Limitation du modèle à une seule table

- Pour le moment, nous avons manipulé des bases de données contenant une seule et unique table (dotée d'une clé primaire) Ce modèle n'est pas pertinent et conduit à dupliquer l'information. Par exemple pour une base de données de livres, on stockerait sur chaque enregistrement les informations du livre, de l'auteur et de l'éditeur.

Limitation du modèle à une seule table

- Pour le moment, nous avons manipulé des bases de données contenant une seule et unique table (dotée d'une clé primaire) Ce modèle n'est pas pertinent et conduit à dupliquer l'information. Par exemple pour une base de données de livres, on stockerait sur chaque enregistrement les informations du livre, de l'auteur et de l'éditeur.
- Pour de multiples raisons (espace occupé, efficacité pour les recherches ou les modifications, ...) une base de données est constituée d'un ensemble de tables liées entre elles.

C3 Modèle entité-association

1. ??

Limitation du modèle à une seule table

- Pour le moment, nous avons manipulé des bases de données contenant une seule et unique table (dotée d'une clé primaire) Ce modèle n'est pas pertinent et conduit à dupliquer l'information. Par exemple pour une base de données de livres, on stockerait sur chaque enregistrement les informations du livre, de l'auteur et de l'éditeur.
- Pour de multiples raisons (espace occupé, efficacité pour les recherches ou les modifications, ...) une base de données est constituée d'un ensemble de tables liées entre elles.
- Le modèle entité-association permet de concevoir des bases de données de façon efficace.

Définitions

- Une entité est une modélisation d'un objet concret ou abstrait à propos duquel on souhaite conserver des informations. Une entité doit pouvoir être identifiée de façon unique via un identifiant d'identité.

Définitions

- Une entité est une modélisation d'un objet concret ou abstrait à propos duquel on souhaite conserver des informations. Une entité doit pouvoir être identifiée de façon unique via un identifiant d'identité. Par exemple un livre (identifié par son ISBN), une facture (identifié par son code), un client (identifié par son email), un anniversaire (identifié par une personne et une date), une transaction commerciale (identifié par un code) ...

Définitions

- Une entité est une modélisation d'un objet concret ou abstrait à propos duquel on souhaite conserver des informations. Une entité doit pouvoir être identifiée de façon unique via un identifiant d'identité. Par exemple un livre (identifié par son ISBN), une facture (identifié par son code), un client (identifié par son email), un anniversaire (identifié par une personne et une date), une transaction commerciale (identifié par un code) ...
- Une entité possède un ou plusieurs attributs.

Définitions

- Une entité est une modélisation d'un objet concret ou abstrait à propos duquel on souhaite conserver des informations. Une entité doit pouvoir être identifiée de façon unique via un identifiant d'identité. Par exemple un livre (identifié par son ISBN), une facture (identifié par son code), un client (identifié par son email), un anniversaire (identifié par une personne et une date), une transaction commerciale (identifié par un code) ...
- Une entité possède un ou plusieurs attributs. Par exemple, l'entité *film* peut avoir les attributs date, titre, année, ...

Définitions

- Une entité est une modélisation d'un objet concret ou abstrait à propos duquel on souhaite conserver des informations. Une entité doit pouvoir être identifiée de façon unique via un identifiant d'identité. Par exemple un livre (identifié par son ISBN), une facture (identifié par son code), un client (identifié par son email), un anniversaire (identifié par une personne et une date), une transaction commerciale (identifié par un code) ...
- Une entité possède un ou plusieurs attributs. Par exemple, l'entité *film* peut avoir les attributs date, titre, année, ...
- Une instance d'une entité est un objet en particulier.

Définitions

- Une entité est une modélisation d'un objet concret ou abstrait à propos duquel on souhaite conserver des informations. Une entité doit pouvoir être identifiée de façon unique via un identifiant d'identité. Par exemple un livre (identifié par son ISBN), une facture (identifié par son code), un client (identifié par son email), un anniversaire (identifié par une personne et une date), une transaction commerciale (identifié par un code) ...
- Une entité possède un ou plusieurs attributs. Par exemple, l'entité *film* peut avoir les attributs date, titre, année, ...
- Une instance d'une entité est un objet en particulier. Par exemple, *Forrest Gump* est une instance de l'entité *Film*.

Définitions

- Une entité est une modélisation d'un objet concret ou abstrait à propos duquel on souhaite conserver des informations. Une entité doit pouvoir être identifiée de façon unique via un identifiant d'identité. Par exemple un livre (identifié par son ISBN), une facture (identifié par son code), un client (identifié par son email), un anniversaire (identifié par une personne et une date), une transaction commerciale (identifié par un code) ...
- Une entité possède un ou plusieurs attributs. Par exemple, l'entité *film* peut avoir les attributs date, titre, année, ...
- Une instance d'une entité est un objet en particulier. Par exemple, *Forrest Gump* est une instance de l'entité *Film*.
- Une association est un lien entre plusieurs entités. Le degré d'une association est le nombre d'entités intervenant dans l'association.

Définitions

- Une entité est une modélisation d'un objet concret ou abstrait à propos duquel on souhaite conserver des informations. Une entité doit pouvoir être identifiée de façon unique via un identifiant d'identité. Par exemple un livre (identifié par son ISBN), une facture (identifié par son code), un client (identifié par son email), un anniversaire (identifié par une personne et une date), une transaction commerciale (identifié par un code) ...
- Une entité possède un ou plusieurs attributs. Par exemple, l'entité *film* peut avoir les attributs date, titre, année, ...
- Une instance d'une entité est un objet en particulier. Par exemple, *Forrest Gump* est une instance de l'entité *Film*.
- Une association est un lien entre plusieurs entités. Le degré d'une association est le nombre d'entités intervenant dans l'association. Par exemple, l'association *écrit* de degré 2, relie l'entité *auteur* à l'entité *livre*

C3 Modèle entité-association

3. ??

Définitions

Définitions

- Pour les associations de degré 2 (binaire), on précise de chaque côté d'une association le nombre d'entités concernées. C'est la cardinalité de l'association qui se résume à trois types principaux :

Définitions

- Pour les associations de degré 2 (binaire), on précise de chaque côté d'une association le nombre d'entités concernées. C'est la cardinalité de l'association qui se résume à trois types principaux :
  - 1–1 association directe et exclusive entre deux entités (*one to one*).

Définitions

- Pour les associations de degré 2 (binaire), on précise de chaque côté d'une association le nombre d'entités concernées. C'est la cardinalité de l'association qui se résume à trois types principaux :
  - 1–1 association directe et exclusive entre deux entités (*one to one*). Par exemple, un *lycée* a un *proviseur*, un *pays* a une seule *capitale*.

Définitions

- Pour les associations de degré 2 (binaire), on précise de chaque côté d'une association le nombre d'entités concernées. C'est la cardinalité de l'association qui se résume à trois types principaux :
  - 1–1 association directe et exclusive entre deux entités (*one to one*). Par exemple, un *lycée* a un *proviseur*, un *pays* a une seule *capitale*.
  - 1–* (aussi noté 1–1..N) association d'une instance de la première entité à un ensemble d'instances de la seconde (*one to many*).

Définitions

- Pour les associations de degré 2 (binaire), on précise de chaque côté d'une association le nombre d'entités concernées. C'est la cardinalité de l'association qui se résume à trois types principaux :
  - 1–1 association directe et exclusive entre deux entités (*one to one*). Par exemple, un *lycée* a un *proviseur*, un *pays* a une seule *capitale*.
  - 1–* (aussi noté 1–1..N) association d'une instance de la première entité à un ensemble d'instances de la seconde (*one to many*). Par exemple, un *propriétaire* peut avoir plusieurs *voitures*, un *client* peut avoir plusieurs *numéro de téléphone*.

Définitions

- Pour les associations de degré 2 (binaire), on précise de chaque côté d'une association le nombre d'entités concernées. C'est la cardinalité de l'association qui se résume à trois types principaux :
  - 1–1 association directe et exclusive entre deux entités (*one to one*). Par exemple, un *lycée* a un *proviseur*, un *pays* a une seule *capitale*.
  - 1–* (aussi noté 1–1..N) association d'une instance de la première entité à un ensemble d'instances de la seconde (*one to many*). Par exemple, un *propriétaire* peut avoir plusieurs *voitures*, un *client* peut avoir plusieurs *numéro de téléphone*.
  - *–* (aussi noté 1..N–1..N) association d'un ensemble d'instances à un autre ensemble d'instance.

