

Base de données

Chapitre 1 : Formalisation des bases de données relationnelles

Joel Cavat

2021

Définition

Une base de données est une collection d'informations persistantes organisée de manière à pouvoir être facilement manipulée.

Elle est définie par

- des entités,
- la nature des liens entre ses entités,
- des contraintes.

Base de données relationnelle

Base de données relationnelle

Une base de données relationnelle permet de regrouper des données structurées dans un ensemble de **tables** organisées en **lignes** et **colonnes**. Elle permet de renforcer les contraintes d'intégrité.

Formalisation

- Réalisé par Edgar F. Codd en 1970
 - algèbre relationnelle
 - basé sur des fondations mathématiques
 - théorie des ensembles
 - concept de relations
 - logique du premier ordre
- Objectif
 - comment décrire formellement une base de données

Utilité

Représenter de manière non-ambiguë un moyen de décrire les données, leur structure et les opérations sans considération informatique.

Bénéficier de l'apport des mathématiques (pour démontrer, prouver, ...)

Permet une totale indépendance entre les **programmes** (impératifs et procéduraux) et l'**organisation des données** (déclarative/descriptive).

Thèmes abordés

- **formalisme du modèle relationnel** (présent chapitre, description des données, de leur structure et des contraintes)
- algèbre relationnelle (description des opérations)
- normalisation (éviter la redondance et les incohérences)

Ensemble

Un ensemble est une collection d'éléments. Chaque élément est unique.

Produit cartésien

Le produit cartésien de deux ensembles X et Y , noté $X \times Y$, est l'ensemble des couples (x, y) tel que $x \in X, y \in Y$.

- noté également $\{(x, y) \mid x \in X, y \in Y\}$

Produit cartésien

Exemple

$P = \{\text{Paul}, \text{Joël}, \text{Orestis}\}$

$N = \{\text{Malaspinas}, \text{Cavat}\}$

$P \times N = \{(\text{Paul}, \text{Malaspinas}), (\text{Paul}, \text{Cavat}),$
 $(\text{Joël}, \text{Malaspinas}), (\text{Joël}, \text{Cavat}),$
 $(\text{Orestis}, \text{Malaspinas}), (\text{Orestis}, \text{Cavat})\}$

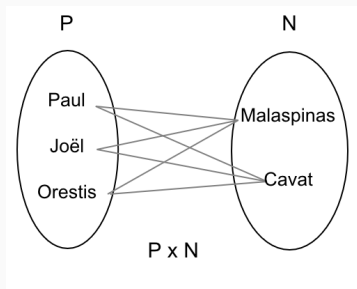


Figure 1: représ. ensembliste

$P \times N$	
Prénom (P)	Nom (N)
Paul	Malaspinas
Paul	Cavat
Joël	Malaspinas
Joël	Cavat
Orestis	Malaspinas
Orestis	Cavat

Figure 2: représ. tabulaire

Schéma d'une relation (~table)

Schéma d'une relation

Le schéma d'une relation R dénoté $R(A_1, A_2, \dots, A_n)$ est composé d'un nom R et d'attributs A_1, A_2, \dots, A_n .

Une **table** est le nom commun d'un schéma d'une relation.

Exemples

- Visiteur(id_visiteur, prenom, nom, email)
- Conference(id_conf, intitule, date_debut, date_fin, prix)
- Hotel(id_hotel, nom_hotel, adresse)

Relation

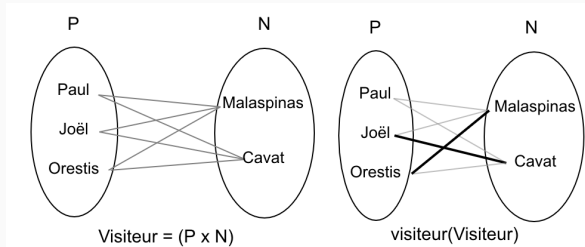
Une relation r d'un schéma R , noté $r(R)$, est un sous-ensemble du produit cartésien défini par $R : r(R) \subseteq A_1 \times A_2 \times \dots \times A_n$.

Une relation est un ensemble de tuples (*n-uplets*)
 $(a_1, a_2, \dots, a_n), a_i \in A_i$

Un **tuple** représente un enregistrement (une ligne) d'une table alors d'une **relation** représente l'ensemble des enregistrements d'une table.

