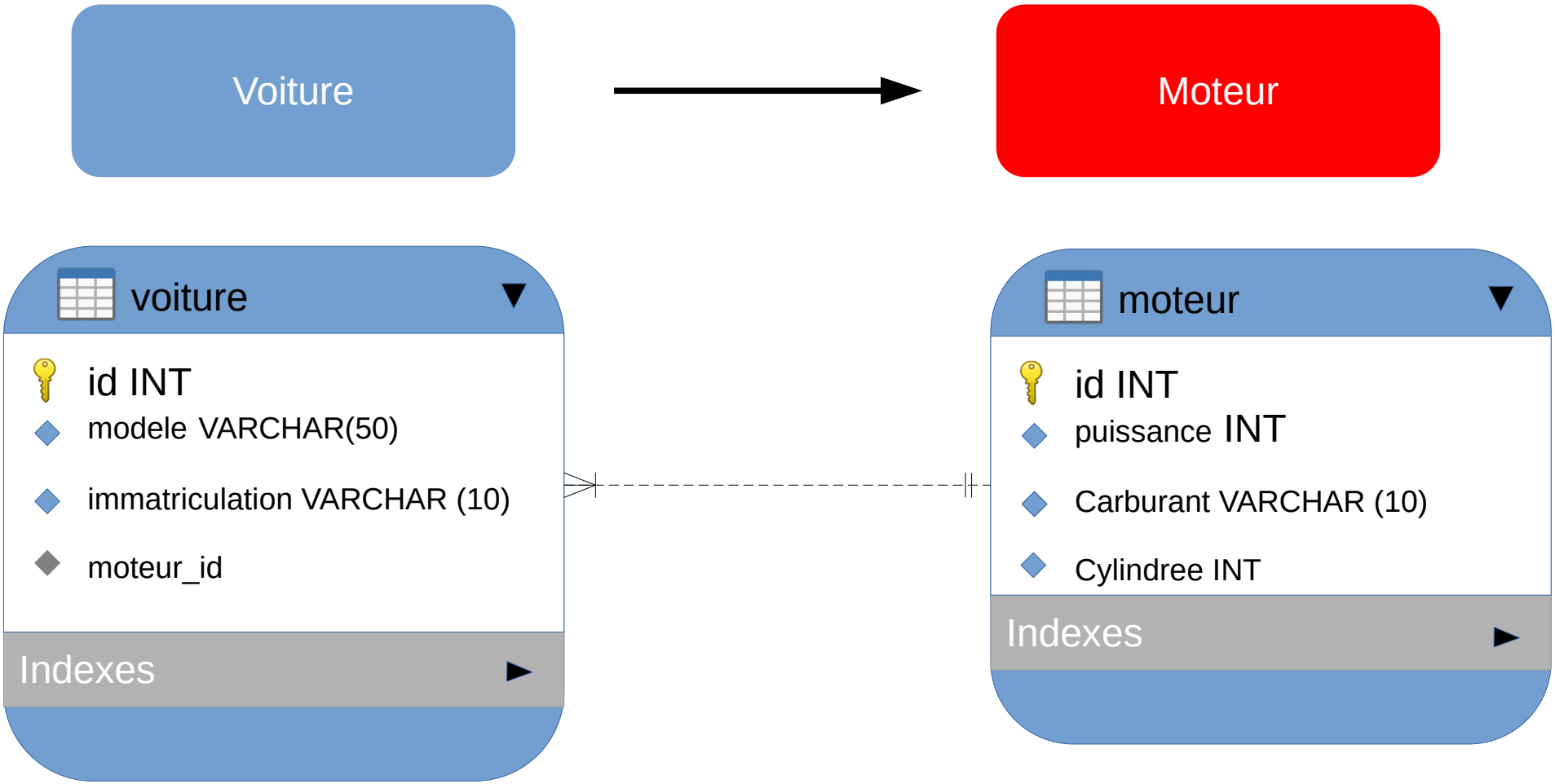
@OneToOne

One-To-One (uni directionnelle)

Une voiture possède un seul moteur.