Définitions

- Pour les associations de degré 2 (binaire), on précise de chaque côté d'une association le nombre d'entités concernées. C'est la cardinalité de l'association qui se résume à trois types principaux :
  - 1–1 association directe et exclusive entre deux entités (*one to one*). Par exemple, un *lycée* a un *proviseur*, un *pays* a une seule *capitale*.
  - 1–* (aussi noté 1–1..N) association d'une instance de la première entité à un ensemble d'instances de la seconde (*one to many*). Par exemple, un *propriétaire* peut avoir plusieurs *voitures*, un *client* peut avoir plusieurs *numéro de téléphone*.
  - *–* (aussi noté 1..N–1..N) association d'un ensemble d'instances à un autre ensemble d'instance. Par exemple, un *livre* peut avoir plusieurs *auteurs* et un *auteur* peut écrire plusieurs *livres*.

Définitions

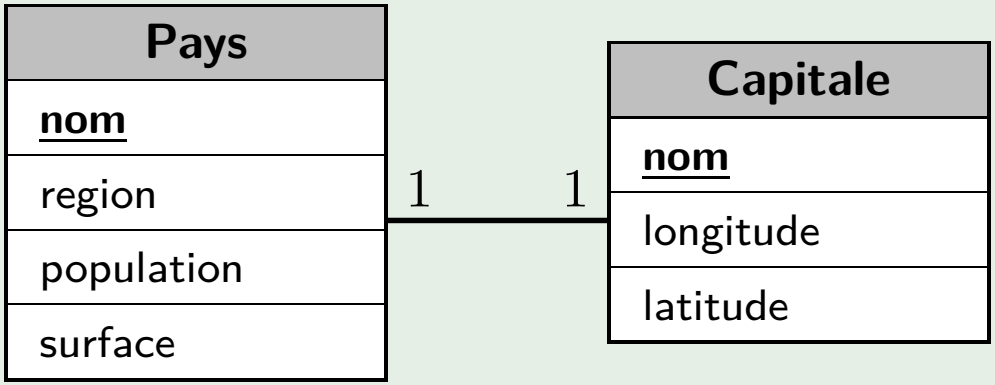
- Pour les associations de degré 2 (binaire), on précise de chaque côté d'une association le nombre d'entités concernées. C'est la cardinalité de l'association qui se résume à trois types principaux :
  - 1–1 association directe et exclusive entre deux entités (*one to one*). Par exemple, un *lycée* a un *proviseur*, un *pays* a une seule *capitale*.
  - 1–* (aussi noté 1–1..N) association d'une instance de la première entité à un ensemble d'instances de la seconde (*one to many*). Par exemple, un *propriétaire* peut avoir plusieurs *voitures*, un *client* peut avoir plusieurs *numéro de téléphone*.
  - *–* (aussi noté 1..N–1..N) association d'un ensemble d'instances à un autre ensemble d'instance. Par exemple, un *livre* peut avoir plusieurs *auteurs* et un *auteur* peut écrire plusieurs *livres*.
- Les associations de types *–* peuvent être séparées entre deux associations de type 1–* à l'aide d'une nouvelle entité.

Définitions

- Pour les associations de degré 2 (binaire), on précise de chaque côté d'une association le nombre d'entités concernées. C'est la cardinalité de l'association qui se résume à trois types principaux :
  - 1–1 association directe et exclusive entre deux entités (*one to one*). Par exemple, un *lycée* a un *proviseur*, un *pays* a une seule *capitale*.
  - 1–* (aussi noté 1–1..N) association d'une instance de la première entité à un ensemble d'instances de la seconde (*one to many*). Par exemple, un *propriétaire* peut avoir plusieurs *voitures*, un *client* peut avoir plusieurs *numéro de téléphone*.
  - *–* (aussi noté 1..N–1..N) association d'un ensemble d'instances à un autre ensemble d'instance. Par exemple, un *livre* peut avoir plusieurs *auteurs* et un *auteur* peut écrire plusieurs *livres*.
- Les associations de types *–* peuvent être séparées entre deux associations de type 1–* à l'aide d'une nouvelle entité.
  Par exemple, en créant une entité *attribution*, un *livre* a plusieurs *attributions* (car il a été écrit par plusieurs *auteurs*) et un auteur à plusieurs *attributions* (car il a écrit plusieurs livres)

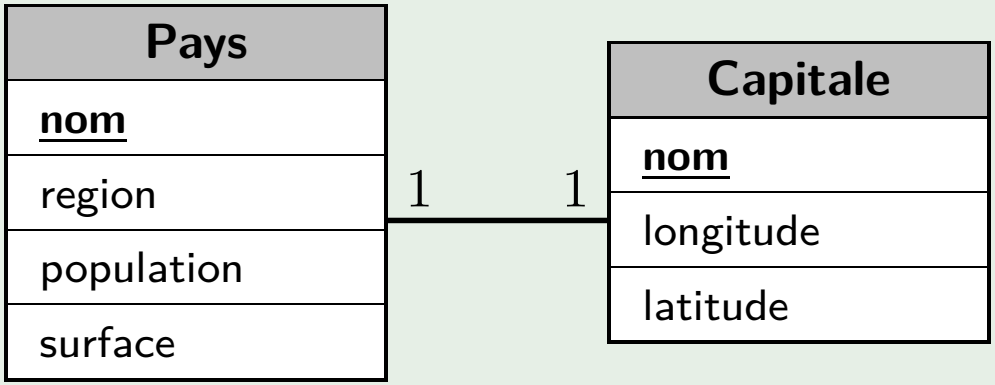
Exemples

- Un exemple d'association *one to one* :

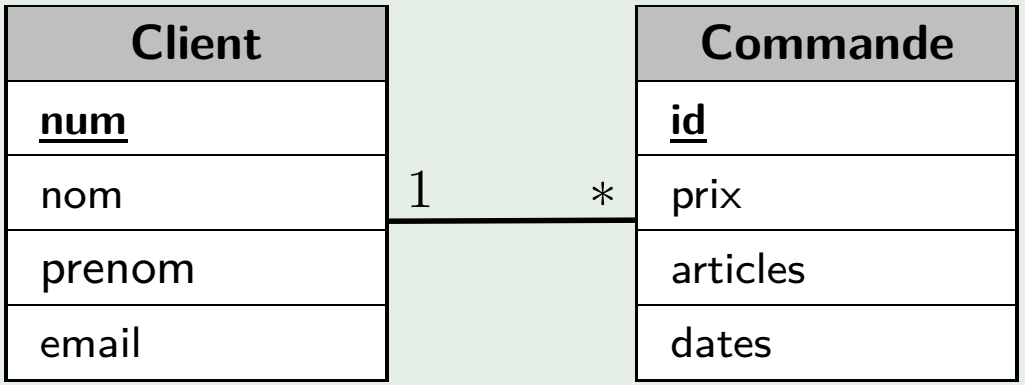


Exemples

- Un exemple d'association *one to one* :



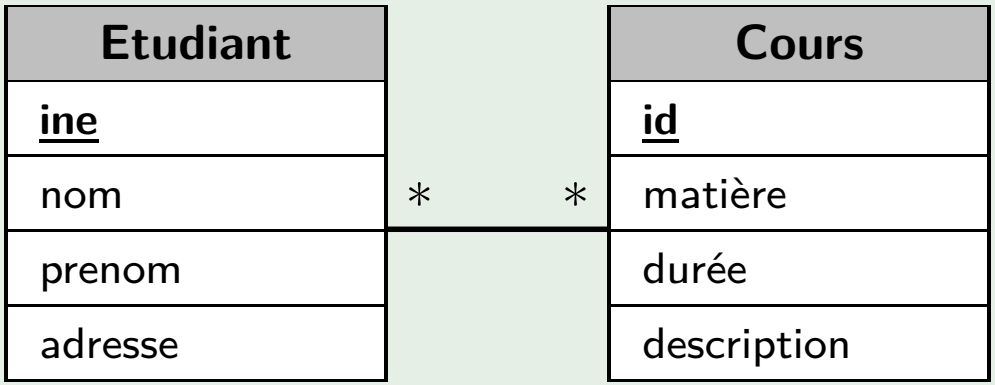
- Un exemple d'association *one to many* :