Relation

Exemple



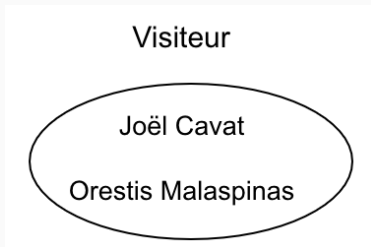
P x N

Prénom (P)	Nom (N)
Paul	Malaspinas
Paul	Cavat
Joël	Malaspinas
Joël	Cavat
Orestis	Malaspinas
Orestis	Cavat

schéma: Visiteur(prenom, nom)
relation: visiteur(Visiteur) \subseteq P x N

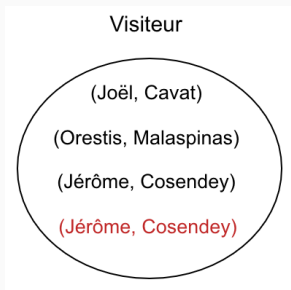
Prénom (P)	Nom (N)
Paul	Malaspinas
Paul	Cavat
Joël	Malaspinas
Joël	Cavat
Orestis	Malaspinas
Orestis	Cavat

Une relation est également un ensemble (par définition)



Relation et tuple

Par définition, un **ensemble** est une collection d'éléments uniques. Il n'existe donc **pas deux tuples de mêmes valeurs** dans une relation.



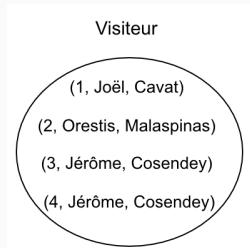
Clé primaire

La clé primaire d'un schéma est un sous-ensemble d'attributs du schéma de la relation permettant d'identifier de manière unique un tuple.

- Deux tuples distincts d'une relation ne peuvent avoir la même clé (contrainte d'unicité).
- Une clé est obligatoirement renseignée
- Il peut exister plusieurs clés candidates
- Une clé artificielle peut être créée
 - si aucune clé n'est candidate
 - *(si les clés candidates sont mal adaptées pour l'indexation)*

Exemples

- `Visiteur(id_visiteur, prenom, nom)`



La clé primaire décrit une contrainte forte a elle seule. Dans l'exemple précédent elle précise:

- connaissant un identifiant, celui-ci ne peut correspondre qu'à un prénom et un nom

Clé étrangère

Une clé étrangère est un attribut (ou ensemble d'attributs) référançant une clé primaire d'une autre table

Remarques

- la valeur doit exister dans l'autre relation (contrainte d'intégrité référentielle ou **dépendance d'inclusion**)

Utilité

- permet d'éviter les redondances (et anomalies)
- indique une cardinalité (correspond à **un**)

redondance et cardinalité

id_conf	nom	id_theme	intitulé
1	ScalaDays 2018	1	MicroService
2	LambdaDays 2019	2	Prog. fonctionnelle
3	Curry On 2019	3	Système formel
4	ScalaDays 2020	2	Prog. fonctionnelle

Figure 3: relation comportant des redondances

id_conf	nom	id_theme	id_theme	intitulé
1	ScalaDays 2018	1	1	MicroService
2	LambdaDays 2019	2	2	Prog. fonctionnelle
3	Curry On 2019	3	3	Système formel
4	ScalaDays 2020	2		

Figure 4: séparation + clé étrangère

Formalisation

Theme(id_theme, intitule)

Conference(id_conference, nom, id_theme)

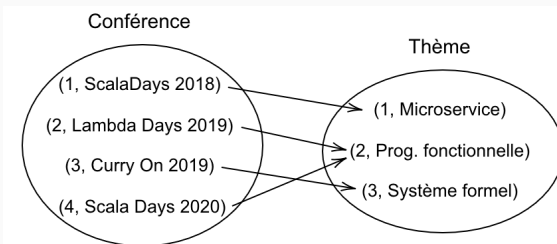
$\text{id_theme} \subseteq \text{Theme.id_theme}$

Il est utile de se représenter cette “liaison” à l’aide d’un diagramme sagital et de flèches

Représentons les associations à l'aide d'arcs (flèches). Le sens a une importance, il décrit un lien de déduction entre l'origine (conférence) et son extrémité (thème).

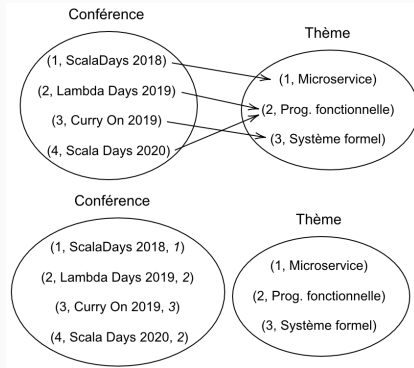
Interprétation de l'illustration

- connaissant une conférence, je peux connaître son thème ; une conférence n'a donc qu'un seul thème (card. max de 1)
- étant donné qu'une conférence n'a qu'un thème, chaque conférence devient l'**origine** d'un arc.

