جامعة عبد المالك السعدي
ⵜⴰⵙⴷⴰⵡⵉⵜ ⵏ ⵄⴰⴱⴷⵍⵎⴰⵍⵉⴽ ⵙⵙⴰⵄⴷⵉ
Université Abdelmalek Essaadi

Ecole Nationale des sciences appliquées Al-Hoceima
Université Abdelmalek Essaadi

ENSAH

Administration et optimisation des bases de données

ID1-S2

2021/2022

Mohamed CHERRADI

ORACLE

Plan du cours

1. Rappel
- 1.1. Les dépendances fonctionnelles
- 1.2. Les formes normales
- 1.3. Normalisation et dénormalisation
- 1.3. Les contraintes d'intégrité
- 1.5. Algèbre relationnelle
- 1.5. SQL

2. Administration des bases de données
- 2.1. Gestion des utilisateurs, privilèges, Quottas, profils
- 2.2. L'interface PHPMyAdmin de MySQL
- 2.3. Séquence, Synonym, Transaction, Verrous, Trigger, procédure, fonction, vue, (PL/SQL)
- 2.4. Sauvegarde et restauration de la base de données
- 2.5. Tunning de la base de données
- 2.5. Les outils de monitoring de la base de données
- 2.6. La différence entre l'environnement OLTP et OLAP

3. Optimisation des bases de données
- 3.1. Les index, HINT, Plan d'éxecution logique et physique
- 3.2. La procédure ANALYSE(), EXPLAIN
- 3.3. Les techniques de débuggage et d'analyse d'une requête
- 3.4. Mesure temps d'exécution d'une requête

Volume Horaire (VH)

Elément(s) du module	Volume horaire (VH)					
	Cours	TD	TP	Activités Pratiques	Evaluation	VH global
Administration et Optimisation des Bases de Données	24	10	30	10	6	70
VH global du module	**24**	**10**	**30**	**10**	**6**	**70**
% VH	**34,28**	**14,28**	**42,85**	**14,28**	**8,5**	**100%**

Modes d'évaluation & DIDACTIQUE DU MODULE

- Contrôles continus
- Contrôle TP
- Exposé
- Atelier Data
- Examen

- Contrôles continus : 20%
- Contrôle TP : 25%
- Atelier Data : 25%
- Examen : 30%

- Démarches didactiques:
  - Notes de cours, Travaux dirigés
- Travaux pratiques:
  - MySQL, Oracle

1. Rappel

Dépendance fonctionnelles

- Le but des dépendances fonctionnelles et de la théorie de la normalisation (resp. dénormalisation) est de s'assurer que le schéma relationnel défini pour une base de données est correctement construit. Une mauvaise schéma relationnel peut en effet entrainer des anomalies lors de manipulation de la base de données.

Dépendance fonctionnelles

Produit	Quantité	Couleur	Fournisseur	Adresse
parapluie	110	rouge	Labaleine	Paris
chapeau	50	vert	Lemelon	Lyon
sac à main	65	noir	Toutcuir	Lyon
parasol	15	jeune	Labaleine	Paris
ombrelle	5	rouge	Labaleine	Paris
ceinture	25	vert	Letour	Nantes
sac à main	65	noir	Legrand	Paris

- Cette relation est mal construite : les informations concernant les fournisseurs sont redondantes
- Cette solution n'est pas raisonnable. La base de données va très rapidement devenir incohérente.

- Quel est la solution pour construire une base de données bien conçue et cohérente ?

Dépendance fonctionnelles

- La solution est dont d'éviter toute redondance dans la base de données. Ceci exige que le schéma de la base de données soit **bien construit**. La méthode pour cela se décompose en deux étapes :

1. étude des dépendances entre données;
2. décomposition et normalisation des relations

Dépendance fonctionnelles

- **Définition**:

Soit R(X,Y,Z) un schéma de relation, avec X, Y, Z des ensembles d'attributs. On dit qu'il existe **une dépendance fonctionnelle** entre X et Y si la connaissance d'une valeur de X détermine au plus une valeur de Y. La notation adoptée est la suivante : X->Y

- Remarque:

Dépendance fonctionnelles expriment la sémantique entre les attributs

Propriétés des dépendances fonctionnelles (ou axiomes d'Amstrong)

Propriété	Equivalence
DF1 (Réflexivité)	X->Y ⇒ Y->X
DF2 (augmentation)	X->Y ⇒ X,Z ->Y,Z
DF3 (transitivité)	X->Y ET Y->Z ⇒ X->Z
DF4 (pseudo-transitivité)	X -> Y ET Y,W -> Z ⇒ X,W -> Z
DF5 (union)	X -> Y ET X -> Z ⇒ X -> Y,Z