Exemples

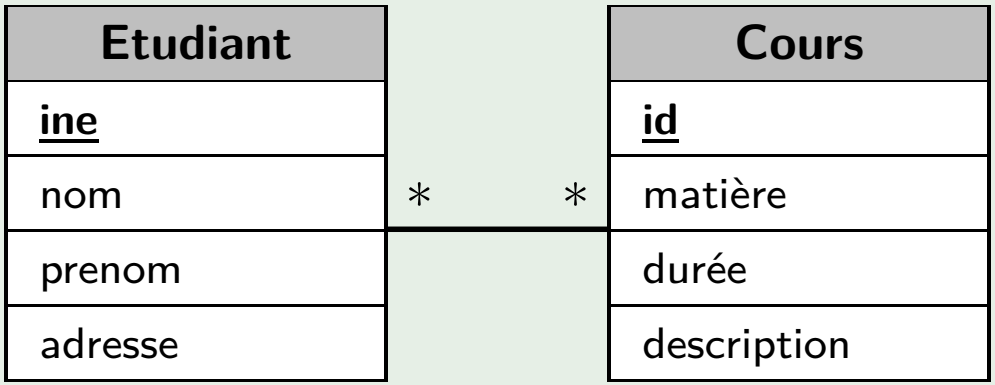
Exemples

- Un exemple d'association *many to many* :

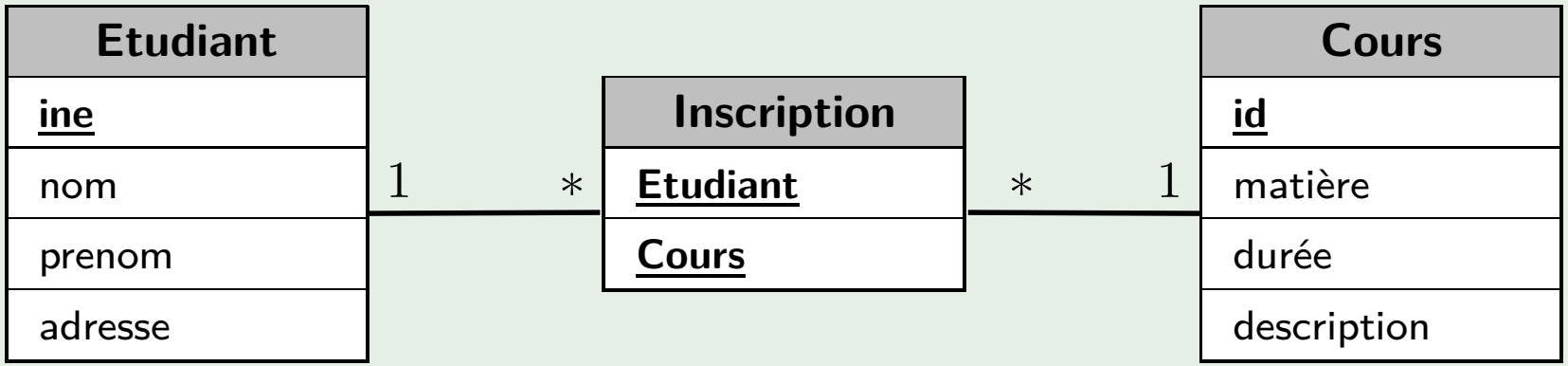


Exemples

- Un exemple d'association *many to many* :



- Sa transformation en deux associations *one to many* à l'aide d'une table de liaison



Méthode

Pour passer du modèle entité association au modèle relationnel :

- Une entité devient une relation (c'est à dire une table)

Méthode

Pour passer du modèle entité association au modèle relationnel :

- Une entité devient une relation (c'est à dire une table)
- L'identifiant d'identité devient la clé primaire de cette table

Méthode

Pour passer du modèle entité association au modèle relationnel :

- Une entité devient une relation (c'est à dire une table)
- L'identifiant d'identité devient la clé primaire de cette table
- On transforme les associations suivant les cas de figure

Cas des associations *one to one* : fusion

Deux entités associées en *one to one* peuvent fusionner dans la même relation.

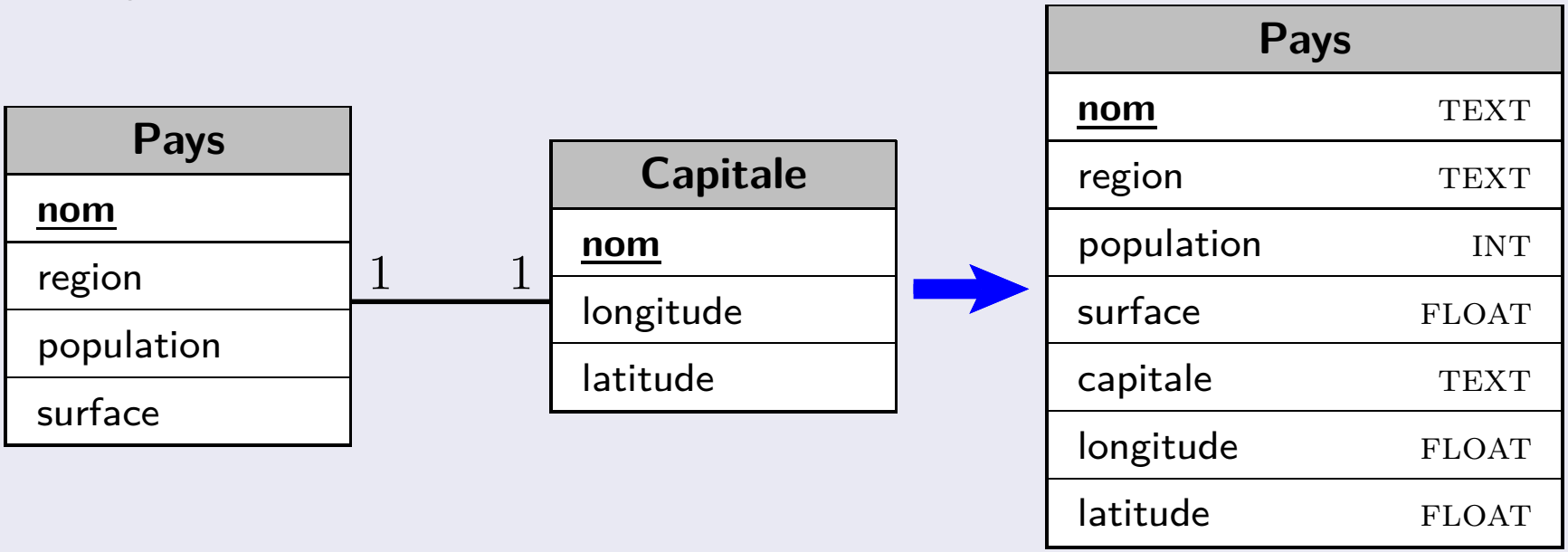
Cas des associations *one to one* : fusion

Deux entités associées en *one to one* peuvent fusionner dans la même relation.

Par exemple, les entités *pays* et *capitale* peuvent fusionner dans une seule table *pays* en ajoutant dans cette table les attributs des capitales.

