotações **@Id**;

4. Para os atributos com anotações @Id que também forem chaves estrangeiras utilizar as anotações **@ManyToOne** e **@JoinColumn** para indicar a coluna de chave estrangeira;

CHAVE COMPOSTA

Criar a classe de chave primária:

```
public class ReservaPK implements Serializable {
	private int id;
	private int cliente;
	private int pacote;
	...
}
```

CHAVE COMPOSTA

FIAP

Criar a classe Reserva com o atributo id próprio:

```
@Entity(name="Reserva")
@Table(name="TAB_RESERVA")
@IdClass(ReservaPK.class)
public class Reserva implements Serializable {


    @Id
    @SequenceGenerator(name="seqReserva",
    sequenceName="SEQ_RESERVA",allocationSize=1)
    @GeneratedValue(generator="seqReserva",
    strategy=GenerationType.SEQUENCE)
    @Column(name="COD_RESERVA")
    private int id;


    @Column(name="DAT_RESERVA")
    private Calendar data;
    ...
}
```

CHAVE COMPOSTA

Os atributos que fazem parte tanto da chave estrangeira quanto da primária ficam assim:

```
@Id
@ManyToOne
@JoinColumn(name="COD_CLIENTE")
private Cliente cliente;
```

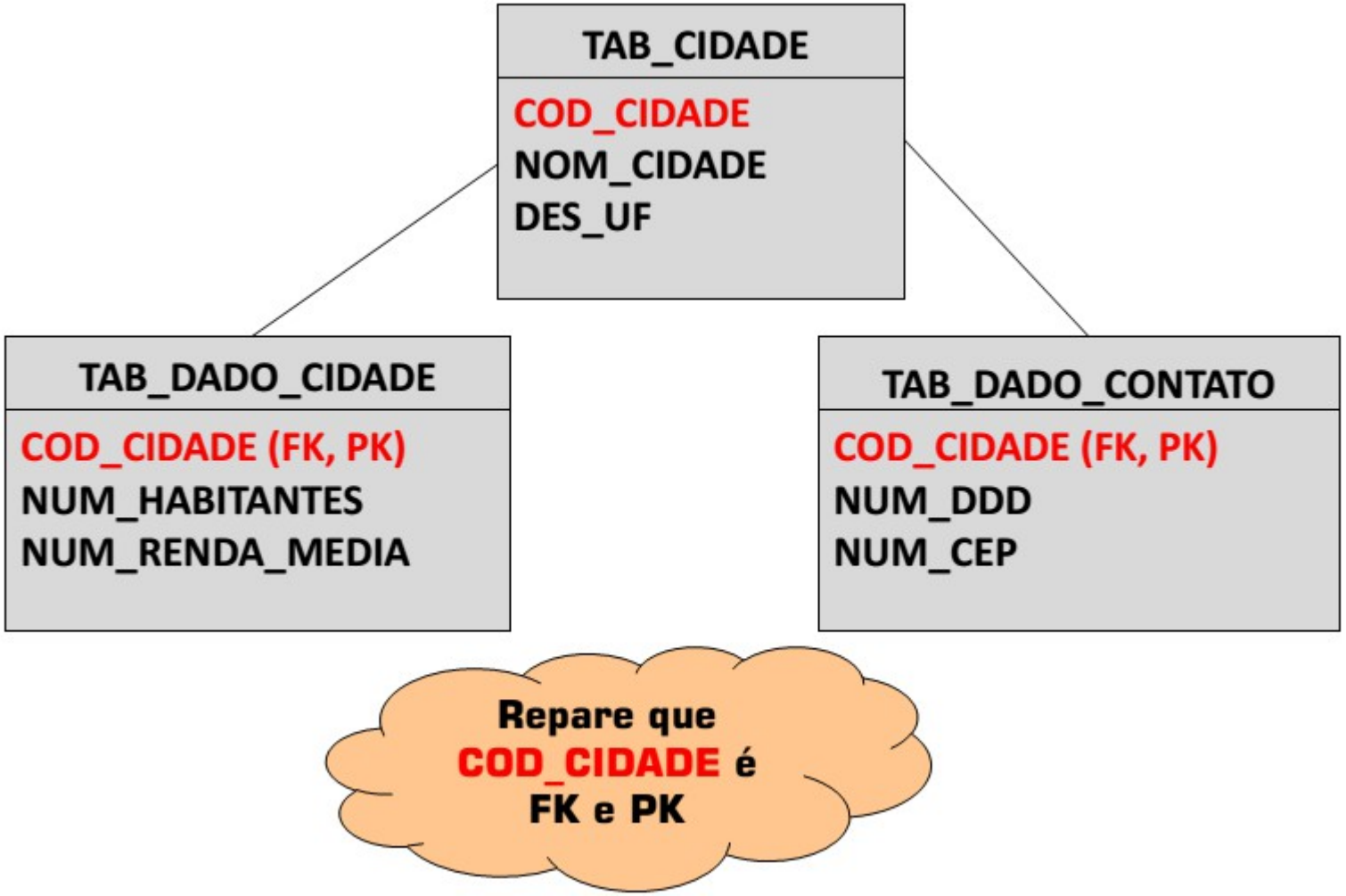
```
@Id
@ManyToOne
@JoinColumn(name="COD_PACOTE")
private Pacote pacote;
```

