Introduction à la science informatique, 2018-2019

Bases de Données

Gilles Subrenat[1]

[1]Laboratoire XLIM-SIC
Gilles.Subrenat@univ-poitiers.fr

Journées de formation 2018-2019, 2 et 3 octobre 2018

Plan général : A - Bases de Données

Plan → A - Bases de Données

- Manipulations sur une base de données
  - Stocker des données
  - Retrouver des données
  - Modifier des données
  - Effacer des données
    $\Rightarrow$ CRUD (Create, Read, Update, Delete)
- Contraintes
  - cohérence (données correctes)
  - optimisation (rapidité)
  - universalité (langage haut niveau)
  - robustesse (gestion des dysfonctionnements)
  - sécurité (piratage)

- Exemples
  - mes DVD
  - les favoris de mon navigateur
  - mon blog sur internet
  - les formations, les emplois du temps, ...
  - les comptes d'une banque
  - les cartes d'un calculateur d'itinéraires
  - les horaires des trains
  - les comptes d'un réseau social
  - l'archivage des pages d'un moteur de recherche
  - big data : ensemble des heures de départ/arrivée des trains sur un an
  - ...
- Caractéristiques
  - confidentialité
  - volume
  - temps de traitement

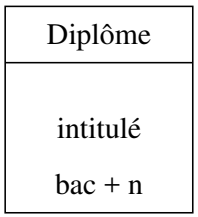
- SGBDR (Système de Gestion de Bases de Données Relationnel)
  - basé sur le langage SQL (Structured Query Language)
  - exemples : Oracle, MySQL, MariaDB, PostgreSQL, Access, SQL Server, SQLite, ...
- Moteurs de base de données
  - fonctionnement interne
  - exemples : MyISAM, InnoDB, ...
- Accès simultanés
  - modifications simultanées sur les mêmes données
  - modifications simultanées sur des données indépendantes

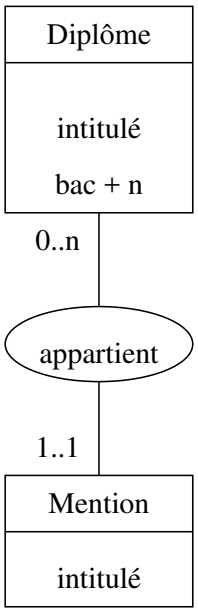
  $\Rightarrow$ notion de verrouillage (géré par le SGBDR)
- Anomalies
  - panne/crash au milieu d'instructions de modification
    $\rightarrow$ risque d'incohérence

  $\Rightarrow$ notion de transaction (gérée par l'utilisateur et le SGBDR)