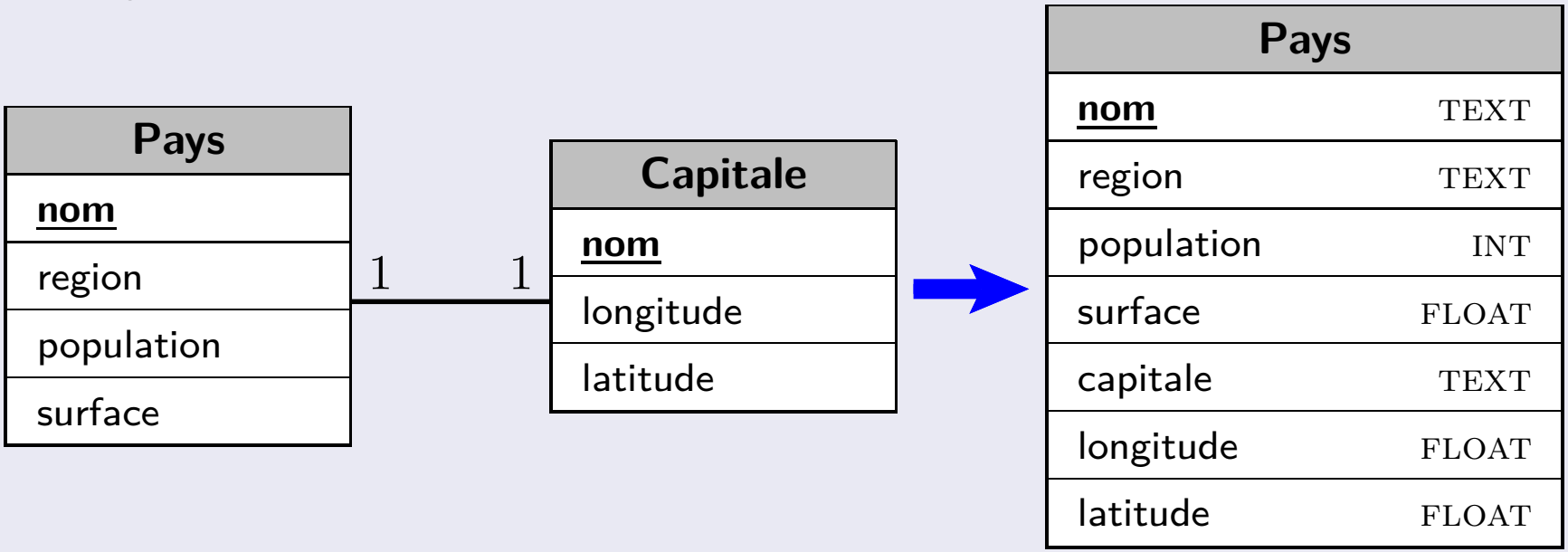
Cas des associations *one to one* : fusion

Deux entités associées en *one to one* peuvent fusionner dans la même relation.
Par exemple, les entités *pays* et *capitale* peuvent fusionner dans une seule table *pays* en ajoutant dans cette table les attributs des capitales.



Cas des associations *one to one* : fusion

Deux entités associées en *one to one* peuvent fusionner dans la même relation.
Par exemple, les entités *pays* et *capitale* peuvent fusionner dans une seule table *pays* en ajoutant dans cette table les attributs des capitales.



On obtient alors le schéma relationnel suivant :

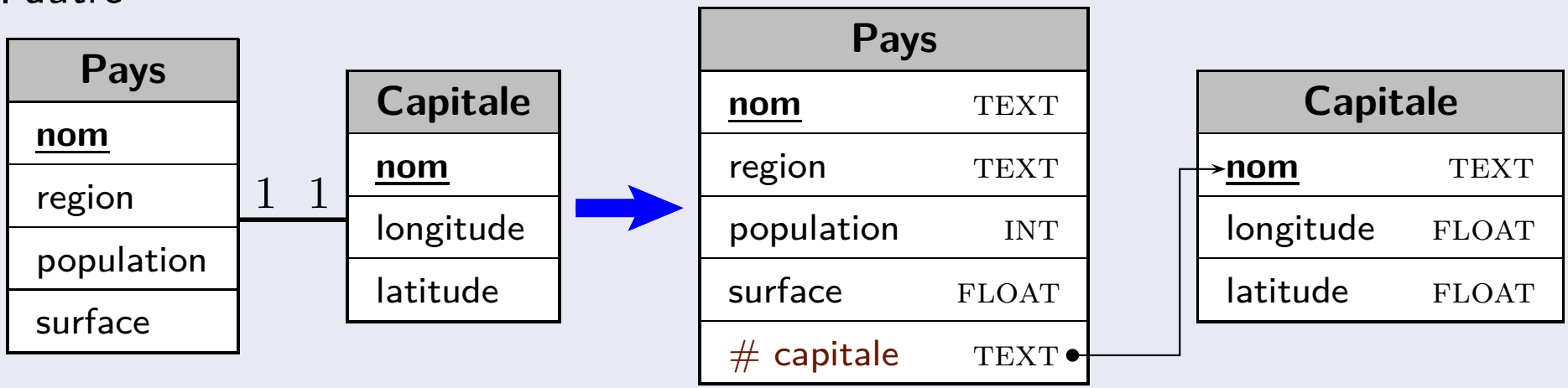
Pays (nom, region, population, surface, capitale, longitude, latitude)

Cas des associations *one to one* : clé étrangère

on peut aussi choisir de garder les deux entités séparées et donc dans deux relations différentes, on introduit alors le concept de clé étrangère c'est à dire la clé primaire d'une autre table qui indique dans l'une des tables la référence vers l'autre

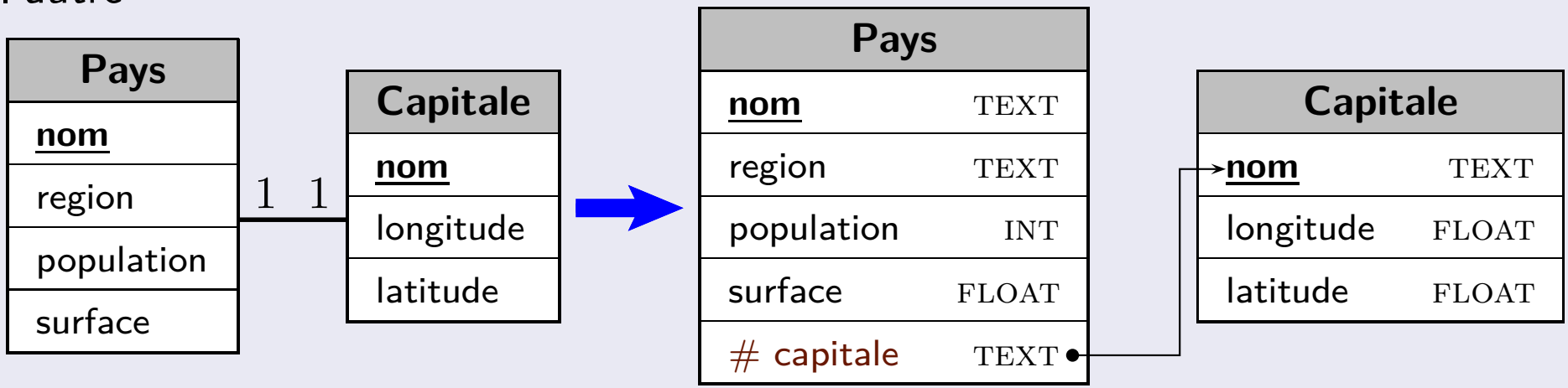
Cas des associations *one to one* : clé étrangère

on peut aussi choisir de garder les deux entités séparées et donc dans deux relations différentes, on introduit alors le concept de clé étrangère c'est à dire la clé primaire d'une autre table qui indique dans l'une des tables la référence vers l'autre



Cas des associations *one to one* : clé étrangère

on peut aussi choisir de garder les deux entités séparées et donc dans deux relations différentes, on introduit alors le concept de clé étrangère c'est à dire la clé primaire d'une autre table qui indique dans l'une des tables la référence vers l'autre

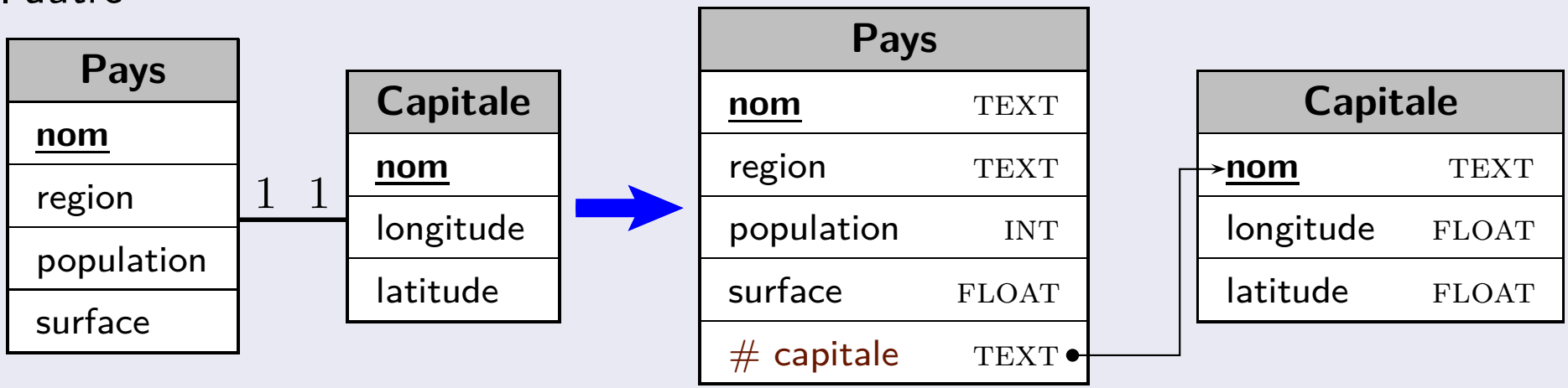


On obtient alors le schéma relationnel suivant :

