Introduction aux Bases de Données

De la gestion des fichiers aux bases de données
Partie 1/2

Anis SAIED

IPEIN

2025/26

anis.saied@ipein.ucar.tn

Introduction

Le but de ce chapitre est

- Assimiler les fondements des Bases de Données Relationnelles (BDR)
- Avantages des BD par rapport aux fichiers ?

Contexte

Considérons une application de gestion de bibliothèque qui utilise des fichiers texte pour stocker les informations des **livres**, des **étudiants** et leurs **emprunts**.

Problématique

Comment Modéliser le système d'informations

On commence par se poser les questions suivantes:
- Comment modéliser ce problème ? comment passer du monde réel au monde virtuel ? Quelles informations une bibliothèque (simplifiée) doit-elle gérer ?

- Des **Livres**
- Des etudiants : **Emprunteurs** ;
- Des emprunts de livres par des emprunteurs : **Emprunts** ;

→On doit avoir des informations sur ces trois types d'objets/concepts :

- Livre : numéro, titre, auteur, année,. . .
- Emprunteur : nom, prénom, de quoi l'identifier, . . .
- Emprunt : qui a emprunté quoi, quand ? Quand le document doit-il être rendu ? . . .

Comment mettre en oeuvre le système d'information de la bibliothèque ?

→Solution disponible : **Fichiers texte**

Les Fichiers texte I

Exemple

Livres.txt :

```
ISBN, Titre, Auteur, Année
978-3-16-148410-0, Python 3, A. Jamel, 2020
978-0-13-110362-7, Intro BD, B. Med., 2018
```

Etudiants.txt :

```
ID, Nom, Prénom, Adresse
001, B.A., Ali, 12 Rue Liberté
002, D.S., Bacem, 46 Avenue Tunis
```

Emprunts.txt :

```
ID Emprunteur, ISBN Livre, Titre Livre, Date Emprunt, Date Retour
001, 978-3-16-148410-0, Python 3, 2024-08-01, 2024-08-15
002, 978-0-13-110362-7, Intro BD, 2024-08-05, 2024-08-19
```

Ce model de données textuel présente plusieurs inconvénients !

Limites des fichiers texte

1. Redondance des données

Les mêmes informations sont **dupliquées** dans plusieurs fichiers ce qui entraîne : Du gaspillage d'espace; Un risque élevé d'erreurs lors des mises à jour; Une difficulté à garantir l'unicité des données...

Exemple : Si le titre ou le ISBN d'un livre change, quels fichiers doivent être **mis à jour** ? sachant que toute omission entraîne une incohérence.

Livres.txt :

```
ISBN, Titre, Auteur, Année
978-3-16-148410-0, Python 3, A. Jamel, 2020
978-0-13-110362-7, Intro BD, B. Med., 2018
```

Emprunts.txt :

```
ID Emprunteur, ISBN Livre, Titre Livre, Date Emprunt
001, 978-3-16-148410-0, Python 3, 2024-08-01, 2024-08-15
002, 978-0-13-110362-7, Intro BD, 2024-08-05, 2024-08-19
```

→ Si le titre ou le ISBN du livre "Python 3" est modifié, il faut le mettre à jour dans *Livres.txt* et *Emprunts.txt*.

Limites des fichiers texte

2. Incohérence des données

Une incohérence apparaît lorsque des fichiers censés contenir la même information ne sont plus synchronisés.

Exemple : La suppression d'un livre dans `Livres.txt` n'est pas suivie automatiquement (par erreur humaine ou technique) par la suppression des emprunts associés dans `Emprunts.txt`.

Livres.txt :

```
ISBN, Titre, Auteur, Année
978-3-16-148410-0, Python 3, A. Jamel, 2020(supprimé)
978-0-13-110362-7, Intro BD, B. Med., 2018
```

Emprunts.txt :

```
ID Emprunteur, ISBN Livre, Titre Livre, Date Emprunt
001, 978-3-16-148410-0, Python 3, 2024-08-01, 2024-08-15
002, 978-0-13-110362-7, Intro BD, 2024-08-05, 2024-08-19
```

→ Le livre "Python 3" a été supprimé de *Livres.txt*, mais il est toujours présent dans *Emprunts.txt*, créant ainsi une incohérence.

Limites des fichiers texte

3. Sécurité des données

Il est difficile d'affiner le contrôle d'accès aux données spécifiques dans chaque fichier et de sécuriser les données sensibles.

Exemple :

Comment empêcher un utilisateur non autorisé d'accéder ou de modifier (ajouter, modifier ou supprimer) une ligne du fichier `Emprunts.txt` ?

Livres.txt :

```
ISBN, Titre, Auteur, Année
978-3-16-148410-0, Python 3, A. Jamel, 2020
978-0-13-110362-7, Intro BD, B. Med., 2018
```

Emprunts.txt :

```
ID Emprunteur, ISBN Livre, Titre Livre, Date Emprunt, Date Retour
001, 978-3-16-148410-0, Python 3, 2024-08-01, 2024-08-15
002, 978-0-13-110362-7, Intro BD, 2024-08-05, 2024-08-19
```

→ Si le fichier *Emprunts.txt* est stocké en clair et accessible à tous les utilisateurs, des personnes non autorisées pourraient y accéder ou le modifier, compromettant ainsi la **confidentialité** et l'**intégrité des données**.

→ Il est difficile de mettre en place des **contrôles d'accès** *granulaires* pour chaque fichier individuel.

Limites des fichiers texte

4. Couplage fort entre le contenu des fichiers texte et le code source

Le **contenu** des fichiers texte est étroitement **lié au code source**, ce qui rend les **modifications difficiles** et **sensible aux erreurs**.

Exemple :

Modifier le format de *Emprunts.txt* nécessite des changements dans le code qui lit ce fichier.