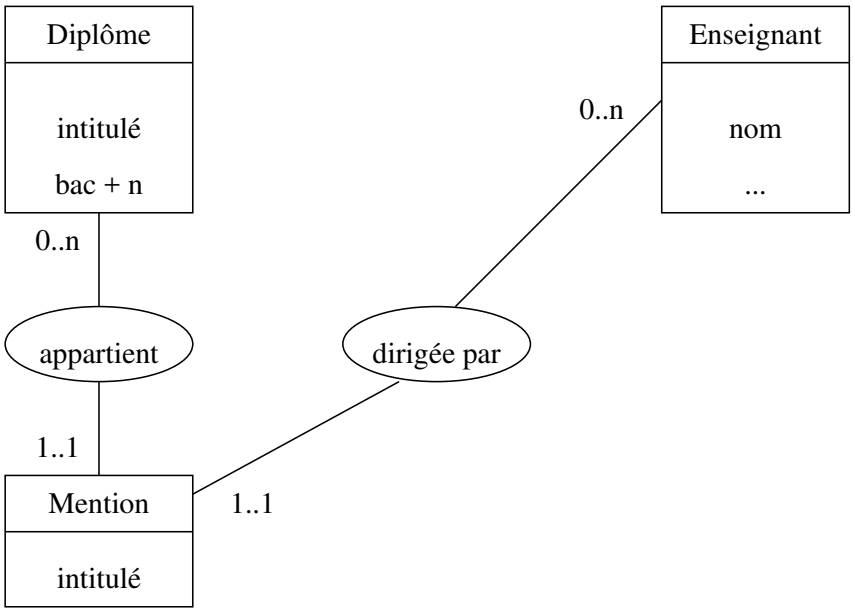
Plan → A - Bases de Données

- Identifier
  - les différents regroupements de données
  - les relations entre eux
  - les cardinalités
- But
  - spécifier : description non ambiguë (pour les développeurs)
  - discuter avec un client non informaticien
- Exemple : formations universitaires
  - diplômes, mentions, parcours
  - unités d'enseignements
  - enseignants
  - disciplines
  - ...
- Définitions - outils
  - MCD (Modèle Conceptuel de Données)
  - SEA (Schéma Entités-Associations)

