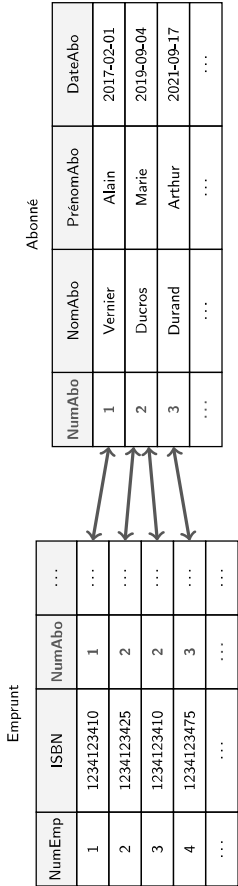


Requêtes multi-relations : jointures

- Requêtes utilisant plusieurs tables :
 - Clause FROM : permet de spécifier les tables sources de données
 - Clause WHERE : jointure sur les attributs liés

```
SELECT *  
FROM Abonné, Emprunt  
WHERE Abonné.NumAbo = Emprunt.NumAbo;
```



- La sortie est une table contenant le produit conditionnel des relations
- Les attributs de la sortie sont l'union des attributs.

← Emprunt

→ Abonné

NumEmp	ISBN	NumAbo	DateEmp	DateRet	NumAbo	NomAbo	PrénomAbo	DateAbo
1	1234123410	1	2017-03-01	2017-04-02	1	Vernier	Alain	2017-02-01
2	1234123425	2	2019-12-12	2020-01-05	2	Ducros	Marie	2019-09-04
3	1234123410	2	2021-10-15	2021-11-17	2	Ducros	Marie	2019-09-04
4	1234123475	3	2021-11-22	NULL	3	Durand	Arthur	2021-09-17
...

- Équi-jointure entre les deux tables (clé primaire et clé étrangère).

Notation	Représente
*	tous les attributs de toutes les tables
Table.Attribut	l'attribut Attribut de la table Table
Table.*	tous les attributs de la table Table

- La notation Table.Attribut est nécessaire pour faire référence à Attribut s'il est présent dans plusieurs tables de la requête.
- C'est vrai pour toutes les clauses de la requête (SELECT, WHERE, etc.).
- Il n'est pas possible de spécifier « tous les attributs sauf certains », il faut alors tous les énumérer.

- Sans jointure, le résultat est le produit cartésien des tables.
- En général, ce n'est pas le résultat attendu !

```
SELECT *  
FROM Emprunt, Abonné;
```

←

Emprunt

→

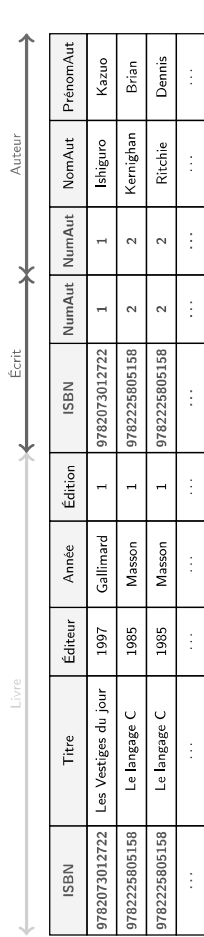
Abonné

NumEmp	ISBN	NumAbo	DateEmp	DateRet	NumAbo	NomAbo	PrénomAbo	DateAbo
1	1234123410	1	2017-03-01	2017-04-02	1	Vernier	Alain	2017-02-01
1	1234123410	1	2017-03-01	2017-04-02	2	Ducros	Marie	2019-09-04
1	1234123410	1	2017-03-01	2017-04-02	3	Durand	Arthur	2021-09-17
...
1	1234123410	1	2017-03-01	2017-04-02	148	Bernard	Françoise	2021-11-20
2	1234123425	2	2019-12-12	2020-01-05	1	Vernier	Alain	2017-02-01
2	1234123425	2	2019-12-12	2020-01-05	2	Ducros	Marie	2019-09-04
2	1234123425	2	2019-12-12	2020-01-05	3	Durand	Arthur	2021-09-17
...

Jointures multiples

- Si la requête utilise N tables, il faut N-1 jointures.
- Exemple.

```
SELECT *
FROM Livre, Auteur
WHERE Livre.ISBN = Ecrit.ISBN
AND Ecrit.NumAut = Auteur.NumAut;
```



Équi-Jointure

- La jointure supprime les n-uplets n'apparaissant pas dans l'une des tables.

- Exemple :

```
SELECT Abonné.*  
FROM Abonné, Emprunt  
WHERE Abonné.NumAbo = Emprunt.NumAbo;
```

- Les abonnés dont le numéro n'apparaît pas dans Emprunt sont supprimés du résultat.
- Résultat : liste des abonnés qui ont déjà effectué (au moins) un emprunt.
- Chaque abonné apparaîtra dans le résultat autant de fois qu'il a fait d'emprunts.