FIAP

MÚLTIPLAS TABELAS

MÚLTIPLAS TABELAS

Uma única entidade pode ser representada por mais de uma tabela no banco de dados:



MÚLTIPLAS TABELAS

FIAP

1. Criar uma entidade contendo todos os atributos envolvidos nas n tabelas;
2. A entidade deve possuir na classe a anotação **@SecondaryTable** ou **@SecondaryTables** , com os parâmetros:
    1. **name**: nome da tabela secundária
    2. **pkJoinColumns**: colunas envolvidas na chave primária:
        1. **@PrimaryKeyJoinColumn**: nome da coluna de chave primária
3. Cada atributo da tabela secundária anotadas com **@Column** deverão apontar para a tabela secundária por meio do parâmetro **table**.

Exemplo:

```
@SecondaryTable(name="TAB_DADO_CIDADE",
pkJoinColumns={@PrimaryKeyJoinColumn(name="COD_CIDADE")})
public class Cidade {
    @Column(name="NUM_HABITANTES", table="TAB_DADO_CIDADE")
    private int totalHabitantes;
}
```

MÚLTIPLAS TABELAS

FIAP

Existindo mais de uma tabela secundária, basta utilizar a anotação **@SecondaryTables**:

```
@SecondaryTables(value={
      @SecondaryTable(name="TAB_DADO_CIDADE",
      pkJoinColumns={@PrimaryKeyJoinColumn(name="COD_CIDADE")})
      ,
      @SecondaryTable(name="TAB_DADO_CONTATO",
      pkJoinColumns={@PrimaryKeyJoinColumn(name="COD_CIDADE")})
})
public class Cidade {


      @Column(name="NUM_HABITANTES", table="TAB_DADO_CIDADE")
      private int totalHabitantes;


      @Column(name="NUM_DDD", table="TAB_DADO_CONTATO")
      private int ddd;
}
```

MÚLTIPLAS TABELAS

FIAP

Ao persistir uma entidade Cidade parte dos dados será armazenado na TAB_DADO_CIDADE e parte na TAB_DADO_CONTATO:

```
Cidade c = new Cidade();
c.setNome("BAURU");
c.setDdd(14);
c.setTotalHabitantes(300000);
```

TAB_CIDADE	
COD_CIDADE	NOM_CIDADE
1	BAURU

```
... persist(c);
```

TAB_DADO_CIDADE		
COD_CIDADE	NUM_HABITANTES	...
1	300000	...

TAB_DADO_CONTATO		
COD_CIDADE	NUM_DDD	...
1	14	...

FIAP

HERANÇA

HERANÇA

A herança entre classes pode ser também mapeada para o modelo de dados;

Para tanto existe uma anotação **@Inheritance** que define, na classe pai, a estratégia de mapeamento a ser utilizada (parâmetro **strategy**)

Existem três estratégias possíveis (definidas no *enum* **InheritanceType**):

Uma única tabela para toda a hierarquia da herança (SINGLE_TABLE)

- Mais eficiente
- Pode produzir uma tabela com muitos campos

Uma tabela separada para cada subclasse (JOINED)