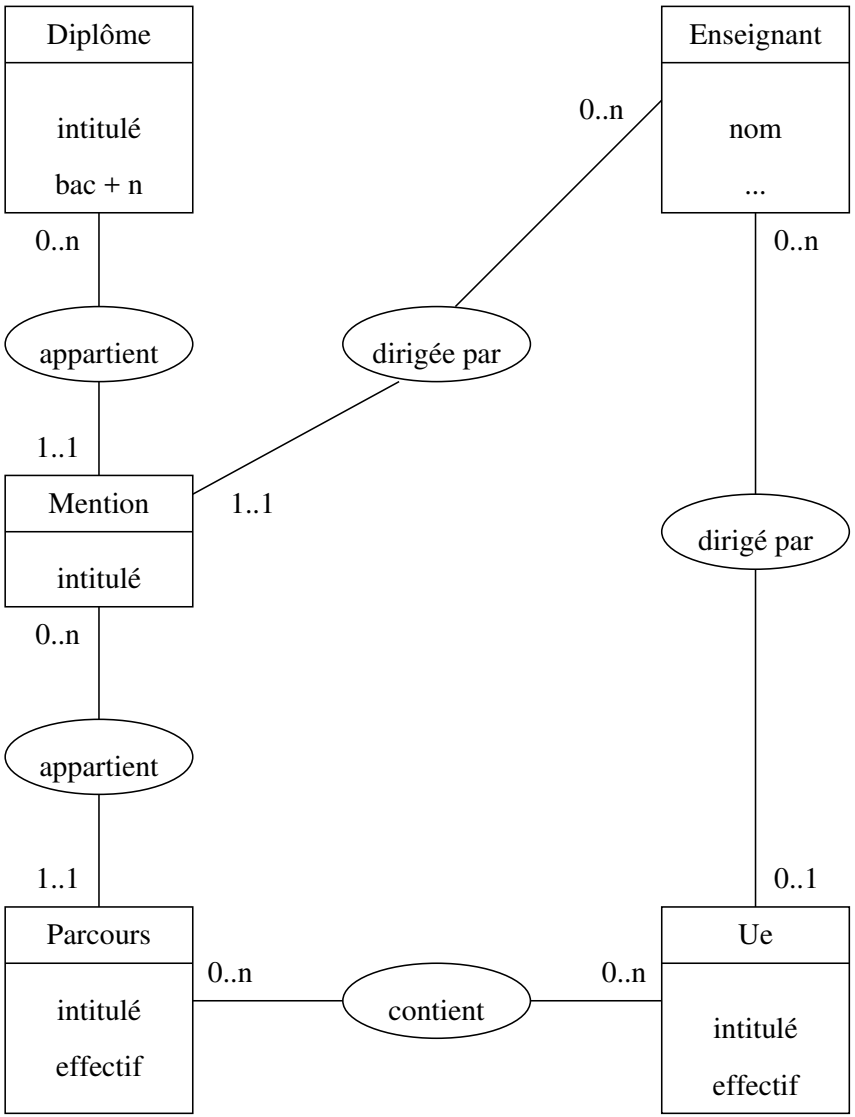

Diplôme
intitulé
bac + n

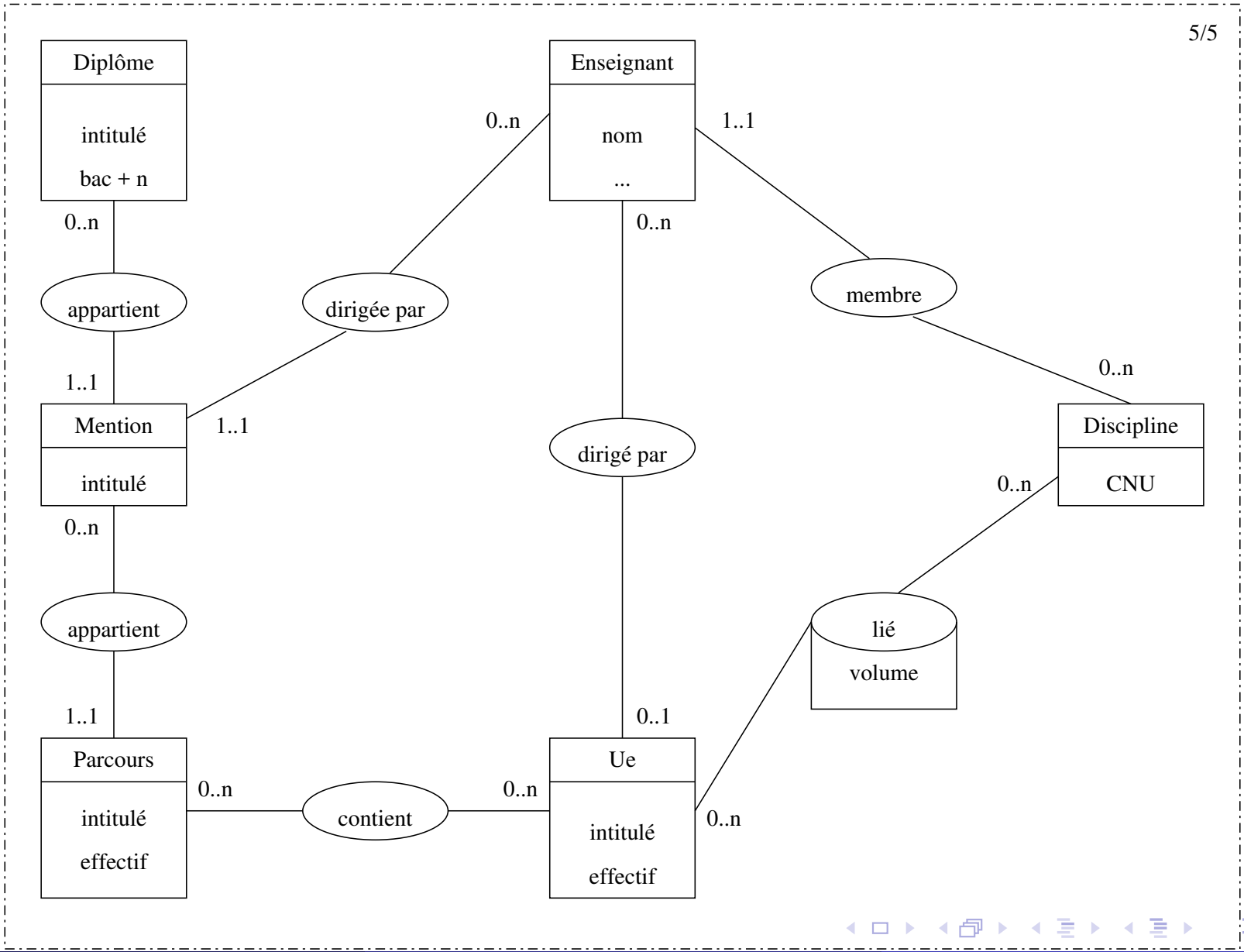

Diplôme
intitulé
bac + n
0..n
appartient
1..1
Mention
intitulé

3/5


Diplôme
intitulé
bac + n
Enseignant
nom
...
0..n
0..n
appartient
dirigée par
1..1
1..1
Mention
intitulé