Exemple

- Soit la table ligne_de_commande (N° de commande, N° article, désignation, quantité commandé). On a les dépendances fonctionnelles suivantes:

✓ N° article -> désignation

✓ N° de commande, N° article -> quantité commandé

Normalisation

- La normalisation est un processus de décomposition d'une table universelle en plusieurs tables en évitant les problèmes **d'incohérence** et de **redondances**
- **Objectif:**
  - définir une notion de "qualité" de schéma
  - pouvoir comparer deux schémas de relation

- **Remarque:**

  + Un "bon" schéma de base de données est un schéma qui vérifie les 3 formes normales

  + La normalisation nécessite les dépendances fonctionnelles

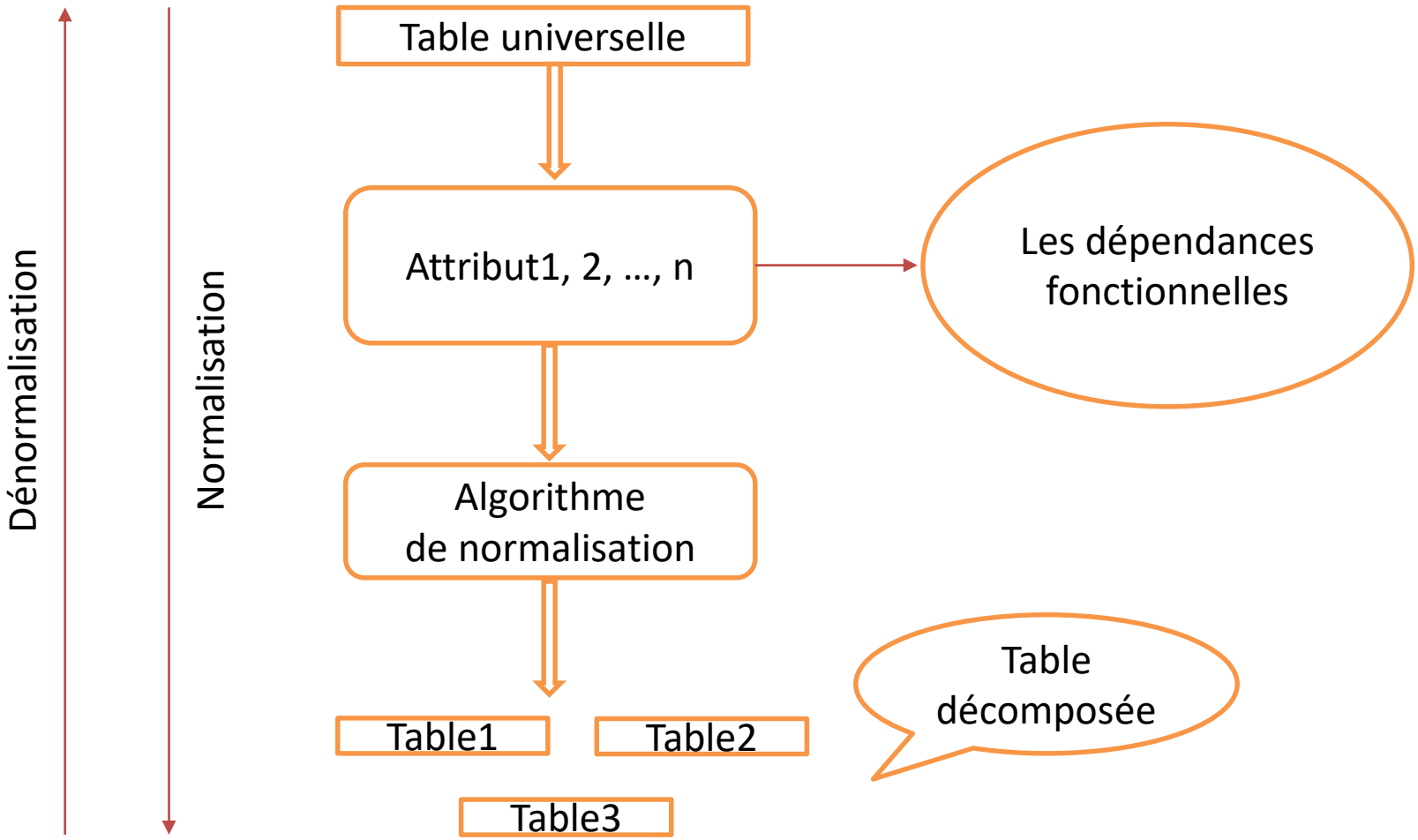
Normalisation

- **NB:**

  On ne présente ici que les formes normales dont la définition utilise exclusivement les dépendances fonctionnelles. Si on prend en compte d'autres dépendances entre données comme les dépendances multivaluées on obtient alors les 4FN et 5FN. Les **3FN** reste cependant l'objectif de normalisation le plus "classique".