Et ce moteur fait partie d'une seule voiture



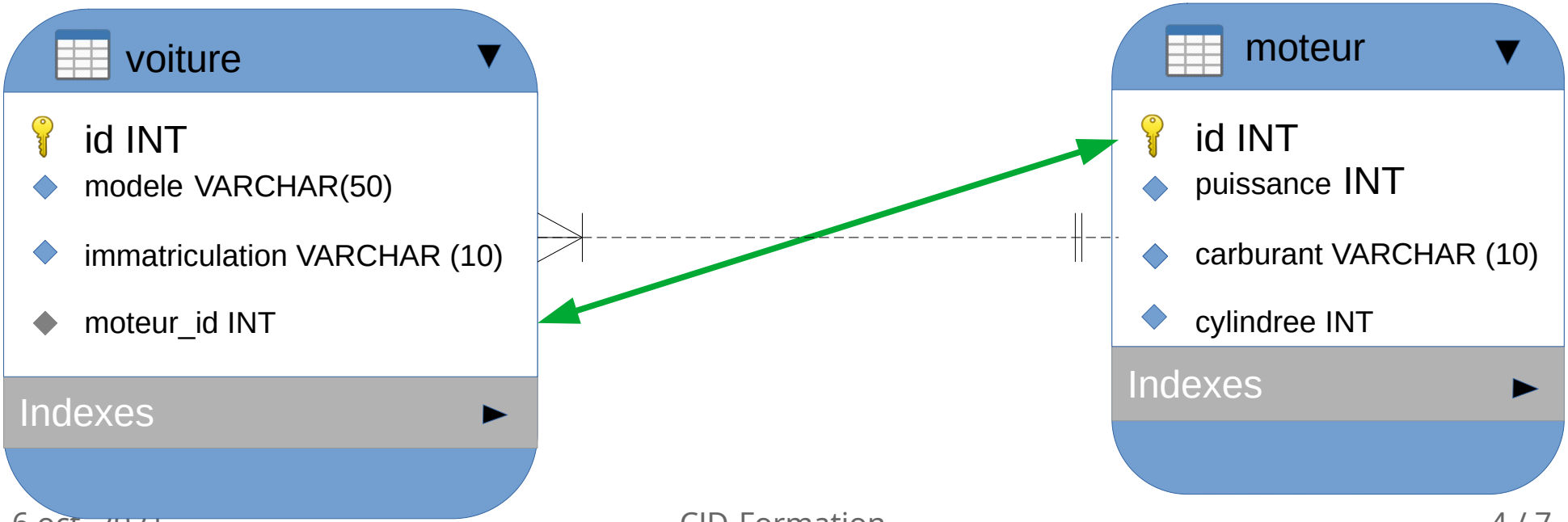
Voici le programme

- Préparer les tables coté BDD (créer les relations entre les tables)
- Coder les classes en entities en java
- Créer une classe executable pour manipuler les entités

Créons les tables

```
CREATE TABLE `moteur` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `puissance` INT DEFAULT NULL,
  `carburant` varchar(10) DEFAULT NULL,
  `cylindree` INT DEFAULT NULL,
  PRIMARY KEY (`id`)
. . .
);
```

```
CREATE TABLE `voiture` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `modele_name` varchar(45) DEFAULT NULL,
  `immatriculation` varchar(10) DEFAULT NULL,
  `moteur_id` INT DEFAULT NULL,
  PRIMARY KEY (`id`)
. . .
) ;
```



Rappel FK

Table : Voiture

id	modele	immatriculation	moteur_id
1	picasso	CDF-548-13	100
2	kangoo	Zg-047-84	200

Colonne
Clé etrangère

Table : Moteur

id	puissance	cylindree	carburant
100	125	1800	essence
200	90	1600	diesel

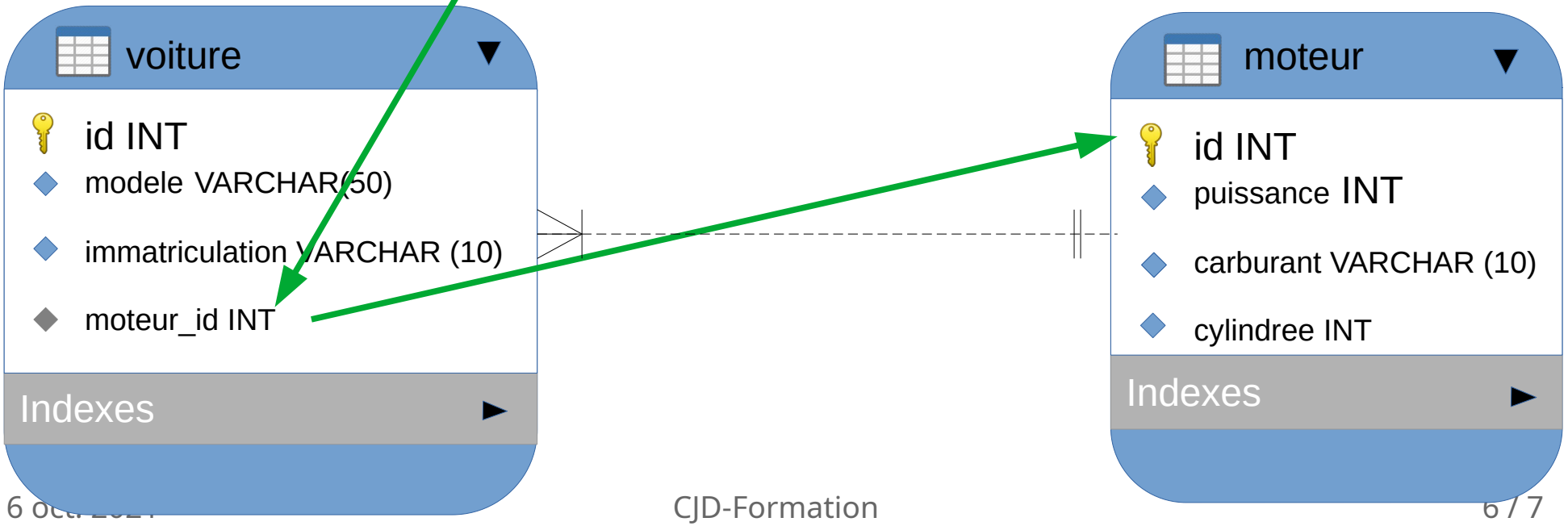
SQL créer la FK

```
CREATE TABLE `voiture` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `modele_name` varchar(45) DEFAULT NULL,
  `immatriculation` varchar(10) DEFAULT NULL,
   `moteur_id` INT DEFAULT NULL,
  PRIMARY KEY (`id`),

  CONSTRAINT    `FK_to_moteur_id`
  FOREIGN KEY   (`moteur_id`)
  REFERENCES    `moteur` (`id`)

) ;
```

Foreign key



FK : intégrité référentielle

- Cette contrainte garanti que la relation ne pourra être détruite que sous certaines conditions
- Que les données insérées devront respecter ces contraintes et seront donc valides.

  La base de données peut à ce titre retourner des erreurs que l'on doit donc anticiper via hibernate comme on le fait en SQL, en s'assurant que l'on suit bien les règles qui régissent notre modèle.