Format initial de *Emprunts.txt* :

```
ID Emprunteur, ISBN Livre, Titre Livre, Date Emprunt, Date Retour
001, 978-3-16-148410-0, Python 3, 2024-08-01, 2024-08-15
```

Format modifié de *Emprunts.txt* :

```
ID Emprunteur, ISBN, État Livre, Titre, Date Emprunt, Date Retour
001, 978-3-16-148410-0, Bon état, Python 3, 2024-08-01,
```

→ Ajouter un champ **État Livre** pour indiquer l'état du livre nécessite une mise à jour du code source pour gérer ce nouveau champ.

→ Sans ajustement du code, le traitement des données dans le fichier peut être incorrect ou échouer.

Limites des fichiers texte

5. Recherche inefficace

La **recherche d'informations** spécifiques ou le **tri des données** devient rapidement **complexe** et **lent**.

Exemple :
Comment trouver tous les livres empruntés par un même emprunteur sans parcourir manuellement tous les enregistrements ?

- → Trouver tous les livres empruntés par un emprunteur nécessite de parcourir le fichier *Emprunts.txt* en entier (parcours, strip, split, conversion de types, comparaisons, faire attention aux index ...).
- → La recherche et le tri des données deviennent longs et peu pratiques avec des fichiers texte volumineux.

Autres limites des fichiers texte

De nombreuses autres limites apparaissent selon les contextes d'utilisation :

- Absence de contrôle d'intégrité *(ex. impossibilité d'imposer l'unicité d'un ID)* ;
- Difficulté à gérer les accès concurrents *(ex. deux utilisateurs modifient simultanément le même fichier)* ;
- Absence de transactions *(ex. une mise à jour échoue au milieu et laisse les données incohérentes)* ;
- Risques élevés de perte des données *(ex. suppression accidentelle d'un fichier)* ;
- Faible capacité d'évolution du modèle *(ex. ajout d'un champ nécessitant de modifier tous les fichiers)*.

Conclusion : Ces limites justifient la nécessité d'un **modèle de données structuré**, comme celui proposé par les Bases de Données Relationnelles (BDR).

Solution fiable

Base de données

Fichier texte

Modèle textuel

BDR

Modèle relationnel

Du fichier à la base de données I

Modèle Textuel → Modèle Relationnels

Données en Fichiers Texte

Les données enregistrées dans des fichiers texte sont souvent organisées en lignes, chaque ligne représentant un enregistrement avec des valeurs séparées par des délimiteurs.

Exemple :

Etudiants.txt :

```
ID, Nom, Prénom, Adresse
001, B.A., Ali, 12 Rue Liberté
002, D.S., Bacem, 46 Avenue Tunis
```

Du fichier à la base de données II

Modèle Textuel → Modèle Relationnels

Modélisation en Relations

En base de données relationnelle (BDR), les mêmes données peuvent être modélisées sous forme de Relations : **ensemble** d'enregistrements.

Chaque **enregistrement** est représenté comme un ensemble de couples (paires clé-valeur), où chaque **clé** est un **attribut** et chaque **valeur** est la **donnée** associée.

Exemple : **Relation Etudiants**

	Clé	:	**Valeur**
Enregistrement	ID	:	001
	Nom	:	B.A.
	Prénom	:	Ali
	Adresse	:	12 Rue Liberté
Enregistrement	ID	:	002
	Nom	:	D.S.
	Prénom	:	Bacem
	Adresse	:	46 Avenue Tunis

Du fichier à la base de données III

Modèle Textuel → Modèle Relationnels

Lignes de Texte

```
001, B.A. , Ali, 12 Rue Liberté
002, D.S. , Bacem, 46 Avenue Tunis
```

Transformation

Couples Clé-Valeur

```
{ID:001 , Nom:B.A , Prénom:Ali , Adresse:12 Rue Liberté}
{ID:002 , Nom:D.S , Prénom:Bacem , Adresse:46 Avenue Tunis}
```

Conversion

Relation / Table Relationnelle

ID	Nom	Prénom	Adresse
001	B.A.	Ali	12 Rue Liberté
002	D.S.	Bacem	46 Avenue Tunis

Modèle Relationnel

Relation

Une **Relation** est une table représentant un ensemble de tuples.

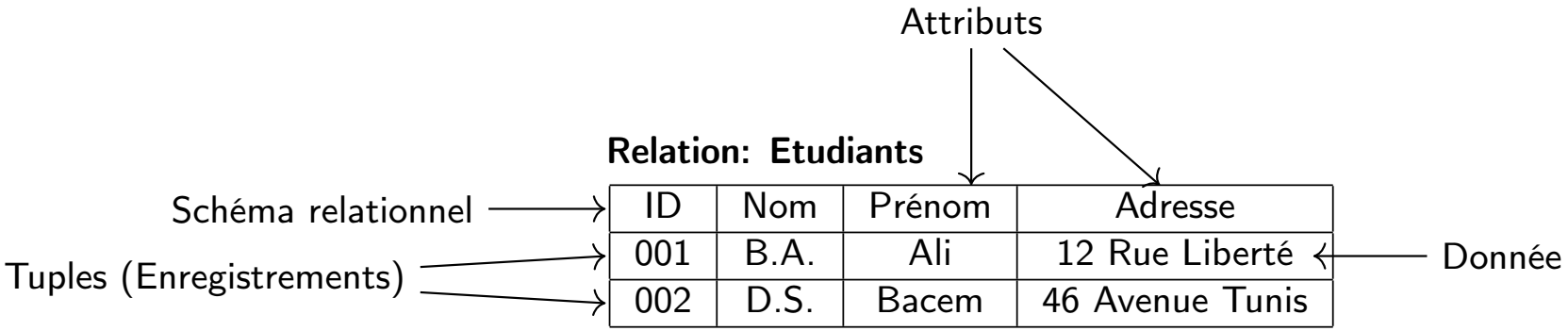


Schéma relationnel S : Ensemble d'attributs décrivant les données d'une relation, noté $S = (A_1, \ldots, A_n)$, avec $i \neq j \Rightarrow A_i \neq A_j$.