- La normalisation est un processus **réversible** (.i.e., a partir des tables décomposés on peut constitue la table universelle (dénormalisation))

Normalisation



Les formes normales

- **1FN** : une relation est en première forme normale ssi tout attribut a une valeur atomique
- **2FN** : une relation est en deuxième forme normale ssi elle est en 1FN et si tous les attributs non clés sont pleinement dépendants des clés (toutes les dépendance entre une clé et un attribut non clé sont élémentaires)
- **3FN** : une relation est en troisième forme normale ssi elle est en 2FN et si tous les attributs non clés sont directement et pleinement dépendants des clés (si tout attribut non clé ne dépend pas d'un autre attribut non clé)
- D'autre formes ont été proposée:
- Boyce-codd, 4FN, 5FN, 6FN

Exemple (1FN)

- **1FN** c'est le point de départ vers les autres formes normales
- **1FN** exige chaque valeur d'un attribut soit atomique (un attribut atomique est celui ayant une seule valeur) table ouvrage

Code ouvrage	Titre	Auteur
001	Base de données	CHERRADI, BOUDAA
002	Administration Linux	EL ALLAOUI, EL MOUSSAOUI

- Ici l'attribut « **Auteur** » c'est un attribut qui accepte plusieurs valeur, est donc n'est pas **atomique**. Et par la suite cette relation n'est pas en première FN

Comment passer en première forme normal

1. Sortir l'attribut non atomique de la table initiale
2. Transformer l'attribut non atomique en table, ajouter dans cette nouvelle table la clé primaire de la première table

Code ouvrage	Titre
001	Base de données
002	Administration Linux

Code auteur	Auteur	#code ouvrage
1	CHERRADI	001
2	BOUDAA	001
3	EL ALLAOUI	002
4	EL MOUSSAOUI	002

Exemple (2FN)

- Cette deuxième forme normale exige que les tables est déjà en première forme normale
- Elle ne concerne que les tables a **clé primaire composé** (composée de plusieurs attributs)
- La règle impose que les attributs non-clé primaire dépendent de la totalité de la clé primaire

Table article_commande
N° commande
N° article
Désignation
Quantité commandé

Clé primaire composé

- Comme vous remarquez la clé de cette nouvelle table est composé de deux attributs.
- Est-ce que cette table est en 1FN? OUI
- Est-ce que cette table est en 2FN (i.e., toute attribut non clé dépend de la totalité de la clé)? NON
- Cette règle **n'est pas vérifie** parce que l'attribut «*désignation*» dépend qu'une partie de la clé, qui est N° article. Et donc cette relation n'est pas en 2FN

Processus de normalisation en deuxième forme normale

1. Regrouper dans une table les attributs dépendant de la totalité de la clé et conserver cette clé pour cette table
2. Regrouper dans une table les attributs dépendant d'une partie de la clé, et faire de cette partie la clé primaire de cette nouvelle table

Table ligne_de_commande
N° commande
N° article
Quantité commandé

Table article
N° article
Désignation

Exemple (3FN)

- La mise en 3ème forme normale ne s'applique sur les tables déjà en première et en deuxième forme normale
- La règle a pour objet l'élimination des dépendances fonctionnelles transitives au sein d'une table

Table commande
N° commande
Date commande
N° client
Nom client

- Est-ce que cette table est en 1FN? OUI
- Est-ce que cette table est en 2FN? OUI
- Est-ce que cette table est en 3FN ? Cette table a une dépendance fonctionnelle transitive, donc n'est pas en **3FN**
- N° client (attribute non-clé) determine nom client

Processus de normalisation en troisième forme normale

1. Conserver dans la table initiale les attributs dépend directement de la clé
2. Regrouper dans une autre table les attributs dépendant transitivement de la clé; l'attribut de transition reste dupliqué dans la table initiale, et devient la clé primaire de la nouvelle table

Table commande
N° commande
Date commande
N° client

Table client
N° client
Nom client

Les contraintes d'intégrité (CI)