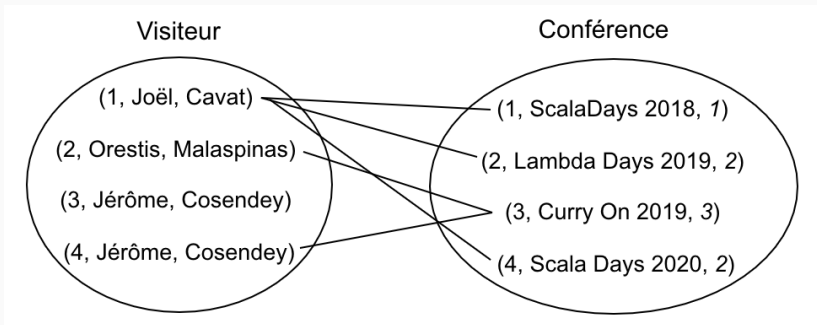


Clé étrangère

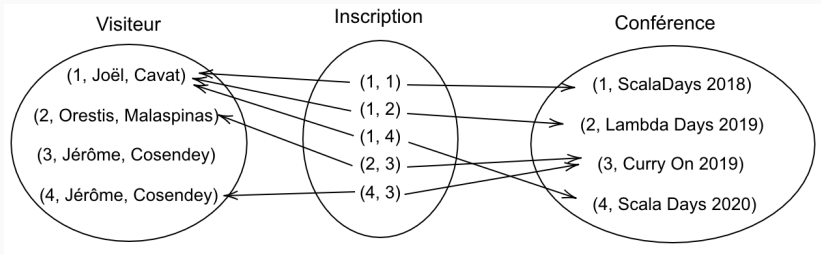
Dans l'univers relationnel, les arcs n'existent pas. En les supprimant, nous comprenons l'utilité de la clé étrangère : elle permet de représenter une association. Elle apparaît du côté de l'origine de l'arc.



Association plusieurs-à-plusieurs ???



Clé étrangère



Fonction

Une fonction d'un ensemble A vers un ensemble B est une relation (un sous-ensemble $A \times B$) où chaque élément de A est unique (appartient à un et un seul couple de la relation.)

Dépendance fonctionnelle

Dépendance fonctionnelle

Les dépendances fonctionnelles sont des contraintes qui décrivent les liens qui peuvent exister entre des attributs.

Dépendance fonctionnelle

Une DF, dénotée $X \rightarrow Y$, entre deux ensembles d'attributs X et Y d'une relation R , spécifie une contrainte entre ces deux ensembles.

$X \rightarrow Y$ peut être interprété de différentes manières:

- connaissant X je peux déduire Y
- connaissant X il ne peut y avoir qu'un Y
- X détermine Y

Remarques:

- X est appelé le **déterminant**
- Y est appelé le **déterminé**

Exemple 1:

1. $\{\text{id_theme}\} \rightarrow \{\text{intitule}\}$
2. $\{\text{id_conference}\} \rightarrow \{\text{nom}, \text{id_theme}\}$

Signification:

1. connaissant l'identifiant d'un thème, il est possible de connaître son intitulé. Sémantiquement, correspond à un thème.
2. l'identifiant d'une conférence nous permet de connaître le nom de la conférence et l'identifiant de son thème associé. Sémantiquement, correspond à une conférence.

Exemple 2:

1. $\{\text{no_isbn}\} \rightarrow \{\text{titre}, \text{description}\}$
2. $\{\text{no_exemplaire}\} \rightarrow \{\text{no_isbn}, \text{date_acquisition}, \text{prix}\}$
3. $\{\text{no_exemplaire}, \text{no_emprunteur}, \text{date_emprunt}\} \rightarrow \{\text{date_retour}, \text{date_echeance}\}$

Quiz: écrivez leur signification et trouvez une sémantique

- 1.
- 2.
- 3.

Utilité

En plus de permettre de modéliser des contraintes supplémentaires, elles permettent de déterminer les identifiants.

Remarques

Les attributs déterminants sont généralement une **clé primaire** ou **uniques**.

Exemple 1

Le schéma `Personne(no_avs, nom, prenom, date_naissance)` et sa DF $\{no_avs\} \rightarrow \{nom, prenom, date_naissance\}$ notée:

```
Personne(id_avs, nom, prenom, date_naissance)
DF1: {no_avs} → {nom, prenom, date_naissance}
```

se simplifie ainsi:

```
Personne(id_avs, nom, prenom, date_naissance)
```

Le choix de la clé décrit une DF.

Exemple 2

Le schéma `Client(no_client, nom, prenom, no_tel)` et ses DF:

- $\{no_client\} \rightarrow \{nom, prenom\}$
- $\{no_tel\} \rightarrow \{nom, prenom\}$

se simplifie ainsi:

```
Client(id_client, nom, prenom, no_tel)
```

ou de préférence (préférez une seule clé primaire):

```
Client(id_client, nom, prenom, no_tel)  
no_tel UNIQUE
```

Quiz

Quelle serait la limitation de cette DF ? est-elle plus restrictive ou plus permissive ?