4/5


Diplôme
intitulé
bac + n
Enseignant
nom
...
0..n
0..n
0..n
appartient
dirigée par
1..1
Mention
intitulé
1..1
dirigé par
0..n
appartient
1..1
0..1
Parcours
intitulé
effectif
0..n
contient
0..n
Ue
intitulé
effectif


5/5
Diplôme
intitulé
bac + n
Enseignant
nom
...
0..n
1..1
0..n
0..n
appartient
dirigée par
membre
1..1
0..n
Mention
intitulé
1..1
dirigé par
Discipline
CNU
0..n
0..n
appartient
lié
volume
1..1
0..1
Parcours
intitulé
effectif
0..n
contient
0..n
Ue
intitulé
effectif
0..n

- Table
  - ensemble d'informations structurées en tableau
  - exemple : table de personnes : nom, naissance, nationalité
- Enregistrement ou ligne
  - une entrée dans la table
  - exemple : Tintin, 1929-01-10, Belge
- Champ ou colonne
  - un attribut de la table
  - exemple : âge
- Type
  - ensemble des valeurs possibles pour un champ
  - exemples : int, float, double, decimal, boolean, date, datetime, varchar, text, longtext, binary, ...
- Cardinalité
  - rapport "numérique" entre deux tables
  - exemple : $1 - n$, un article est relié à un ou plusieurs auteurs
  - exemple : $1 - 1$, une noticie est reliée à un et seul produit