Pays (nom, region, population, surface, #capitale)
Capitale (nom, longitude, latitude)

Cas des associations *one to one* : clé étrangère

on peut aussi choisir de garder les deux entités séparées et donc dans deux relations différentes, on introduit alors le concept de clé étrangère c'est à dire la clé primaire d'une autre table qui indique dans l'une des tables la référence vers l'autre



On obtient alors le schéma relationnel suivant :

Pays (nom, region, population, surface, #capitale)
Capitale (nom, longitude, latitude)

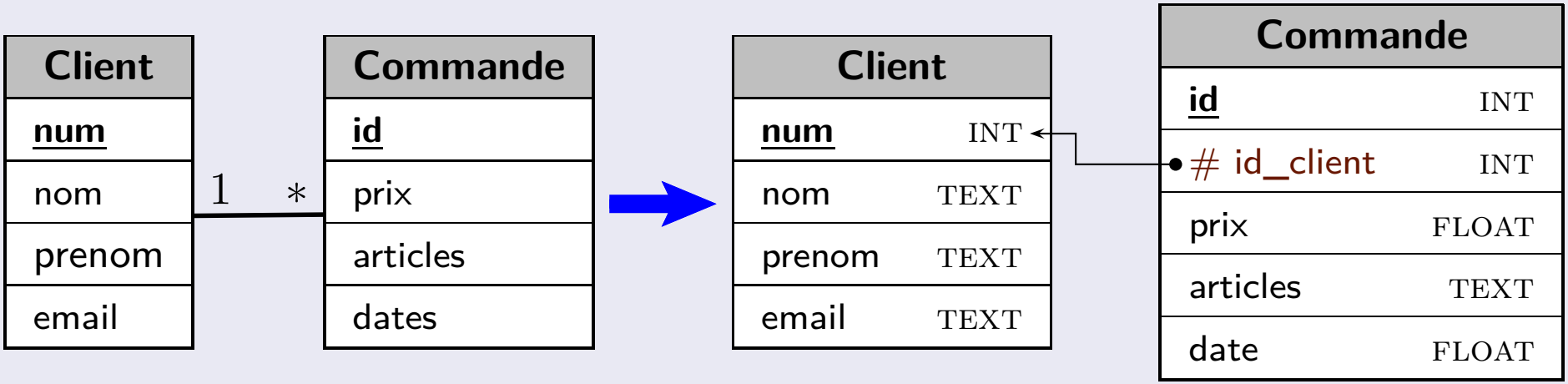
⚠ Intégrité référentielle : un pays doit avoir une capitale !

Cas des associations *one to many*

On utilise là aussi la clé étrangère de façon à ce qu'un élément du côté *"many"* de l'association soit associé à un unique élément du côté *"one"*.

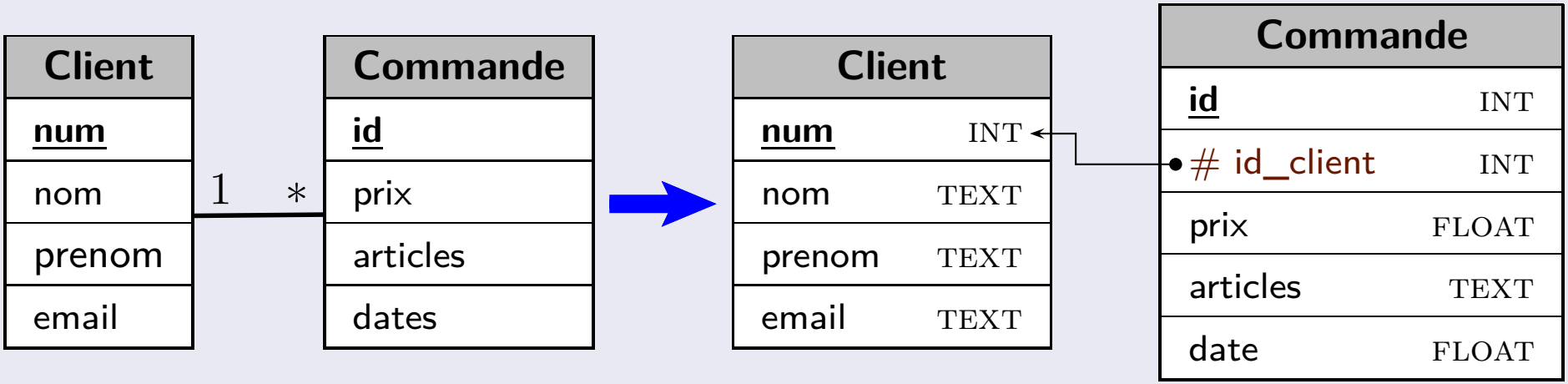
Cas des associations *one to many*

On utilise là aussi la clé étrangère de façon à ce qu'un élément du côté *"many"* de l'association soit associé à un unique élément du côté *"one"*.



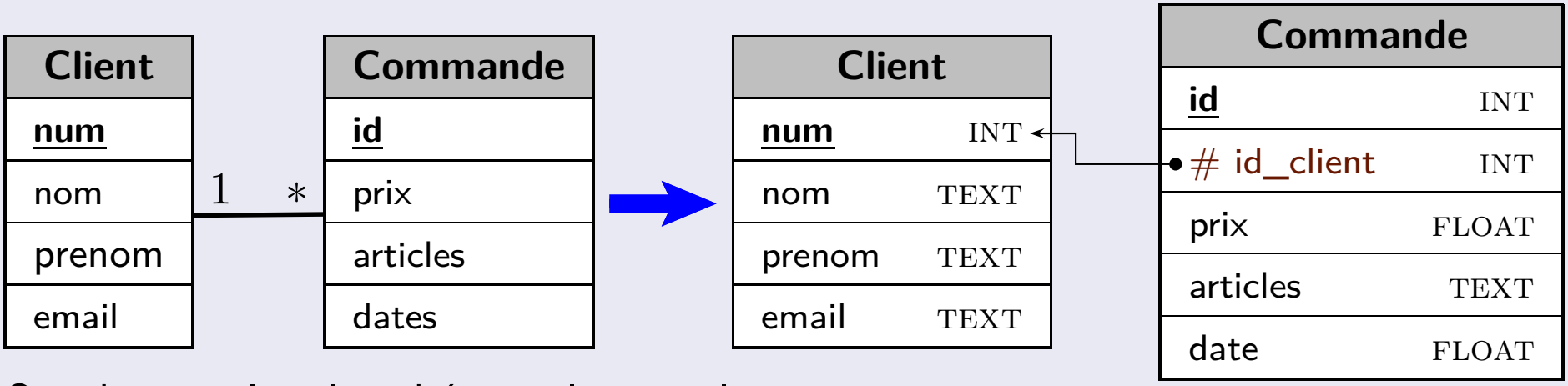
Cas des associations *one to many*

On utilise là aussi la clé étrangère de façon à ce qu'un élément du côté *"many"* de l'association soit associé à un unique élément du côté *"one"*.



Cas des associations *one to many*

On utilise là aussi la clé étrangère de façon à ce qu'un élément du côté *"many"* de l'association soit associé à un unique élément du côté *"one"*.