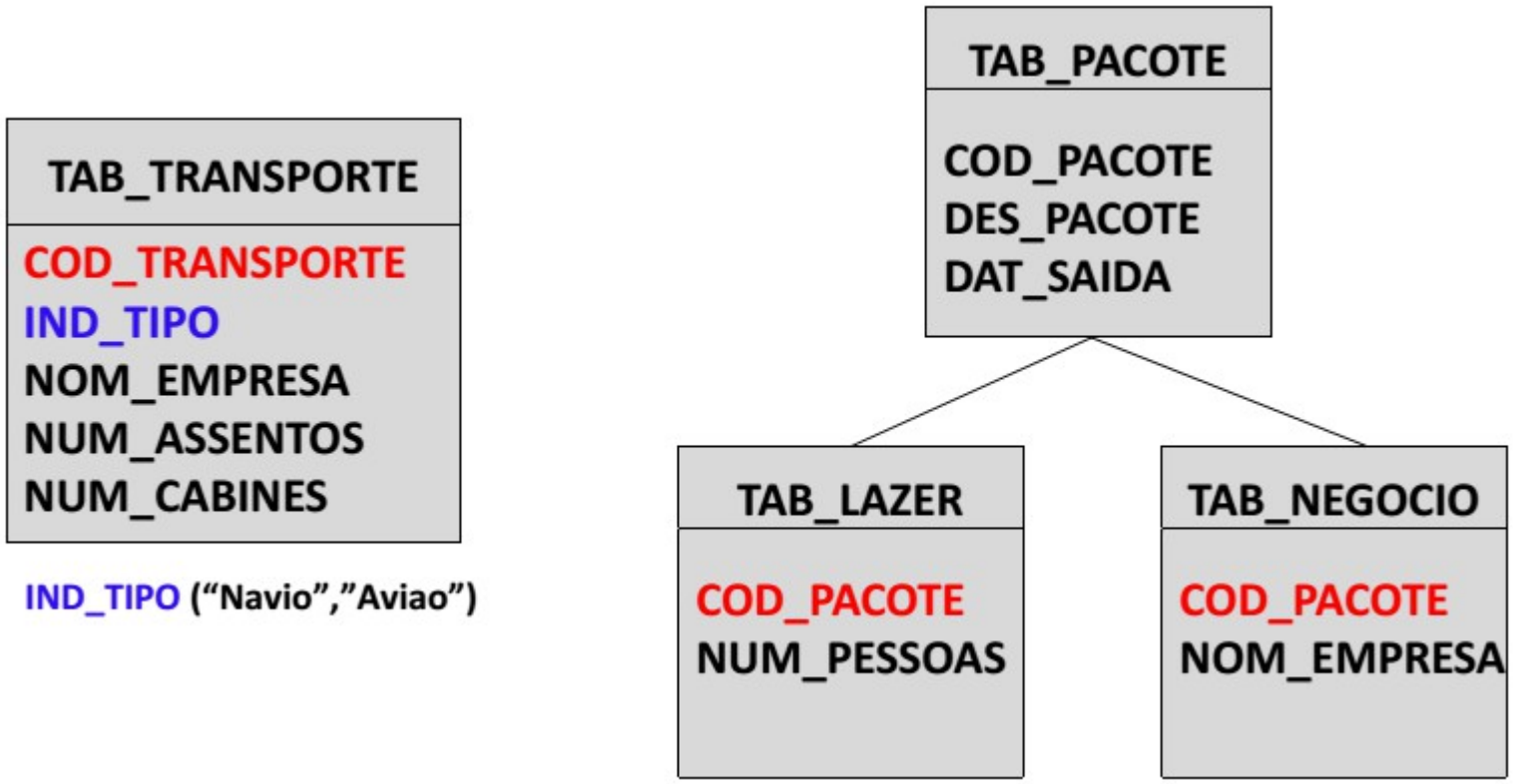
- Necessita de uma operação de *join* entre as tabelas pai e filho
- Cada tabela terá apenas campos específicos da subclasse representada

Uma única tabela para cada classe concreta (TABLE_PER_CLASS)

- Repete-se em cada tabela os atributos da classe filho e pai
- Das três estratégias é a menos utilizada (não estudaremos)

HERANÇA

À esquerda temos um exemplo de **SINGLE_TABLE** (tipos de transporte) e à direita um **JOINED** (tipos de pacotes);



SINGLE TABLE

Na entidade pai deve existir uma anotação indicando o campo na tabela que identificará o tipo de classe representada pelo registro;

@DiscriminatorColumn
- **name**: nome da coluna (do exemplo *IND_TIPO*)
- **discriminatorType**: tipo de dado da coluna (vide *enum* **DiscriminatorType**)

Em todas as entidades (pai e filhas) deve existir uma anotação para indicar qual o valor que o discriminador deve assumir para representar a entidade"(do exemplo: *N* ou *A*);

@DiscriminatorValue
- **value**: valor que tipifica a entidade no banco de dados

Exemplo:

```
@Entity
@DiscriminatorValue(value="A")
public class TransporteAereo extends Transporte

{ ... }
```

JOINED

FIAP

As tabelas filhas devem possui uma chave primária que também será chave estrangeira, apontando para a chave primária da tabela pai;

Nas entidades filhas deve existir uma anotação para indicar o campo de chave primária na tabela pai:

@PrimaryKeyJoinColumn
– **name**: nome do campo de chave primária da tabela pai

Exemplo:

```
@Entity
@Table(name="TAB_NEGOCIO")
@PrimaryKeyJoinColumn(name="COD_PACOTE")
public class Negocio extends Pacote { ... }
```

FIAP

"O sucesso normalmente vem para quem está ocupado demais para pensar nele"