- $\{\text{no_exemplaire}, \text{no_emprunteur}\} \rightarrow \{\text{date_emprunt}, \text{date_retour}, \text{date_echeance}\}$

par rapport à celle-ci:

- $\{\text{no_exemplaire}, \text{no_emprunteur}, \text{date_emprunt}\} \rightarrow \{\text{date_retour}, \text{date_echeance}\}$

Schéma de base de données relationnelle

Le schéma d'une base de données relationnelle est composé de l'ensemble des schémas des relations et l'ensemble des dépendances fonctionnelles.

Schéma de base de données relationnelle

Modèle EA	Modèle relationnel (courant)	Modèle relationnel (formel)
Attribut	Attribut	Attribut
Identifiant	Clé primaire	Clé primaire
Type d'entité	Table	Schéma de relation
Entité	Enregistrement	Tuple
-	Ensemble des enregistrements	Relation
Association	<i>représ. par clé étr.</i>	<i>représ. par une CI réf.</i>

Dépendance d'inclusion

Nous avons vu qu'une clé étrangère d'un schéma de relations doit référencer la clé primaire d'un autre schéma de relations.

La valeur de la clé étrangère doit donc exister dans l'autre relation.

La notation que nous avons employé correspond en fait à une **dépendance d'inclusion (DI)**.

```
Theme(id_theme, intitule)
```

```
Conference(id_conference, nom, id_theme)
```

```
id_theme  $\subseteq$  Theme.id_theme
```

De manière générale:

- Une **DI** indique que les valeurs d'un ou de plusieurs attributs d'un enregistrement doivent exister dans une autre relation
 - La **clé étrangère** est une **DI**
 - Une DI n'est pas nécessairement une clé étrangère
- Permet de renforcer (tout comme les DF) les contraintes d'intégrité
- Se modélise facilement

Imaginons une plateforme en ligne de type Amazon qui met en relation des fournisseurs et des clients.

La page suivante propose deux schémas de relations qui permettent une telle modélisation.

DI: cas d'utilisation

- `Livraison(...)`: ceci permet de savoir les produits livrés par un fournisseur. Un même produit peut être livré par plusieurs fournisseurs différents.
- `Achat(...)`: un client peut acheter un produit auprès du fournisseur de son choix.

extrait du schéma de base de données:

```
Achat(id_achat, id_fourn, id_client, id_produit, date)
```

```
id_fourn  $\subseteq$  Fournisseur.id_fournisseur
```

```
id_client  $\subseteq$  Client.id_client
```

```
id_produit  $\subseteq$  Produit.id_produit
```

```
Livraison(id_fournisseur, id_produit)
```

```
id_produit  $\subseteq$  Produit.id_produit
```

```
id_fournisseur  $\subseteq$  Fournisseur.id_fournisseur
```

DI: cas d'utilisation

Achat(id_achat, id_fourn, id_client, id_produit, date), ref: ...
Livraison(id_fournisseur, id_produit), ref: ...

livraison(Livraison)

id_fourn	id_produit
F1	P1
F1	P2
F2	P1
F3	P4

achat(Achat)

id_achat	id_fourn	id_produit	id_client	date
A1	F2	P1	C2	d1
A2	F1	P4	C1	d2

Remarques

- F1 ne livre pourtant pas le produit P4

DI: cas d'utilisation

Achat(id_achat, id_fourn, id_client, id_produit, date)

(id_fourn, id_produit) \subseteq Livraison.(id_fournisseur, id_produit)

id_client \subseteq Client.id_client

Livraison(id_fournisseur, id_produit)

id_produit \subseteq Produit.id_produit

id_fournisseur \subseteq Fournisseur.id_fournisseur

livraison(Livraison)

id_fourn	id_produit
F1	P1
F1	P2
F2	P1
F3	P4

achat(Achat)

id_achat	id_fourn	id_produit	id_client	date
A1	F2	P1	C2	d1
A2	F1	P4	C1	d2

DI: cas d'utilisation

Première version (sans contrainte)

Achat(id_achat, id_fourn, id_client, id_produit, date)

id_fourn \subseteq Fournisseur.id_fournisseur

id_client \subseteq Client.id_client

id_produit \subseteq Produit.id_produit

Livraison(id_fournisseur, id_produit)

id_produit \subseteq Produit.id_produit

id_fournisseur \subseteq Fournisseur.id_fournisseur

Seconde version (améliorée)

Achat(id_achat, id_fourn, id_client, id_produit, date)

(id_fourn, id_produit) \subseteq Livraison.(id_fournisseur, id_produit)

id_client \subseteq Client.id_client

Livraison(id_fournisseur, id_produit)

id_produit \subseteq Produit.id_produit

id_fournisseur \subseteq Fournisseur.id_fournisseur