- Exemple

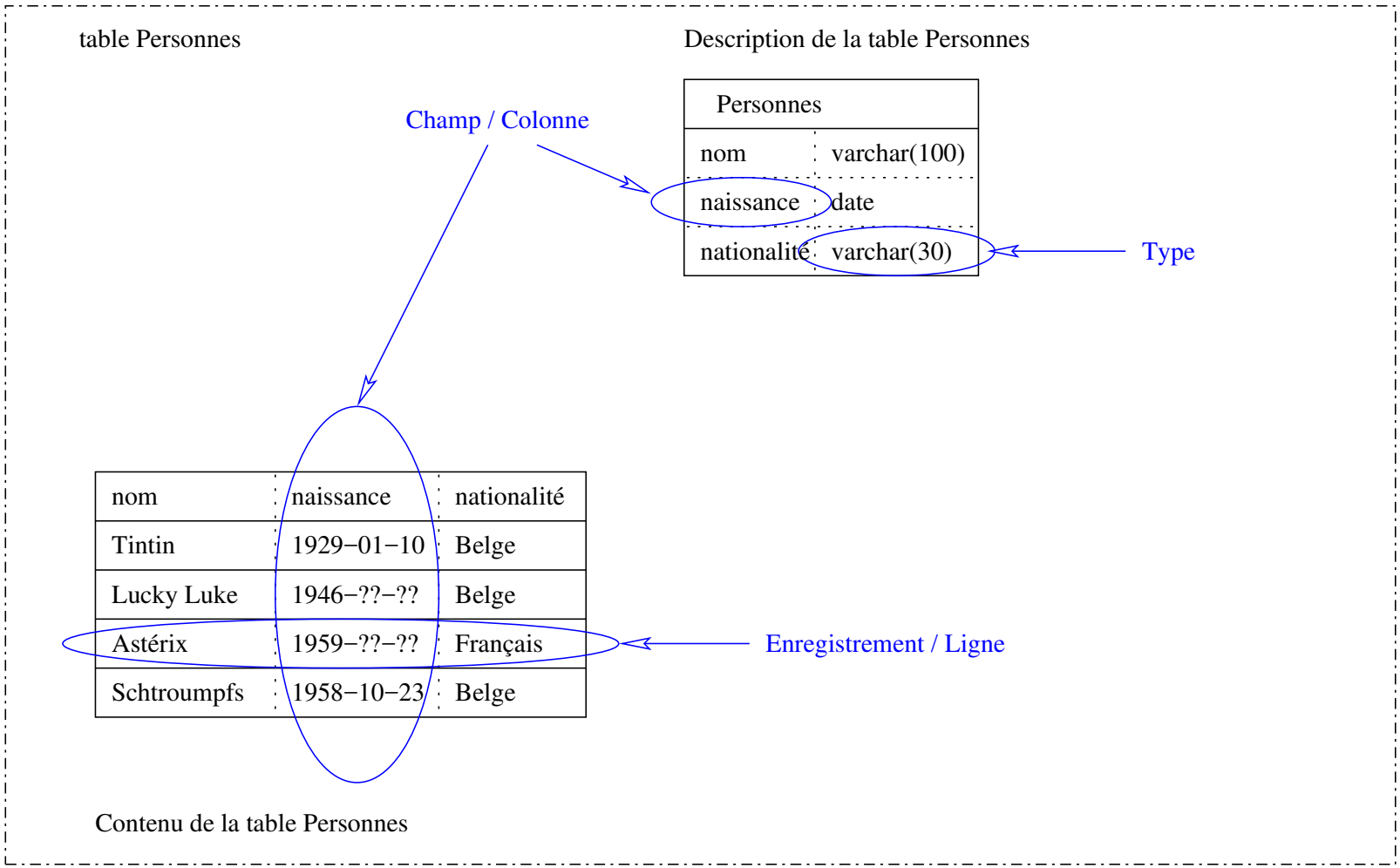


table Personnes
Description de la table Personnes
Personnes
nom
varchar(100)
naissance
date
nationalité
varchar(30)
Champ / Colonne
Type
nom
naissance
nationalité
Tintin
1929−01−10
Belge
Lucky Luke
1946−??−??
Belge
Astérix
1959−??−??
Français
Schtroumpfs
1958−10−23
Belge
Enregistrement / Ligne
Contenu de la table Personnes

- Règles de construction (respect des formes normales)
- Attribut = valeur atomique
  - i.e. non décomposable sans perte
  - exemples : “âge”, “nom”, “prénom”, “date”, ...
- Non redondance
  - il ne faut pas stocker une information :
    - répétée en plusieurs endroits
    - qui peut être calculée à partir d'autres informations,
  - exemple interdit : “nom ville naissance” ($\Rightarrow$ table dédiée)
  - exemple interdit : “âge + date naissance”
- Non respect des règles
  - pour des raisons de performance
  - nécessité de traitements pour gérer les incohérences
  - exemple : solde d'un compte bancaire

Plan → A - Bases de Données

- But : implémentation informatique
- Identification non ambiguë des données
  - clé primaire
- Liens entre les tables
  - clé étrangère
  - table de jointure
- Code de gestion de cohérence
  - valeurs restreintes pour un type
    exemple : âge positif
  - synchronisation des données dupliquées
    exemple : ajout d'une opération bancaire met à jour le solde
  - inter-dépendance des données
    exemple : détention du permis implique une âge supérieur à 18
  - ...