Pour un **attribut** A donné, $D(A)$ est le **domaine** (type) de A: c'est l'ensemble des *valeurs* qui peuvent être prises par l'attribut A.
Ex: $D(\text{ID}) = \text{entier}$
Exemples courants : Entier, Réel, chaine, booléen, Date, ...

On note aussi $S = ((A_1, D(A_1)), \ldots, (A_n, D(A_n))$
Exemple : Etudiants (ID, Nom, Prénom, Adresse)

Modèle Relationnel

Définition

Le modèle relationnel est un modèle de données où les informations sont organisées sous forme de **relations** (ou tables).

Chaque relation est constituée de lignes (**tuples**) et de colonnes (**attributs**).

Exemple :

Une table "Etudiants" avec des colonnes comme ID, Nom, Prénom, et Adresse.

Une base de données organisée selon un modèle relationnel est dite **base de données relationnelle (BDR)** et est gérée à l'aide d'un **SGBD Relationnel (SGBDR)**.

Modèle Relationnel

Clés

Clé Primaire :

Une **clé primaire** est un attribut (ou une combinaison d'attributs) dont le contenu est unique pour chaque enregistrement, permettant d'identifier un et un seul enregistrement dans une table.

Exemple :

Dans la table "Etudiants", la colonne "ID" peut être la clé primaire car chaque étudiant a un ID unique.

Notation : Etudiant(**ID**, Nom, Prenom, Adresse)

Clé Étrangère :

Une **clé étrangère** est un attribut d'une table qui fait référence à la clé primaire d'une autre table, établissant un lien entre les deux tables.

Exemple :

Dans la table "Emprunts", la colonne "ID_Emprunteur" est une clé étrangère qui fait référence à l'ID de la table "Etudiants".

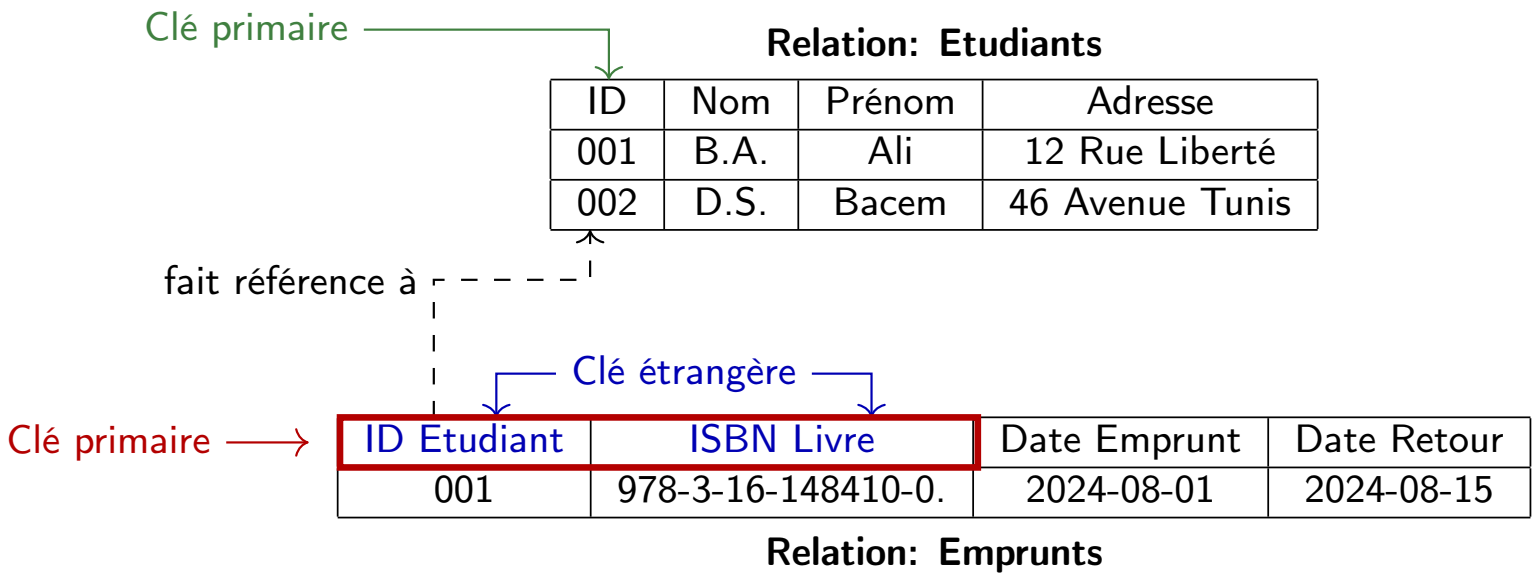
Notation : Emprunts(#**ID_Emprunteur**, ...)

Modèle Relationnel

Intégrité Référentielle

L'**intégrité référentielle** garantit que toutes les clés étrangères pointent vers des enregistrements existants dans la table référencée, assurant ainsi la cohérence des données.
Une « *violation d'intégrité référentielle* » se produit lorsque cette condition n'est pas respectée.
Exemple :

Si un enregistrement est supprimé dans "Etudiants", toutes les emprunts associées dans "Emprunts" doivent être supprimées ou mises à jour (*NULL* à la place des ID supprimés).



BDR

Manipulation de données

Une base de données organisée selon un modèle de données de type relationnel, est dite **base de données relationnelle** (BDR).

Manipulation Théorique :

Les opérations sur les bases de données relationnelles sont définies formellement par l'**algèbre relationnelle**.