On obtient alors le schéma relationnel suivant :

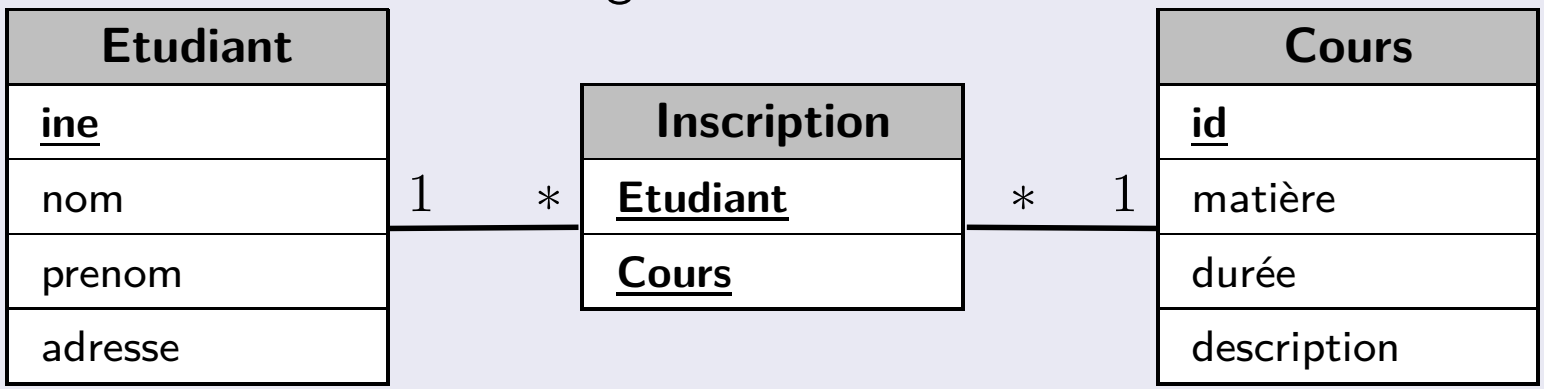
Client (num, nom, prenom, email)
Commande (id, # id_client, prix, articles, date)

Cas des associations *many to many*

On crée trois tables : une pour chacune des entités et la table de liaison, celle-ci a pour clé primaire l'union des clés primaires des deux entités et est en liaison avec celles-ci en utilisant des clés étrangères.

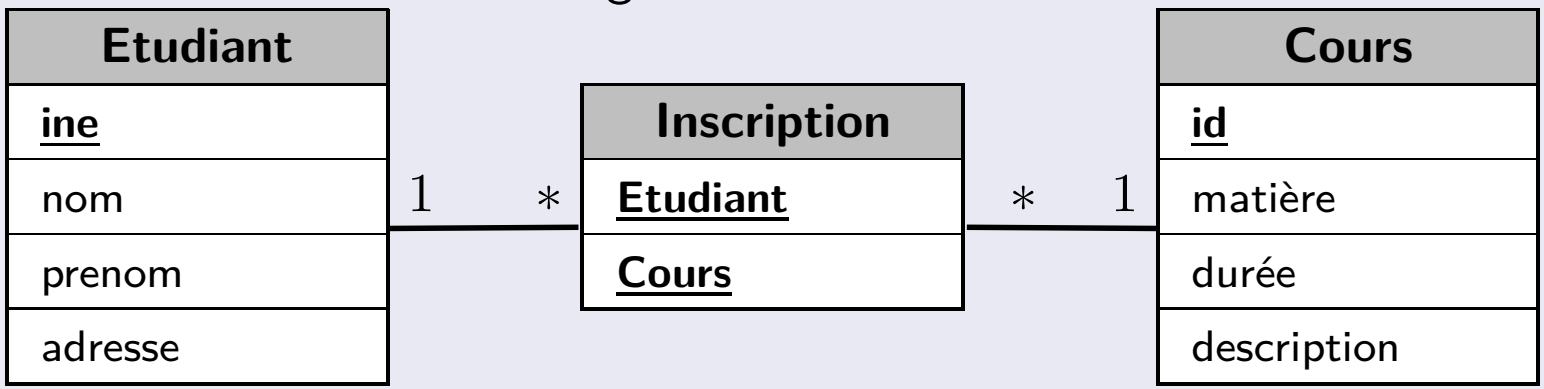
Cas des associations *many to many*

On crée trois tables : une pour chacune des entités et la table de liaison, celle-ci a pour clé primaire l'union des clés primaires des deux entités et est en liaison avec celles-ci en utilisant des clés étrangères.



Cas des associations *many to many*

On crée trois tables : une pour chacune des entités et la table de liaison, celle-ci a pour clé primaire l'union des clés primaires des deux entités et est en liaison avec celles-ci en utilisant des clés étrangères.



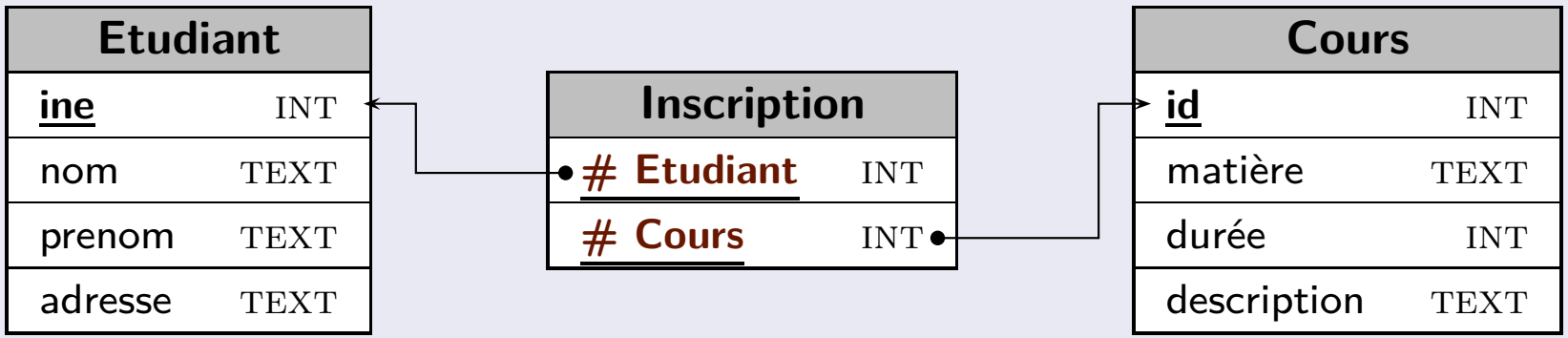


Schéma relationnel

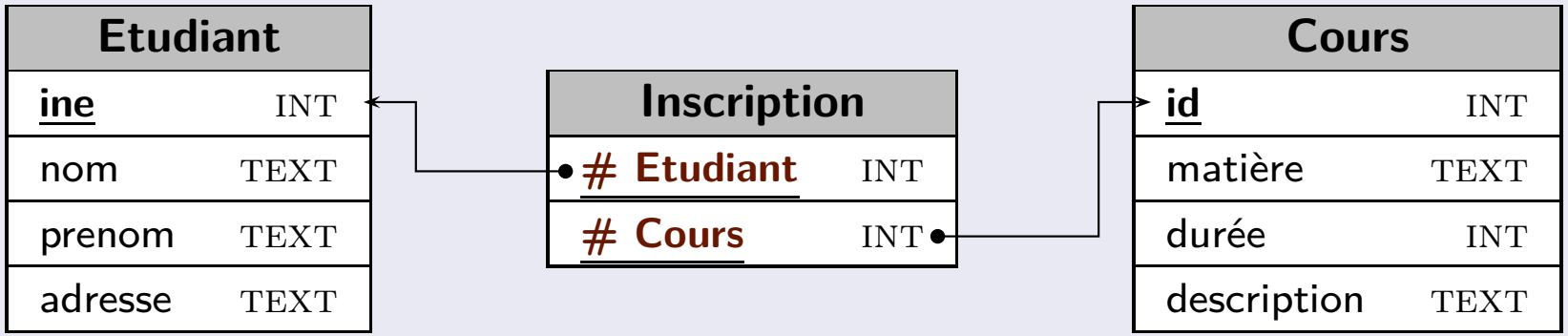
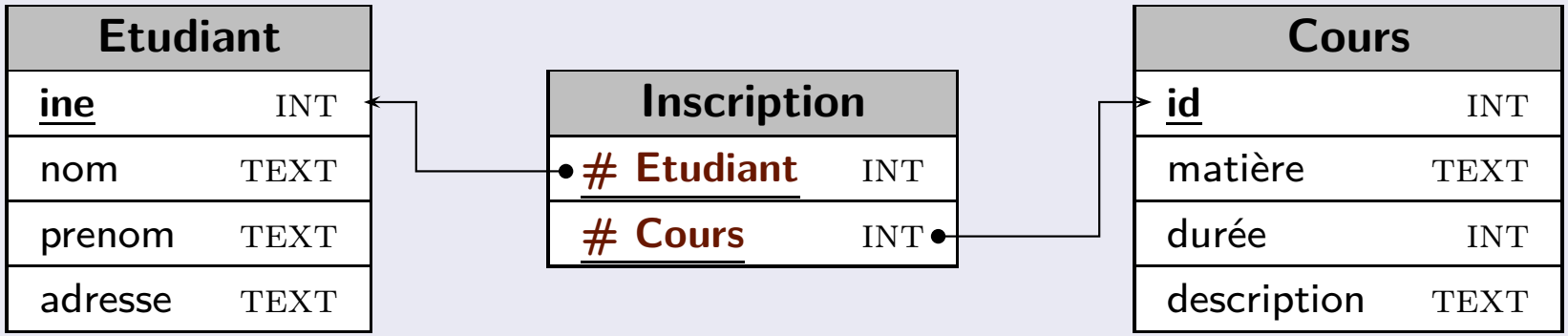


Schéma relationnel



On obtient alors le schéma relationnel suivant :

Etudiant (ine, nom, prenom, adresse)
Cours (id, matière, durée, description)
Inscription (# Etudiant, # cours)

Exemple

Exemple

On souhaite créer une base de données permettant de gérer les notes obtenus par des élèves dans des matières.