- Une contrainte d'intégrité est une règle qui définit la cohérence d'une donnée ou d'un ensemble de données de la BD.
- Il existe deux types de contraintes :
  - sur une colonne unique,
  - ou sur une table lorsque la contrainte porte sur une ou plusieurs colonnes.
- **Remarque:** Les contraintes sont définies au moment de la création des tables.
- **Exemple:**
- PRIMARY KEY, Foreign Key, UNIQUE, NOT NULL, REFERENCES(3 modes: cascade, setNull, restrict=default), CHECK(), …

Exemple (CI)

```
CREATE TABLE employee (
    id number(5) PRIMARY KEY,
    name char(20),
    dept char(10),
    age number(2),
    salary number(10),
    location char(10)
);
```

```
CREATE TABLE employee (
    id number(5) CONSTRAINT
    emp_id_pk PRIMARY KEY, name
    char(20),
    dept char(10),
    age number(2),
    salary number(10),
    location char(10)
);
```

```
CREATE TABLE employee (
        id number(5), name char(20), dept char(10), age number(2),
        salary number(10), location char(10),
        CONSTRAINT emp_id_pk PRIMARY KEY (id)
);
```

Algèbre Relationnelle

- L'algèbre relationnelle a été inventé en 1970 par E. Codd dont le but de **formaliser les opérations** sur les ensembles. Elle constitue une collection d'opération formelles qui agissant sur des relations et produisent des relations
- **Les opérations ensemblistes**: Union, Intersection, Différence, Produit cartésien, jointure, projection, restriction, division,
- Une opération relationnelle agit sur ***une ou plusieurs tables*** et a pour résultat ***une table***

- La projection et la sélection sont des opérations qui s'appliquent à une table
- Les opérations ensemblistes (union, intersection, différence) ne peuvent être utilisés qu'avec **deux tables** ayant les mêmes attributs et fournissent une troisième table ayant les même attributs
- Le produit cartésien et la jointure fournissent une troisième table à partie de deux tables quelconque

Projection

- La projection définit une relation restreinte a un sous-ensemble des attributs d'une table **T**, en extrayant les valeurs des attributs spécifiés

$$\Pi_{A, B, C}(T)$$

- **Exemple:**

Table **Ventes**

Magasin	Produit	Prix
Monoprix	Pomme	2,20
Monoprix	Cerise	4,80
Franprix	Poire	3,75
Franprix	Abricot	2,95
Champion	Fraise	3,75
Monoprix	Abricot	3,50
Franprix	Pomme	2,20
Champion	Pomme	2,40

$\Pi_{\text{Produit, Prix}}$ **(Ventes)**

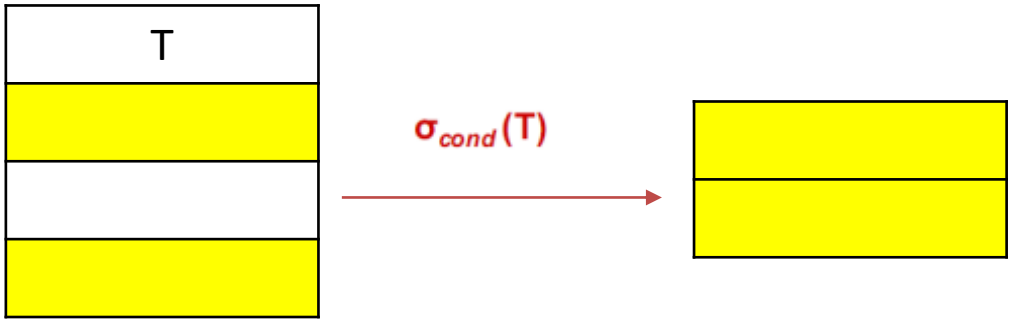
Produit	Prix
Pomme	2,20
Cerise	4,80
Poire	3,75
Abricot	2,95
Fraise	3,75
Abricot	3,50
Pomme	2,40

Π_{Magasin} **(Ventes)**

Magasin
Monoprix
Franprix
Champion

Sélection

- La sélection agit sur une condition, il consiste donc à garder les lignes de la table **T** vérifiant la condition



- **Exemple:**

Ventes

Magasin	Produit	Prix
Monoprix	Pomme	2,20
Monoprix	Cerise	4,80
Franprix	Poire	3,75
Franprix	Abricot	2,95
Champion	Fraise	3,75
Monoprix	Abricot	3,50
Franprix	Pomme	2,20
Champion	Pomme	2,40