Manipulation Pratique :

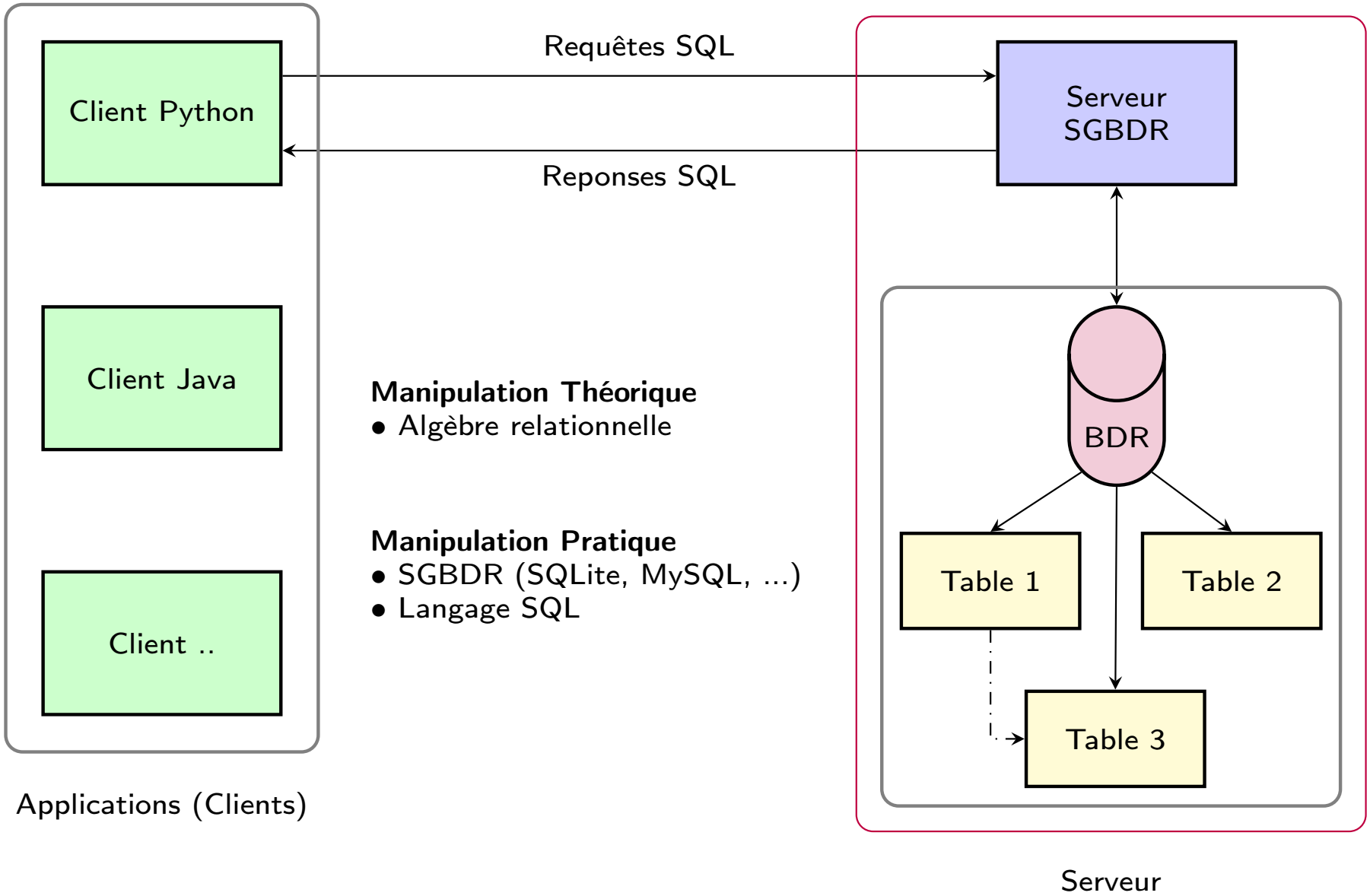
En pratique, les bases de données sont manipulées par les **Systèmes de Gestion de Bases de Données (SGBDR)** à l'aide du langage **SQL** (Structured Query Language).

Ces systèmes (ex: **SQLite**, MySQL, PostgreSQL, ...) permettent de gérer les tables, d'ajouter, modifier, et supprimer des données, et de garantir l'intégrité des données.

Vérification des acquis

1. Quel est le principal problème des fichiers texte ?
   A. Ils sont difficiles à lire
   B. Ils entraînent de la redondance
   C. Ils ne peuvent pas être ouverts sous Windows
2. Dans une table, un tuple correspond à :
   A. Une colonne
   B. Une ligne
   C. Une base de données
3. Une clé étrangère sert à :
   A. Garantir l'unicité
   B. Relier deux tables
   C. Accélérer les recherches

Architecture Client-Serveur pour une Base de Données Relationnelle (BDR)



Manipulation Théorique
- Algèbre relationnelle

Manipulation Pratique
- SGBDR (SQLite, MySQL, ...)
- Langage SQL

anis.saied@ipein.ucar.tn

Algèbre Relationnelle

Définition

L'**algèbre relationnelle** est un ensemble d'opérations formelles qui permet de manipuler et interroger les relations (tables).

Elle se base sur l'utilisation d'opérateurs pour manipuler les relations (tables) afin de produire de nouvelles relations comme résultat.

Syntaxe générale d'une opération :

```
R = Opérateur Relation
R = Relation1 Opérateur Relation2
```

Opérateurs: Les symboles comme $\cup$, $\cap$, σ, π, $\bowtie$, etc., sont utilisés pour représenter des opérations en algèbre relationnelle.

Exemple :

$R3 = R1 \cup R2$

Union (R3 = Tous les éléments de R1 et R2, sans répétition)

Les opérateurs de l'algèbre relationnelle sont classés en deux catégories : **unaires** et **binaires**.