Exemple

On souhaite créer une base de données permettant de gérer les notes obtenus par des élèves dans des matières.

- Les élèves ont les attributs suivants : nom, prénom, date de naissance, et identifiant unique.

Exemple

On souhaite créer une base de données permettant de gérer les notes obtenus par des élèves dans des matières.

- Les élèves ont les attributs suivants : nom, prénom, date de naissance, et identifiant unique.
- Les matières ont les attributs suivants : nom (unique), horaire, coefficient

Exemple

On souhaite créer une base de données permettant de gérer les notes obtenus par des élèves dans des matières.

- Les élèves ont les attributs suivants : nom, prénom, date de naissance, et identifiant unique.
- Les matières ont les attributs suivants : nom (unique), horaire, coefficient
- Chaque élève peut avoir plusieurs notes par matière.

Exemple

On souhaite créer une base de données permettant de gérer les notes obtenus par des élèves dans des matières.

- Les élèves ont les attributs suivants : nom, prénom, date de naissance, et identifiant unique.
- Les matières ont les attributs suivants : nom (unique), horaire, coefficient
- Chaque élève peut avoir plusieurs notes par matière.

- Expliquer pourquoi un schéma relationnel d'une seule table notes n'est pas satisfaisant.

Exemple

On souhaite créer une base de données permettant de gérer les notes obtenus par des élèves dans des matières.

- Les élèves ont les attributs suivants : nom, prénom, date de naissance, et identifiant unique.
- Les matières ont les attributs suivants : nom (unique), horaire, coefficient
- Chaque élève peut avoir plusieurs notes par matière.

- Expliquer pourquoi un schéma relationnel d'une seule table notes n'est pas satisfaisant.

Exemple

On souhaite créer une base de données permettant de gérer les notes obtenus par des élèves dans des matières.

- Les élèves ont les attributs suivants : nom, prénom, date de naissance, et identifiant unique.
- Les matières ont les attributs suivants : nom (unique), horaire, coefficient
- Chaque élève peut avoir plusieurs notes par matière.

- Expliquer pourquoi un schéma relationnel d'une seule table notes n'est pas satisfaisant.
- Proposer un schéma relationnel constitué de 3 tables issu du modèle entité-association.

Union de deux tables

Lorsque deux tables T_1 et T_2 ont le même schéma relationnel (c'est à dire les même colonnes), $T_1 \cup T_2$ contient les enregistrement de T_1 ou T_2 (sans duplication). La syntaxe SQL correspondante est :

Exemple

Union de deux tables

Lorsque deux tables T_1 et T_2 ont le même schéma relationnel (c'est à dire les même colonnes), $T_1 \cup T_2$ contient les enregistrement de T_1 ou T_2 (sans duplication). La syntaxe SQL correspondante est :

```
SELECT * FROM T1 UNION SELECT * FROM T2;
```

Exemple

Union de deux tables

Lorsque deux tables T_1 et T_2 ont le même schéma relationnel (c'est à dire les même colonnes), $T_1 \cup T_2$ contient les enregistrement de T_1 ou T_2 (sans duplication). La syntaxe SQL correspondante est :

```
SELECT * FROM T1 UNION SELECT * FROM T2;
```

Exemple

Table T_1

Id	Nom	Prénom
7	Payet	Jean
28	Hoarau	Paul
42	Untel	Sam
57	Casimir	Tom

Table T_2

Id	Nom	Prénom
12	Martin	Pierre
42	Untel	Sam
45	Grondin	Eric

Table $T_1 \cup T_2$

Id	Nom	Prénom
7	Payet	Jean
28	Hoarau	Paul
42	Untel	Sam
57	Casimir	Tom
12	Martin	Pierre
45	Grondin	Eric

Intersection de deux tables

Lorsque deux tables T_1 et T_2 ont le même schéma relationnel (c'est à dire les même colonnes), $T_1 \cap T_2$ contient les enregistrement apparaissant dans T_1 et dans T_2.

Exemple

Intersection de deux tables

Lorsque deux tables T_1 et T_2 ont le même schéma relationnel (c'est à dire les même colonnes), $T_1 \cap T_2$ contient les enregistrement apparaissant dans T_1 et dans T_2. La syntaxe SQL correspondante est :

```
SELECT * FROM T1 INTERSECT SELECT * FROM T2 ;
```

Exemple

Intersection de deux tables

Lorsque deux tables T_1 et T_2 ont le même schéma relationnel (c'est à dire les même colonnes), $T_1 \cap T_2$ contient les enregistrement apparaissant dans T_1 et dans T_2. La syntaxe SQL correspondante est :

```
SELECT * FROM T1 INTERSECT SELECT * FROM T2 ;
```

Exemple

Table T_1

Id	Nom	Prénom
7	Payet	Jean
28	Hoarau	Paul
42	Untel	Sam
57	Casimir	Tom

Table T_2

Id	Nom	Prénom
12	Martin	Pierre
42	Untel	Sam
45	Grondin	Eric

Table $T_1 \cap T_2$

Id	Nom	Prénom
42	Untel	Sam

Différence de deux tables

Lorsque deux tables T_1 et T_2 ont le même schéma relationnel (c'est à dire les même colonnes), $T_1 - T_2$ contient les enregistrement apparaissant dans T_1 et pas dans T_2.

Exemple

Différence de deux tables

Lorsque deux tables T_1 et T_2 ont le même schéma relationnel (c'est à dire les même colonnes), $T_1 - T_2$ contient les enregistrement apparaissant dans T_1 et pas dans T_2. La syntaxe SQL correspondante est :

```
SELECT * FROM T1 EXCEPT SELECT * FROM T2 ;
```

Exemple

Différence de deux tables

Lorsque deux tables T_1 et T_2 ont le même schéma relationnel (c'est à dire les même colonnes), $T_1 - T_2$ contient les enregistrement apparaissant dans T_1 et pas dans T_2. La syntaxe SQL correspondante est :

```
SELECT * FROM T1 EXCEPT SELECT * FROM T2 ;
```

Exemple

Table T_1

Id	Nom	Prénom
7	Payet	Jean
28	Hoarau	Paul
42	Untel	Sam
57	Casimir	Tom

Table T_2

Id	Nom	Prénom
12	Martin	Pierre
42	Untel	Sam
45	Grondin	Eric

Table $T_1 - T_2$

Id	Nom	Prénom
7	Payet	Jean
28	Hoarau	Paul
57	Casimir	Tom

Produit cartésien de deux tables

On peut réaliser le produit cartésien de deux tables, c'est à dire l'ensemble des enregistrements formé d'un enregistrement de la première table et d'un enregistrement de la seconde.

Exemple

Produit cartésien de deux tables

On peut réaliser le produit cartésien de deux tables, c'est à dire l'ensemble des enregistrements formé d'un enregistrement de la première table et d'un enregistrement de la seconde.
La syntaxe SQL correspondante est :

```
SELECT * FROM T1, T2 ;
```

Exemple

Produit cartésien de deux tables

On peut réaliser le produit cartésien de deux tables, c'est à dire l'ensemble des enregistrements formé d'un enregistrement de la première table et d'un enregistrement de la seconde.
La syntaxe SQL correspondante est :

```
SELECT * FROM T1, T2 ;
```

Exemple

Table T_1

Id	Nom	Prénom
7	Payet	Jean
42	Untel	Sam

Table T_2

Matière	Note
Info	15
Maths	9
Physique	10

Table $T_1 \times T_2$

Id	Nom	Prenom	Matière	Note
7	Payet	Jean	Info	15
42	Untel	Sam	Info	15
7	Payet	Jean	Maths	9
42	Untel	Sam	Maths	9
7	Payet	Jean	Physique	10
42	Untel	Sam	Physique	10

Définition

- La jointure de deux tables T_1 et T_2 sur les colonnes A et B revient à combiner les enregistrements de T_1 et T_2 lorsque les colonnes A et B coïncident.