$\sigma_{Magasin = 'Monoprix'}(Ventes)$

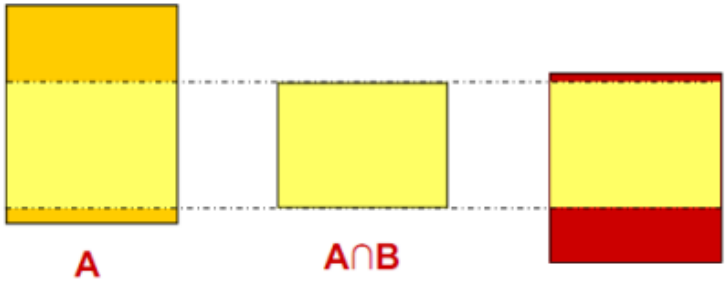
Magasin	Produit	Prix
Monoprix	Pomme	2,20
Monoprix	Cerise	4,80
Monoprix	Abricot	3,50

$\sigma_{Prix > 3}(Ventes)$

Magasin	Produit	Prix
Monoprix	Cerise	4,80
Franprix	Poire	3,75
Champion	Fraise	3,75
Monoprix	Abricot	3,50

Intersection

- L'intersection contient toutes les lignes communes aux deux tables A et B



- **Exemple:**

Fruit

Produit	Prix
Pomme	2,20
Cerise	4,80
Poire	3,75
Abricot	2,95
Fraise	3,75

Primeur

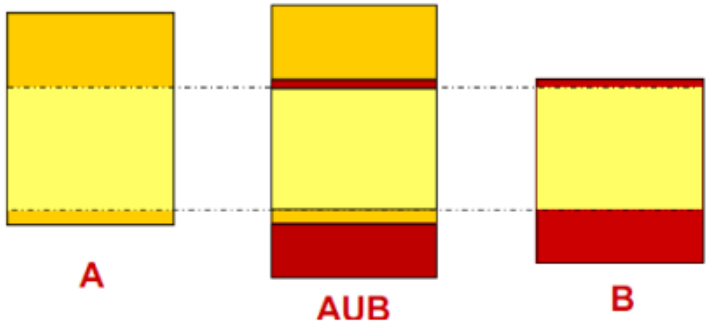
Produit	Prix
Cerise	4,80
Abricot	3,50
Fraise	3,75

Fruit ∩ Primeur

Produit	Prix
Cerise	4,80
Fraise	3,75

Union

- L'union contient toutes toutes les lignes de A et toutes les lignes de B **sans doublon**



- **Exemple:**

Fruit

Produit	Prix
Pomme	2,20
Cerise	4,80
Poire	3,75
Abricot	2,95
Tomate	2,75

Legume

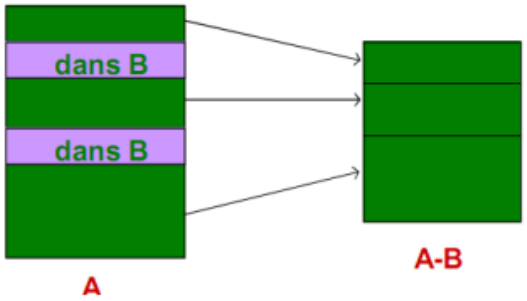
Produit	Prix
Carotte	1,40
Poireau	1,25
Salade	2,05
Tomate	2,75

Fruit U Legume

Produit	Prix
Pomme	2,20
Cerise	4,80
Poire	3,75
Abricot	2,95
Tomate	2,75
Carotte	1,40
Poireau	1,25
Salade	2,05

Différence

- La différence contient toutes les lignes de A qui ne se trouvent pas dans B



- **Exemple:**

Fruit

Produit	Prix
Pomme	2,20
Cerise	4,80
Poire	3,75
Abricot	2,95
Fraise	3,75

Primeur

Produit	Prix
Cerise	4,80
Abricot	3,50
Fraise	3,75

Fruit-Primeur

Produit	Prix
Pomme	2,20
Poire	3,75
Abricot	2,95

Primeur-Fruit

Produit	Prix
Abricot	3,50

Division

- La division de R1 par R2, sachant que R1 et R2 ont au moins un attribut commun (c'est à dire de même nom et de même domaine), produit une relation R3 possède les attributs non communs aux deux relations initiales et est formée de tous les n-uplets qui, concaténés à chacun des n-uplets du diviseur.
- **Exemple:**