Opérateurs de l'AR

Opérateurs Unaires

Les opérateurs unaires sont appliqués sur une seule relation :

Sélection (σ) : Filtre les lignes d'une relation qui satisfont un prédicat spécifique.
Exemple : $\sigma_{\text{condition}}(R)$

Projection (π) : Sélectionne un sous-ensemble de colonnes d'une relation.
Exemple : $\pi_{\text{colonnes}}(R)$

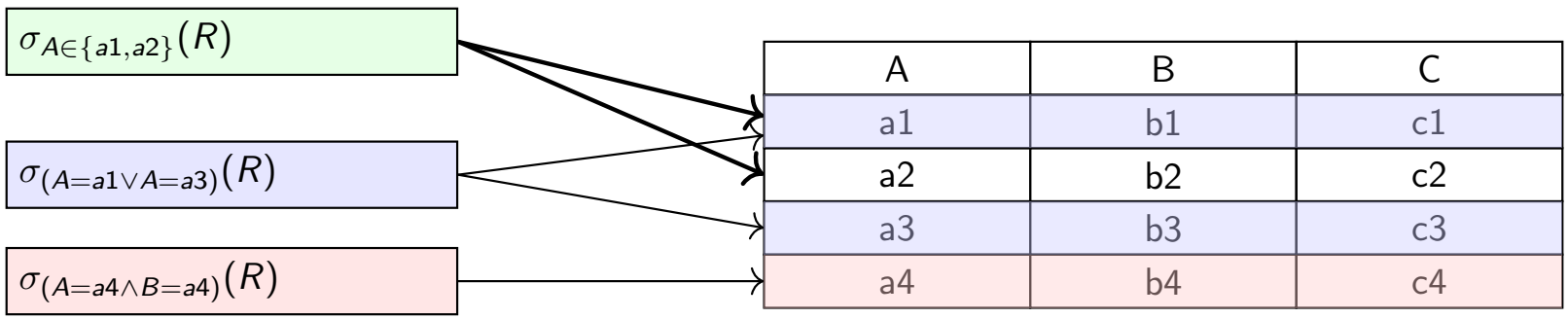
Renommage (ρ) : Renomme les attributs d'une relation.
Exemple : $\rho_{B \to A}(R)$

Opérateur de Sélection (σ)

L'opérateur de sélection (σ) est utilisé pour filtrer les lignes d'une relation qui satisfont un prédicat spécifique. Cet opérateur est unaire et s'applique donc à une seule relation.

Syntaxe: $$R = \sigma_{\text{condition}}(\text{Relation})$$

Où `condition` est une expression booléenne qui filtre les tuples de la relation, et dans laquelle on peut utiliser les opérateurs logiques suivants : **AND** ($\wedge$), **OR** ($\vee$), **NOT** ($\neg$), **IN** ($\in$), et **LIKE**



Exemple : Sélectionner les emprunts où la **Date de Retour** est le 2024-08-15.

$$R = \sigma_{\text{Date Retour}=\text{'2024-08-15'}}(\text{Emprunts})$$

R :

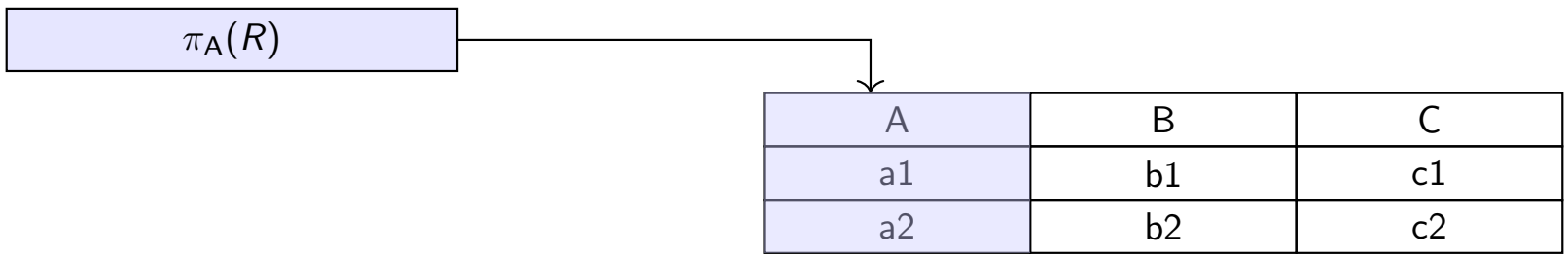
ID Emprunteur	ISBN Livre	Titre Livre	Date Emprunt	Date Retour
001	978-3-...	Python 3	2024-08-01	2024-08-15

Opérateur de Projection (π)

L'opérateur de projection (π) est utilisé pour sélectionner des colonnes spécifiques d'une relation. Cet opérateur est aussi unaire et s'applique donc à une seule relation.

Syntaxe: $$R = \pi_{\text{colonnes}}(\text{Relation})$$

Où `colonnes` est une liste des colonnes que vous souhaitez conserver dans le résultat.



Exemple : Projeter les colonnes **ID Emprunteur** et **Date Retour** d'une table d'emprunts. $$R = \pi_{\text{ID Emprunteur,Date Retour}}(\text{Emprunts})$$

R :

ID Emprunteur	**Date Retour**
001	2024-08-15
002	2024-08-19

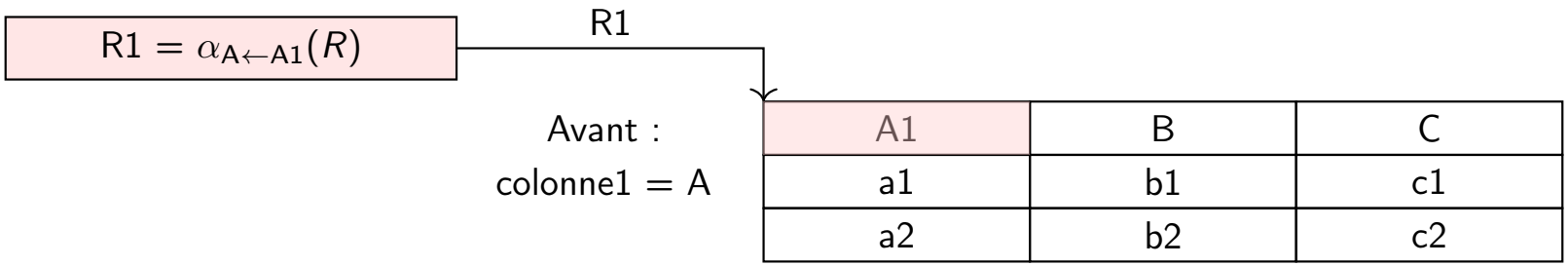
Renommage (α)

Le renommage est un opérateur unaire qui produit une relation R' ayant les mêmes tuples/enregistrements et le même schémaque la relation R de départ, mais dans lequel un attribut a été renommé.