Définition

- La jointure de deux tables T_1 et T_2 sur les colonnes A et B revient à combiner les enregistrements de T_1 et T_2 lorsque les colonnes A et B coïncident.
- Cette jointure se note $T_1 \bowtie_{A=B} T_2$

Définition

- La jointure de deux tables T_1 et T_2 sur les colonnes A et B revient à combiner les enregistrements de T_1 et T_2 lorsque les colonnes A et B coïncident.
- Cette jointure se note $T_1 \bowtie_{A=B} T_2$
- Cette notion est à celle de clé étrangère, on effectuera souvent les jointures avec A une clé primaire et B une clé étrangère correspondante.

Définition

- La jointure de deux tables T_1 et T_2 sur les colonnes A et B revient à combiner les enregistrements de T_1 et T_2 lorsque les colonnes A et B coïncident.
- Cette jointure se note $T_1 \bowtie_{A=B} T_2$
- Cette notion est à celle de clé étrangère, on effectuera souvent les jointures avec A une clé primaire et B une clé étrangère correspondante.
- La syntaxe SQL correspondante est :

```
SELECT * FROM T1 JOIN T2 on T1.A = T2.B
```

Définition

- La jointure de deux tables T_1 et T_2 sur les colonnes A et B revient à combiner les enregistrements de T_1 et T_2 lorsque les colonnes A et B coïncident.
- Cette jointure se note $T_1 \bowtie_{A=B} T_2$
- Cette notion est à celle de clé étrangère, on effectuera souvent les jointures avec A une clé primaire et B une clé étrangère correspondante.
- La syntaxe SQL correspondante est :
```
SELECT * FROM T1 JOIN T2 on T1.A = T2.B
```
- On peut joindre plus de deux tables.

Exemple

Exemple

Table Auteurs

Id	Prénom	Nom
1	Isaac	Asimov
4	Franck	Herbert
7	Jules	Verne

Exemple

Table Auteurs

Id	Prénom	Nom
1	Isaac	Asimov
4	Franck	Herbert
7	Jules	Verne

Table Livres

Num	Auteur	Titre
1	4	Dune
2	1	Les robots
3	7	L'île mystérieuse
4	1	Fondation

Exemple

Table Auteurs

Id	Prénom	Nom
1	Isaac	Asimov
4	Franck	Herbert
7	Jules	Verne

Table Livres

Num	Auteur	Titre
1	4	Dune
2	1	Les robots
3	7	L'île mystérieuse
4	1	Fondation

La jointure de Auteurs et Livres sur les colonnes Id et Auteur donne :

Exemple

Table Auteurs

Id	Prénom	Nom
1	Isaac	Asimov
4	Franck	Herbert
7	Jules	Verne

Table Livres

Num	Auteur	Titre
1	4	Dune
2	1	Les robots
3	7	L'île mystérieuse
4	1	Fondation

La jointure de Auteurs et Livres sur les colonnes Id et Auteur donne :

Id	Prénom	Nom	Num	Auteur	Titre
1	Isaac	Asimov	2	1	Les robots
1	Isaac	Asimov	4	1	Fondation
4	Franck	Herbert	1	4	Dune
7	Jules	Verne	3	7	L'île mystérieuse

Exemple

Table Auteurs

Id	Prénom	Nom
1	Isaac	Asimov
4	Franck	Herbert
7	Jules	Verne

Table Livres

Num	Auteur	Titre
1	4	Dune
2	1	Les robots
3	7	L'île mystérieuse
4	1	Fondation

La jointure de Auteurs et Livres sur les colonnes Id et Auteur donne :

Id	Prénom	Nom	Num	Auteur	Titre
1	Isaac	Asimov	2	1	Les robots
1	Isaac	Asimov	4	1	Fondation
4	Franck	Herbert	1	4	Dune
7	Jules	Verne	3	7	L'île mystérieuse

```
SELECT * FROM Auteurs JOIN Livres on Auteurs.ID = Livres.Auteur
```

Exemple

Table Auteurs

Id	Prénom	Nom
1	Isaac	Asimov
4	Franck	Herbert
7	Jules	Verne

Table Livres

Num	Auteur	Titre
1	4	Dune
2	1	Les robots
3	7	L'île mystérieuse
4	1	Fondation

La jointure de Auteurs et Livres sur les colonnes Id et Auteur donne :

Id	Prénom	Nom	Num	Auteur	Titre
1	Isaac	Asimov	2	1	Les robots
1	Isaac	Asimov	4	1	Fondation
4	Franck	Herbert	1	4	Dune
7	Jules	Verne	3	7	L'île mystérieuse

```
SELECT * FROM Auteurs JOIN Livres on Auteurs.ID = Livres.Auteur
```

(On préfixe le nom des attributs par celui de la table afin d'éviter toute ambiguïté.)

Requête dans le résultat d'une requête

- Le résultat d'une requête peut-être utilisé afin d'effectuer une autre requête.

Exemple

Requête dans le résultat d'une requête

- Le résultat d'une requête peut-être utilisé afin d'effectuer une autre requête.
- C'est le principe des requêtes imbriquées.

Exemple

Requête dans le résultat d'une requête

- Le résultat d'une requête peut-être utilisé afin d'effectuer une autre requête.
- C'est le principe des requêtes imbriquées.
- C'est souvent dans un `WHERE` ou un `HAVING`

Exemple

Requête dans le résultat d'une requête

- Le résultat d'une requête peut-être utilisé afin d'effectuer une autre requête.
- C'est le principe des requêtes imbriquées.
- C'est souvent dans un `WHERE` ou un `HAVING`

Exemple

Dans la relation *Objet* (Référence : INT, description : TEXT, prix : FLOAT), comment retrouver le (ou les) objets ayant le prix le plus élevé ?

Requête dans le résultat d'une requête

- Le résultat d'une requête peut-être utilisé afin d'effectuer une autre requête.
- C'est le principe des requêtes imbriquées.
- C'est souvent dans un `WHERE` ou un `HAVING`

Exemple

Dans la relation *Objet* (Référence : INT, description : TEXT, prix : FLOAT), comment retrouver le (ou les) objets ayant le prix le plus élevé ?

- `SELECT Référence, MAX(prix) FROM objet;`
  Cette solution n'est pas satisfaisante car elle renverra une seule référence même si plusieurs objets ont le prix maximal.

Requête dans le résultat d'une requête

- Le résultat d'une requête peut-être utilisé afin d'effectuer une autre requête.
- C'est le principe des requêtes imbriquées.
- C'est souvent dans un `WHERE` ou un `HAVING`

Exemple

Dans la relation *Objet* (Référence : INT, description : TEXT, prix : FLOAT), comment retrouver le (ou les) objets ayant le prix le plus élevé ?

- `SELECT Référence, MAX(prix) FROM objet;`
  Cette solution n'est pas satisfaisante car elle renverra une seule référence même si plusieurs objets ont le prix maximal.
- `SELECT Référence, prix FROM objet ORDER BY prix DESC LIMIT 1;`
  De même pour cette solution !

Requête dans le résultat d'une requête

- Le résultat d'une requête peut-être utilisé afin d'effectuer une autre requête.
- C'est le principe des requêtes imbriquées.
- C'est souvent dans un `WHERE` ou un `HAVING`

Exemple

Dans la relation *Objet* (Référence : INT, description : TEXT, prix : FLOAT), comment retrouver le (ou les) objets ayant le prix le plus élevé ?

- `SELECT Référence, MAX(prix) FROM objet;`
  Cette solution n'est pas satisfaisante car elle renverra une seule référence même si plusieurs objets ont le prix maximal.
- `SELECT Référence, prix FROM objet ORDER BY prix DESC LIMIT 1;`
  De même pour cette solution !
- ```
  SELECT Référence, prix FROM Objet
  WHERE note = (SELECT MAX(prix) FROM objet);
  ```
  ```