STOCAREA SI ACCESUL LA DATE

Pentru stocarea datelor am folosit un server MySQL, iar pentru proiectarea bazei de date am folosit tool-ul MySQL Workbench.

Baza de date este compusa din 5 tabele: 3 tabele cu informatii propriu-zise (pentru ingrediente, pizza, respectiv comenzi) si doua tabele de legatura (o tabela de legatura intre pizza si ingredientele pe care le contine, respectiv o alta pentru stabilirea legaturii dintre o comanda si pizza pe care le contine).

	Table	Action	Records [1]	Type	Collation	Size	Overhead
☐	ingredients		8	InnoDB	latin1_swedish_ci	16.0 KiB	-
☐	orders		3	InnoDB	latin1_swedish_ci	16.0 KiB	-
☐	orders_pizzas		5	InnoDB	latin1_swedish_ci	48.0 KiB	-
☐	pizzas		4	InnoDB	latin1_swedish_ci	16.0 KiB	-
☐	pizzas_ingredients		9	InnoDB	latin1_swedish_ci	48.0 KiB	-
	5 table(s)	Sum	29	MyISAM	latin1_swedish_ci	144.0 KiB	0 B

Tabela "ingredients" contine informatii despre ingrediente. Fiecare ingredient este unic determinat printr-un id.

	Field	Type	Collation	Attributes	Null	Default
☐	name	varchar(20)	latin1_swedish_ci		Yes	*NULL*
☐	id	int(11)			No	*None*
☐	price	float			Yes	*NULL*

Tabela "pizzas" contine informatii despre pizza. Fiecare pizza este unic determinata printr-un id.

	Field	Type	Collation	Attributes	Null	Default
☐	id	int(11)			No	*None*
☐	name	varchar(45)	latin1_swedish_ci		Yes	*NULL*
☐	description	varchar(45)	latin1_swedish_ci		Yes	*NULL*
☐	permanent	tinyint(4)			Yes	0
☐	link	varchar(45)	latin1_swedish_ci		Yes	*NULL*
☐	date	varchar(45)	latin1_swedish_ci		Yes	*NULL*

Tabela de legatura intre o pizza si ingedientele continute se bazeaza pe doua campuri "foreign key": id_pizza legat de id din tabela "pizzas", respectiv id_ingredient legat de id-ul din "ingredients".

	Field	Type	Collation	Attributes	Null	Default
☐	id_pizza	int(11)			Yes	*NULL*
☐	id_ingredient	int(11)			Yes	*NULL*

Keyname	Type	Unique	Packed	Field
fk_pizzas_ingredients_1	BTREE	No	No	id_pizza
fk_pizzas_ingredients_2	BTREE	No	No	id_ingredient

Tabela "orders" contine informatii cu privire la comenzi. La fel ca celelalte componente, si o comanda este unic determinata printr-un camp numit "id".

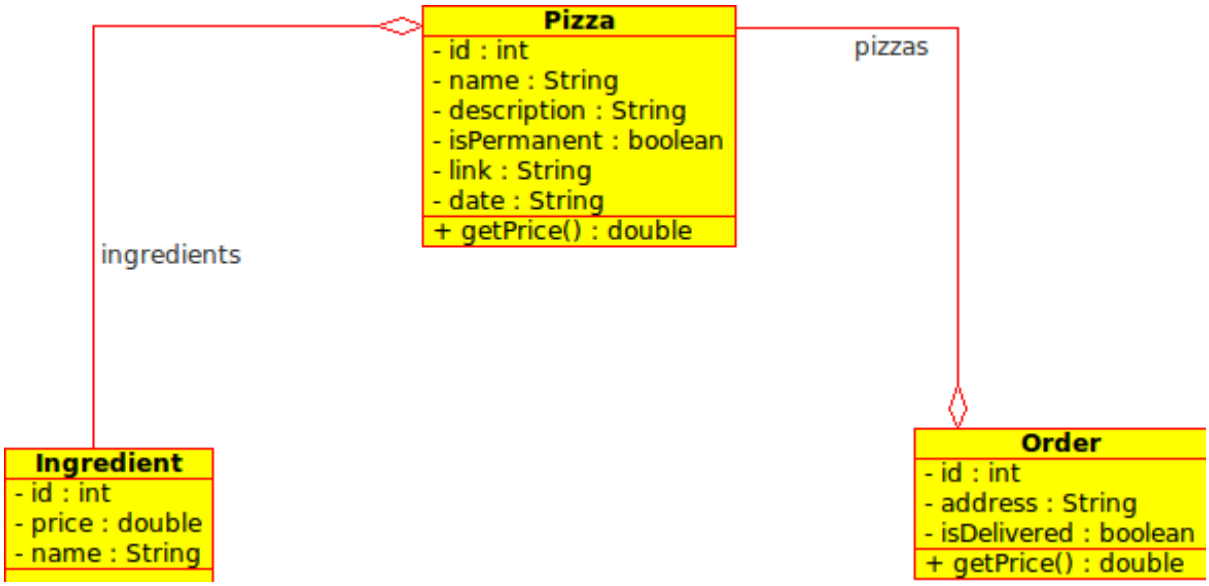
	Field	Type	Collation	Attributes	Null	Default
☐	**id**	int(11)			No	*None*
☐	**address**	varchar(45)	latin1_swedish_ci		Yes	*NULL*
☐	**delivered**	tinyint(4)			Yes	0

Legatura dintre comanda si pizza se realizeaza prin tabela "orders_pizzas", la fel ca mai sus.

	Field	Type	Collation	Attributes	Null	Default
☐	**id_order**	int(11)			Yes	*NULL*
☐	**id_pizza**	int(11)			Yes	*NULL*

Keyname	Type	Unique	Packed	Field
fk_orders_pizzas_1	BTREE	No	No	id_order
fk_orders_pizzas_2	BTREE	No	No	id_pizza

Obiectele java care se mapeaza pe informatiile din aceste tabele sunt definite prin clasele Ingredient, Pizza si respectiv Order din pachetul client, conform diagramei de mai jos:



Maparea propriu-zisa intre aceste obiecte si baza de date se realizeaza prin Hibernate. Legaturile dintre obiecte se implementeaza prin relatia de tipul "one-to-many". De exemplu, maparea unui obiect de tipul Ingredient la tabela ingredients se face in felul urmator:

```
<class
      name="client.Ingredient"
      table="ingredients">
      <id
      name="id"
      column="id">
         <generator class="increment"/>
      </id>
      <property
         name="name"
         column="name"/>
      <property
         name="price"
         column="price"/>
   </class>
```

Ca si exemplu de stabilire a legaturilor dintre obiecte se prezinta Order si Pizza:

```
<class
      name="client.Order"
      table="orders">

(...)

      <set name="pizzas" table="orders_pizzas">
        <key column="id_order"/>
        <many-to-many column="id_pizza"
            unique="true"
            class="client.Pizza"/>
    </set>
</class>
```