But : résoudre des problèmes de compatibilité entre noms d'attributs de deux relations opérandes d'une opération binaire.

Notation :

$$R' = \alpha_{\text{nomAttribut} \leftarrow \text{nouveauNom}}(R)$$



Exercice

Application simple: Opérateurs unaires

Question

Afficher l'écrivan du livre "Pyhon 3".

Relation: Livres

ISBN	Titre	Auteur	Année
978-3-16-148410-0	Python 3	A. Jamel	2020
978-0-13-110362-7	Intro BD	B. Med.	2018

Réponse

$R1 = \sigma_{Titre='Python3'}(Livres)$
$R2 = \pi_{Auteur}(R1)$
$R3 = \alpha_{Auteur \leftarrow Ecrivan}(R2)$

Relation: R3

Ecrivan
A. Jamel

Opérateurs Binaires

Les opérateurs binaires sont appliqués sur deux relations. Voici les principaux :

Union ($\cup$) : Combine les lignes de deux relations, sans les dupliquer.
Exemple : $R \cup S$

Intersection ($\cap$) : Sélectionne les lignes communes entre deux relations.
Exemple : $R \cap S$

Différence ($-$) : Sélectionne les lignes qui sont dans la première relation mais pas dans la deuxième.
Exemple : $R - S$

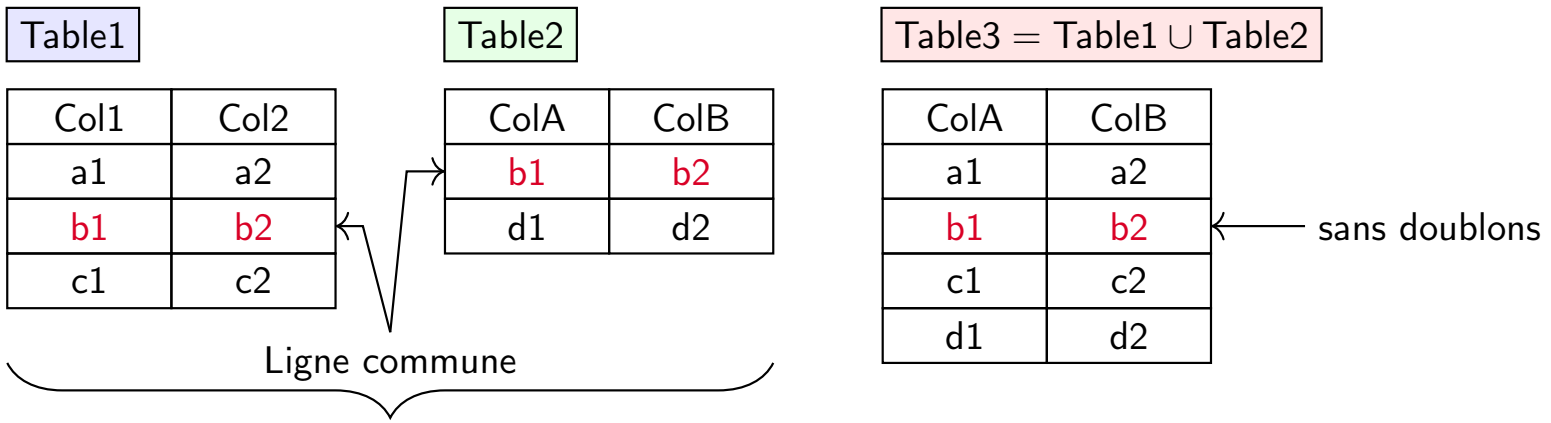
Produit Cartésien ($\times$) : Combine chaque ligne de la première relation avec chaque ligne de la seconde relation.
Exemple : $R \times S$

Jointure ($\bowtie$) : Combine les lignes de deux relations en fonction d'une condition de jointure.
Exemple : $R \bowtie_{\text{condition}} S$

Opérateur d'Union ($\cup$)

L'opérateur d'union ($\cup$) est utilisé pour combiner les lignes de deux relations ayant le même schéma. Cet opérateur est également binaire et s'applique donc à deux relations.Les doublons sont automatiquement éliminés.

Syntaxe : $R = \text{Relation1} \cup \text{Relation2}$



Même schéma: même nombre de colonnes et mêmes types

Exemple: Les noms de tous les étudiants.

$R3 = \pi_{Nom}(R1) \cup \pi_{Nom}(R2)$

R1: Étudiants 1ère année

Nom	ID Étudiant
Ahmed	E1
Fatima	E2
Sami	E3

R2: Étudiants 2ème année

Nom	ID Étudiant
Ahmed	E4
Hala	E5
Sami	E6

R3: Liste de tous les étudiants

Nom
Ahmed
Fatima
Sami
Hala

Opérateur d'Intersection ($\cap$)

L'opérateur d'intersection ($\cap$) est utilisé pour trouver les lignes communes entre deux relations ayant le même schéma.

Syntaxe : $R = \text{Relation1} \cap \text{Relation2}$

Table1

Col1	Col2
a1	a2
b1	b2
c1	c2

Table2

ColA	ColB
b1	b2
d1	d2

Ligne commune

Table3 = Table1 $\cap$ Table2

ColA	ColB
b1	b2

Même schéma: même nombre de colonnes et mêmes types

Exemple: Les noms communs des étudiants dans les deux années.