PARTICIPER

Athlète	Epreuve
Dupont	200 m
Durand	400 m
Dupont	400 m
Martin	110 m H
Dupont	110 m H
Martin	200 m

EPREUVE

Epreuve
200 m
400 m
110 m H

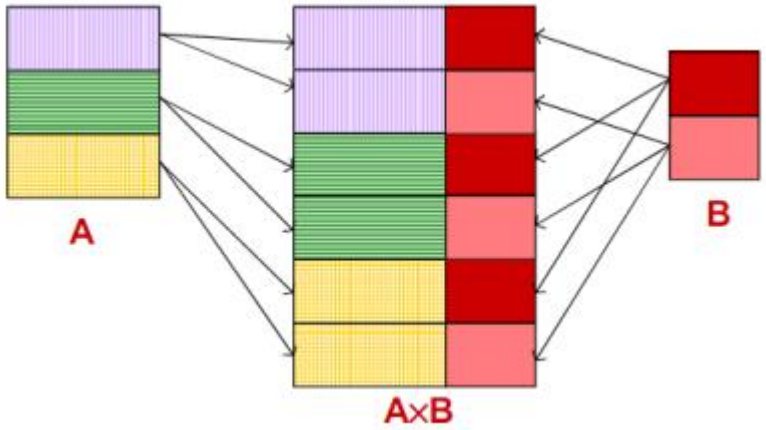
DIVISION (PARTICIPER,EPREUVE)

Athlète
Dupont

"L'athlète Dupont participe à toutes les épreuves"

Produit Cartésien

- Le produit cartésien il permet de faire pour chaque ligne de A fabriquer autant de lignes qu'il y a de lignes dans B par concaténation



nblignes(T_1X T_2) = nblignes(T_1)*nblignes(T_2)

Produit Cartésien

- **Exemple:**

Supermarché5ème

Magasin	Adresse	Horaire	Offre	Prix
Monoprix	Gobelins	20H	Pomme	2,20
Franprix	Mouffetard	20H45	Abricot	2,95
Champion	Monge	22H	Fraise	3,75

Primeur

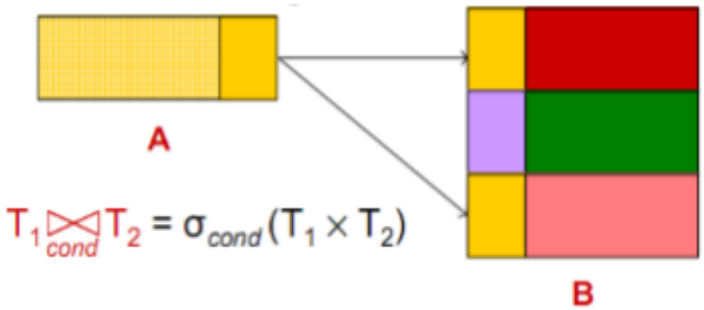
Produit	Prix
Cerise	4,80
Abricot	3,50
Fraise	3,75

Supermarché5ème × Primeur

Magasin	Adresse	Horaire	Offre	Prix	Produit	Prix
Monoprix	Gobelins	20H	Pomme	2,20	Cerise	4,80
Monoprix	Gobelins	20H	Pomme	2,20	Abricot	3,50
Monoprix	Gobelins	20H	Pomme	2,20	Fraise	3,75
Franprix	Mouffetard	20H45	Abricot	2,95	Cerise	4,80
Franprix	Mouffetard	20H45	Abricot	2,95	Abricot	3,50
Franprix	Mouffetard	20H45	Abricot	2,95	Fraise	3,75
Champion	Monge	22H	Fraise	3,75	Cerise	4,80
Champion	Monge	22H	Fraise	3,75	Abricot	3,50
Champion	Monge	22H	Fraise	3,75	Fraise	3,75

Jointure

- La jointure c'est l'opération permettant de « coller » au bout des lignes de la table A toutes les lignes de la table B vérifiant la condition de jointure



- On appelle jointure de T1 et T2 sur la condition cond; la sélection sur cond effectuée sur T1 X T2
- Le cas le plus fréquent est celui où la condition est l'égalité de deux attributs

Jointure

- Exemple:

Etudiant

NumEtu	Nom	Groupe
10	Martin	211
11	Videau	221
22	Durand	221
32	Rossi	211

Projet

Groupe	ResProjet
211	Fournier
221	Astier

Etudiant ⋈ Projet

NumEtu	Nom	Groupe	ResProjet
10	Martin	211	Fournier
11	Videau	221	Astier
22	Durand	221	Astier
32	Rossi	211	Fournier