Inéqui-jointures

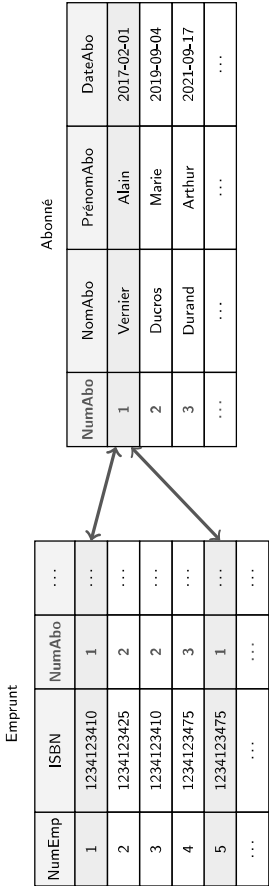
- Une jointure avec un opérateur arithmétique autre que = est appelée inéqui-jointure.
- Exemple :
 - Liste des abonnés ayant emprunté un livre à une date autre que celle de leur inscription

```
SELECT Abonné.*
FROM Abonné, Emprunt
WHERE Abonné.NumAbo = Emprunt.NumAbo
AND Abonné.DateAbo <> Emprunt.DateEmp;
```
 - Si un abonné n'a fait des emprunts que lors de sa journée d'inscription, il n'apparaîtra pas dans le résultat.

Jointure et sélection : clause WHERE

- Exemple : sélection des abonnés de nom « Vernier »

```
SELECT * FROM Abonné, Emprunt
WHERE Abonné.NumAbo = Emprunt.NumAbo
AND Abonné.NomAbo = 'Vernier';
```



Copies multiples d'une table

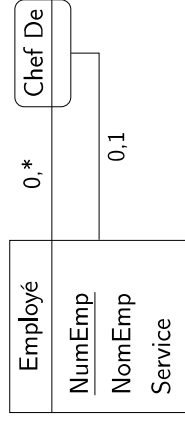
- Certaines requêtes nécessitent plusieurs fois la même table : on crée des alias pour la table.
- Exemple : paires d'abonnés qui se sont inscrits à la même date

```
SELECT A1.NumAbo, A1.NomAbo, A2.NumAbo,
       A2.NomAbo, A2.DateAbo
FROM Abonné A1, Abonné A2
WHERE A1.DateAbo = A2.DateAbo
AND A1.NumAbo < A2.NumAbo;
```
- Tout se passe comme si on avait deux tables identiques indépendantes.
- La condition `A1.NumAbo < A2.NumAbo` permet d'éviter d'associer un abonné avec lui-même (car présent dans les deux tables).

A1.NumAbo	A1.NomAbo	A2.NumAbo	A2.NomAbo	A2.DateAbo
19	Khammar	23	Caranta	2017-10-01
73	Emola	91	Deuring	2019-10-04
...

Copies multiples d'une table (2)

- Exemple 2

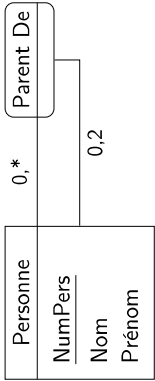


- Le numéro du chef d'un employé est représenté en mettant la clé primaire d'Employé comme clé externe d'Employé :
`EMPLOYÉ(NumEmp, NomEmp, Service, NumChef)`
- Liste des employés avec leur chef :

```
SELECT Employé.*, Chef.*
FROM Employé, Employé Chef
WHERE Employé.NumChef = Chef.NumEmp;
```

Copies multiples d'une table (3)

- Exemple 3



- Les numéros des parents sont représentés comme des clés étrangères dans la relation Personne.
`PERSONNE(NumPers, Nom, Prénom, NumParent1, NumParent2)`
- Liste des personnes avec leurs deux parents :

```
SELECT Personne.*, Parent1.*, Parent2.*
FROM Personne, Personne Parent1, Personne Parent2
WHERE Personne.NumParent1 = Parent1.NumPers
AND Personne.NumParent2 = Parent2.NumPers;
```

- 1 Notations et exemple fil-rouge
- 2 Jointures
- 3 Opérations ensemblistes**
- 4 Exercices
- 5 Requêtes de définition de données
- 6 Sous-requêtes

Union, intersection, différence

- Opérateurs ensemblistes :

```
SELECT ...  
UNION | INTERSECT | EXCEPT  
SELECT ... ;
```

applique l'opération ensembliste aux résultats des deux requêtes.

- UNION : union des résultats des requêtes
- INTERSECT : intersection des résultats
- EXCEPT : différence des résultats.

- Exemple :

```
SELECT NumAbo FROM Emprunt  
WHERE ISBN = '1234123410'  
INTERSECT  
SELECT NumAbo FROM Emprunt  
WHERE ISBN = '1234123475';
```

Exemple 1

- Les deux requêtes doivent être compatibles :
 - même nombre d'attributs
 - et attributs de mêmes domaines.
- Union parfois nécessaire pour avoir tous les n-uplets.

- Exemple :

```
SELECT Employé.*, Chef.NomEmp
FROM Employé, Employé Chef
WHERE Employé.NumChef = Chef.NumEmp
UNION
SELECT Employé.*, NULL
FROM Employé
WHERE NumChef IS NULL;
```

- La jointure de la 1^{ère} requête supprime les employés n'ayant pas de chef.
- La 2^e requête affiche ces employés.
- Résultat : tous les employés sont affichés, avec leur chef s'il existe.

Exemple 2

- Exemple : livres disponibles :
 - « livres qui sont disponibles (qui ne sont pas en cours d'emprunt) »

```
SELECT *  
FROM Livre  
EXCEPT  
SELECT Livre.*  
FROM Emprunt, Livre  
WHERE Livre.ISBN = Emprunt.ISBN  
AND Emprunt.DateRet IS NULL;
```
 - La 1^{ère} requête donne tous les livres (qu'ils apparaissent dans Emprunt ou non).
 - La 2^e requête donne les livres qui sont en cours d'emprunt.
 - Résultat : la différence des deux.