$R3 = \pi_{Nom}(R1) \cap \pi_{Nom}(R2)$

R1: Étudiants 1ère année

ID	Nom
E1	Ali
E2	Ahmed
E3	Hela

R2: Étudiants 2ème année

ID	Nom
E2	Ali
E4	Sami
E5	Chaima

R3: les nom des étudiants communs

Nom
Ali

Opérateur de Différence ($-$)

L'opérateur de différence ($\setminus$ ou $-$) est utilisé pour trouver les lignes présentes dans la première relation mais pas dans la deuxième relation ayant le même schéma.

Syntaxe : $R = \text{Relation1} - \text{Relation2}$

Table1

Col1	Col2
a1	a2
b1	b2
c1	c2

Table2

ColA	ColB
b1	b2
d1	d2

Ligne commune

Table3 = Table1 − Table2

ColA	ColB
a1	a2
c1	c2

Même schéma: même nombre de colonnes et mêmes types

Exemple: Les ID des produits non commandés.

$R3 = \pi_{ID}(R1) - \pi_{IdProd}(R2)$

R1: Produits

ID	Nom
P1	Orange
P2	Pomme
P3	Banane

R2: Commandes

IdProd	Quantité
P1	2.5
P3	3

R3: ID des Produits non commandés

ID
P2

Opérateur de Produit Cartésien ($\times$)

L'opérateur de produit cartésien ($\times$) est utilisé pour combiner chaque ligne de la première relation avec chaque ligne de la deuxième relation. Le résultat est une nouvelle relation contenant toutes les combinaisons possibles des lignes des deux relations.

Syntaxe : $R = \text{Relation1} \times \text{Relation2}$

Table1

Col1	Col2
a1	a2
b1	b2
c1	c2

Table2

ColA	ColB
x1	x2
y1	y2

Table3 = Table1 $\times$ Table2

Col1	Col2	ColA	ColB
a1	a2	x1	x2
a1	a2	y1	y2
b1	b2	x1	x2
b1	b2	y1	y2
c1	c2	x1	x2
c1	c2	y1	y2

anis.saied@ipein.ucar.tn

Produit Cartésien ($\times$)

Exemple

Imaginons que nous ayons deux tables : une pour les clients et une autre pour les produits. Nous voulons obtenir toutes les combinaisons possibles de clients et de produits pour générer des commandes potentielles.

Clients

ClientID	Nom
C1	Ali
C2	Bacem

Produits

ProduitID	Produit
P1	Livre
P2	Stylo

Commandes = Clients $\times$ Produits

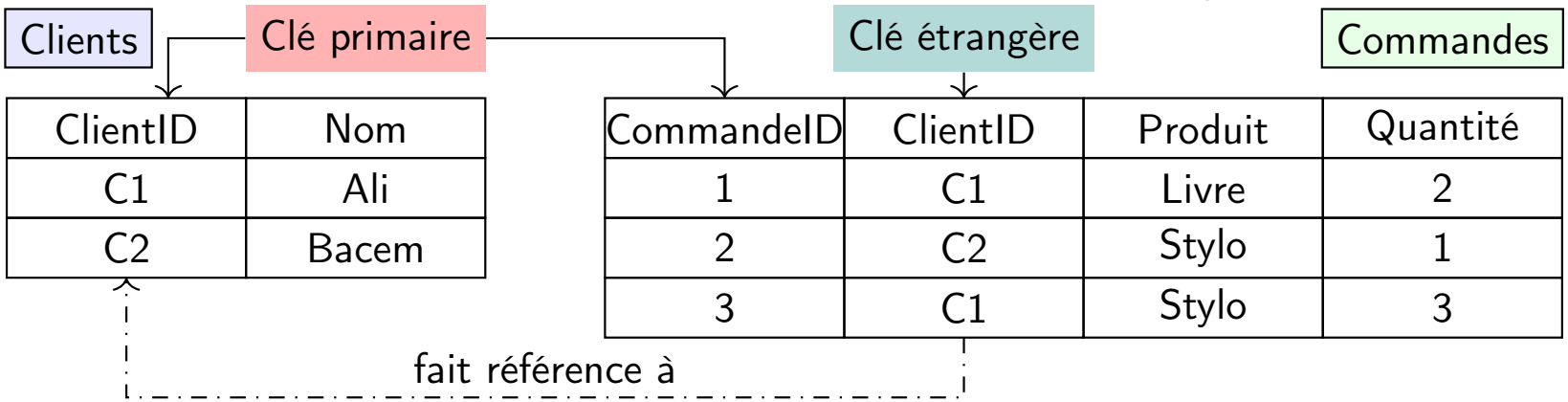
ClientID	Nom	ProduitID	Produit
C1	Ali	P1	Livre
C1	Ali	P2	Stylo
C2	Bacem	P1	Livre
C2	Bacem	P2	Stylo

Résultat : Cette table montre toutes les commandes potentielles où chaque client est associé à chaque produit.

Jointure ($\bowtie$)

$R3 = R1 \underset{\text{condition}}{\bowtie} R2$

La jointure est utilisée pour combiner des lignes de deux ou plusieurs tables en fonction d'une colonne commune entre elles. Elle permet de récupérer des données de plusieurs tables et de les associer de manière significative.



ClientID	Nom
C1	Ali
C2	Bacem

CommandeID	ClientID	Produit	Quantité
1	C1	Livre	2
2	C2	Stylo	1
3	C1	Stylo	3

Exemple : Associer chaque commande à son client.

Principe : Il s'agit de combiner chaque ligne de la table 'Commandes' avec une ligne de la table 'Clients' où le 'ClientID' correspond.

$$\textit{Clients} \underset{\text{Clients.ClientID} = \text{Commandes.ClientID}}{\bowtie} \textit{Commandes}$$

1. On commence par réaliser le produit cartésien entre les tables 'Clients' et 'Commandes'.
2. Ensuite, on sélectionne les lignes où les 'ClientID' correspondent pour obtenir la jointure finale.

Jointure

Étape 1 : Produit Cartésien

Le produit cartésien combine chaque ligne de la table ‘Clients‘ avec chaque ligne de la table ‘Commandes‘, produisant ainsi une table intermédiaire où chaque combinaison possible de lignes est présente.

Produit Cartésien: Clients × Commandes

ClientID	Nom	CommandeID	ClientID	Produit	Quantité
C1	Ali	1	**C1**	Livre	2
C1	Ali	2	C2	Stylo	1
C1	Ali	3	**C1**	Stylo	3
C2	Bacem	1	C1	Livre	2
C2	Bacem	2	**C2**	Stylo	1
C2	Bacem	3	C1	Stylo	3

Le tableau ci-dessus représente le produit cartésien des tables ‘Clients‘ et ‘Commandes‘, combinant chaque ligne de ‘Clients‘ avec chaque ligne de ‘Commandes‘.

Jointure

Étape 2 : Sélection

La sélection de la jointure consiste à filtrer le produit cartésien pour ne conserver que les lignes où les ‘ClientID‘ correspondent (les valeurs des clés étrangères égales aux valeurs de clés primaires).

Selection : $\sigma_{Clients.ClientID=Commandes.ClientID}(Clients \times Commandes)$

ClientID	Nom	CommandeID	ClientID	Produit	Quantité
C1	Ali	1	C1	Livre	2
C1	Ali	2	C1	Stylo	1
C2	Bacem	3	C2	Stylo	3

Le tableau ci-dessus représente la table résultante après avoir appliqué la condition de jointure ‘Clients.ClientID = Commandes.ClientID‘.
Cette étape réduit le produit cartésien à la jointure correcte des deux tables, affichant les colonnes ‘ClientID‘ deux fois.

Jointure

Clients $\underset{\text{Clients.ClientID} = \text{Commandes.ClientID}}{\bowtie}$ *Commandes*

Table Clients

ClientID	Nom
C1	Ali
C2	Bacem

Table Commandes

CommandeID	ClientID	Produit	Quantité
1	C1	Livre	2
2	C2	Stylo	1
3	C1	Stylo	3

Table intermédiare: Produit Cartésien

ClientID	Nom	CommandeID	ClientID	Produit	Quantité
C1	Ali	1	C1	Livre	2
C1	Ali	2	C2	Stylo	1
C1	Ali	3	C1	Stylo	3
C2	Bacem	1	C1	Livre	2
C2	Bacem	2	C2	Stylo	1
C2	Bacem	2	C1	Stylo	3

Table : Résultat de la jointure

ClientID	Nom	CommandeID	ClientID	Produit	Quantité
C1	Ali	1	C1	Livre	2
C1	Ali	3	C1	Stylo	3
C2	Bacem	2	C2	Stylo	1

anis.saied@ipein.ucar.tn

Jointure

Exercice

Q. Afficher le nom du client de la commande N° 1.

R.

$$R1 = \text{Clients} \underset{\text{Clients.ClientID} = \text{Commandes.ClientID}}{\bowtie} \text{Commandes}$$

ou bien

$$R1 = \text{Clients} \bowtie \text{Commandes}$$

$$R2 = \sigma_{\text{CommandeID}=1}(R1)$$

$$R3 = \pi_{\text{Nom}}(R2)$$

Autrement:

$$\pi_{\text{Nom}} \left(\text{Clients} \underset{\substack{\text{Clients.ClientID} = \text{Commandes.ClientID} \\ \wedge \text{CommandeID} = 1}}{\bowtie} \text{Commandes} \right)$$

À Retenir

Modèle Relationnel :

Modèle des données basé sur des tables (relations).

Chaque table est composée de lignes (tuples) et de colonnes (attributs).

Clés primaires et étrangères assurent l'intégrité des données.

Algèbre Relationnelle :

Langage formel pour manipuler et interroger des bases de données relationnelles.

Opérations principales : sélection (σ), projection (π), jointure ($\bowtie$), union ($\cup$), intersection ($\cap$), différence ($-$).

Permet de définir des requêtes de manière abstraite avant leur implémentation en SQL.

SQL (Structured Query Language) :

Langage standard pour la gestion des bases de données relationnelles.

Implémente les concepts théoriques de l'algèbre relationnelle pour manipuler les données.

Evaluation — Opérations de l'AR I

1. Que fait l'opérateur de sélection σ ?

(A) Il sélectionne des colonnes

(B) Il filtre des lignes selon un prédicat (condition ou une expression logique)

(C) Il combine deux relations

2. Que retourne l'opérateur de projection π ?

(A) Un sous-ensemble de colonnes

(B) Un sous-ensemble de lignes

(C) Une jointure

3. La projection peut-elle produire des doublons ?

(A) Oui

(B) Non

(C) Seulement si la relation contient une clé primaire

4. Quelle opération combine toutes les lignes de deux relations ?

(A) Union

(B) Produit cartésien

Evaluation — Opérations de l'AR II

(C) Jointure naturelle

5. La condition nécessaire pour faire une union :

(A) Même nombre de lignes

(B) Même schéma

(C) Aucune condition

6. L'opération $R - S$ retourne :

(A) Les tuples communs à R et S

(B) Les tuples de R uniquement

(C) Les tuples de S uniquement

7. Une jointure équivaut souvent à :

(A) Un produit cartésien + une sélection

(B) Une union + une différence

(C) Une projection + une intersection

8. Quelle opération peut augmenter très fortement le nombre de tuples ?

(A) Projection

Evaluation — Opérations de l'AR III

(B) Produit cartésien

(C) Différence

9. Quelle opération garde uniquement les tuples présents dans les deux relations ?

(A) Intersection

(B) Union

(C) Jointure extérieure

10. Dans $R \bowtie S$, le symbole $\bowtie$ correspond à :

(A) Une sélection

(B) Une jointure

(C) Une projection

Références utiles

Livres :

- R. Elmasri , S. Navathe — Fundamentals of Database Systems
- C. Date — Introduction to Database Systems
- Ramakrishnan , Gehrke — Database Management Systems

Sites web :

- https://en.wikipedia.org/wiki/Relational_model
- https://sqlbolt.com
- https://w3schools.com/sql
- https://db-engines.com