- Clé primaire : un ou plusieurs champs qui désignent de manière unique chaque enregistrement d'une table
- Exemples
  - numéro de sécurité sociale
  - login informatique
  - code ISBN
- Contre-exemples
  - nom
  - et même nom+prénom
- Autre solution
  - entier
  - sans signification
  - auto-incrémenté par le SGBDR

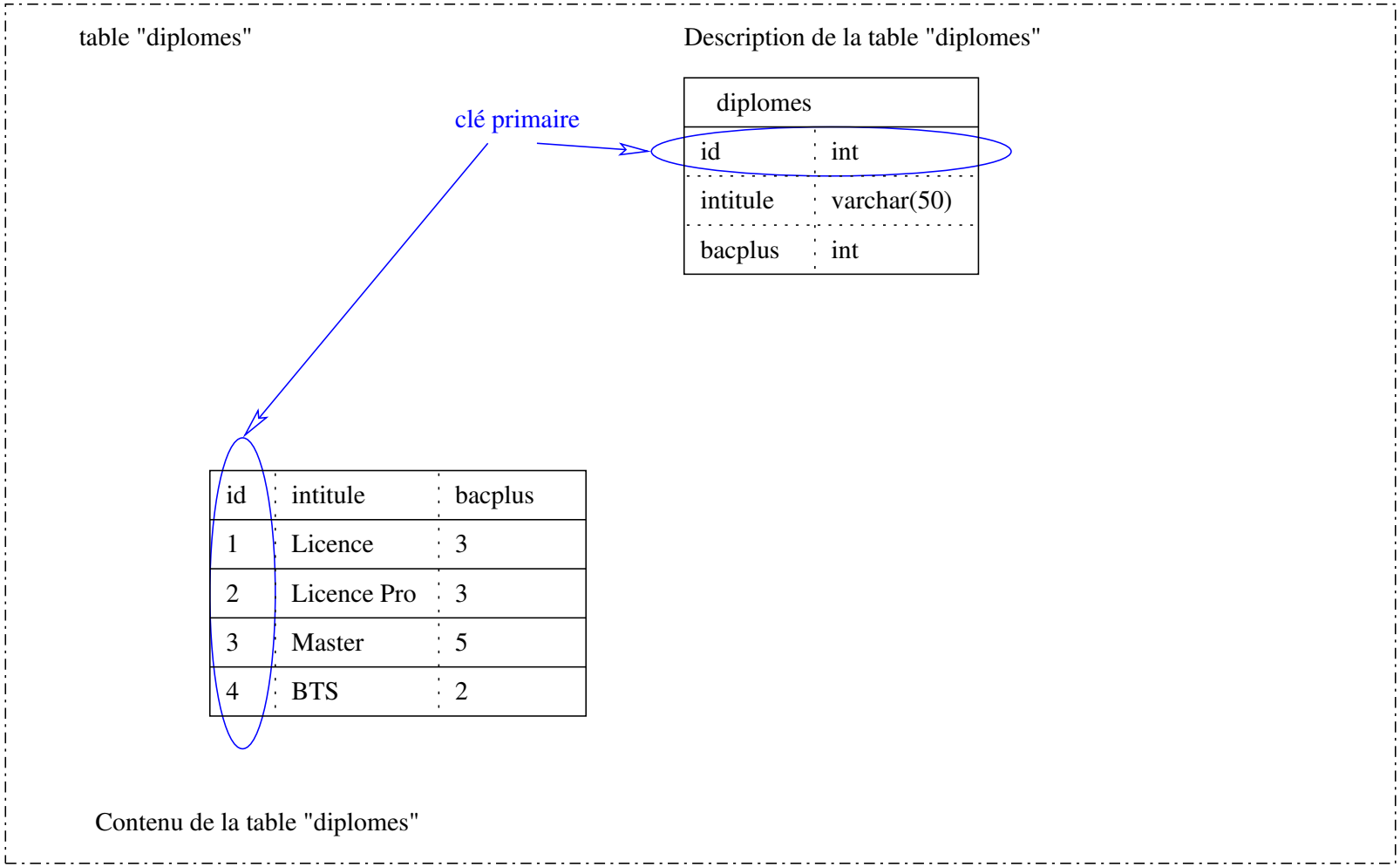
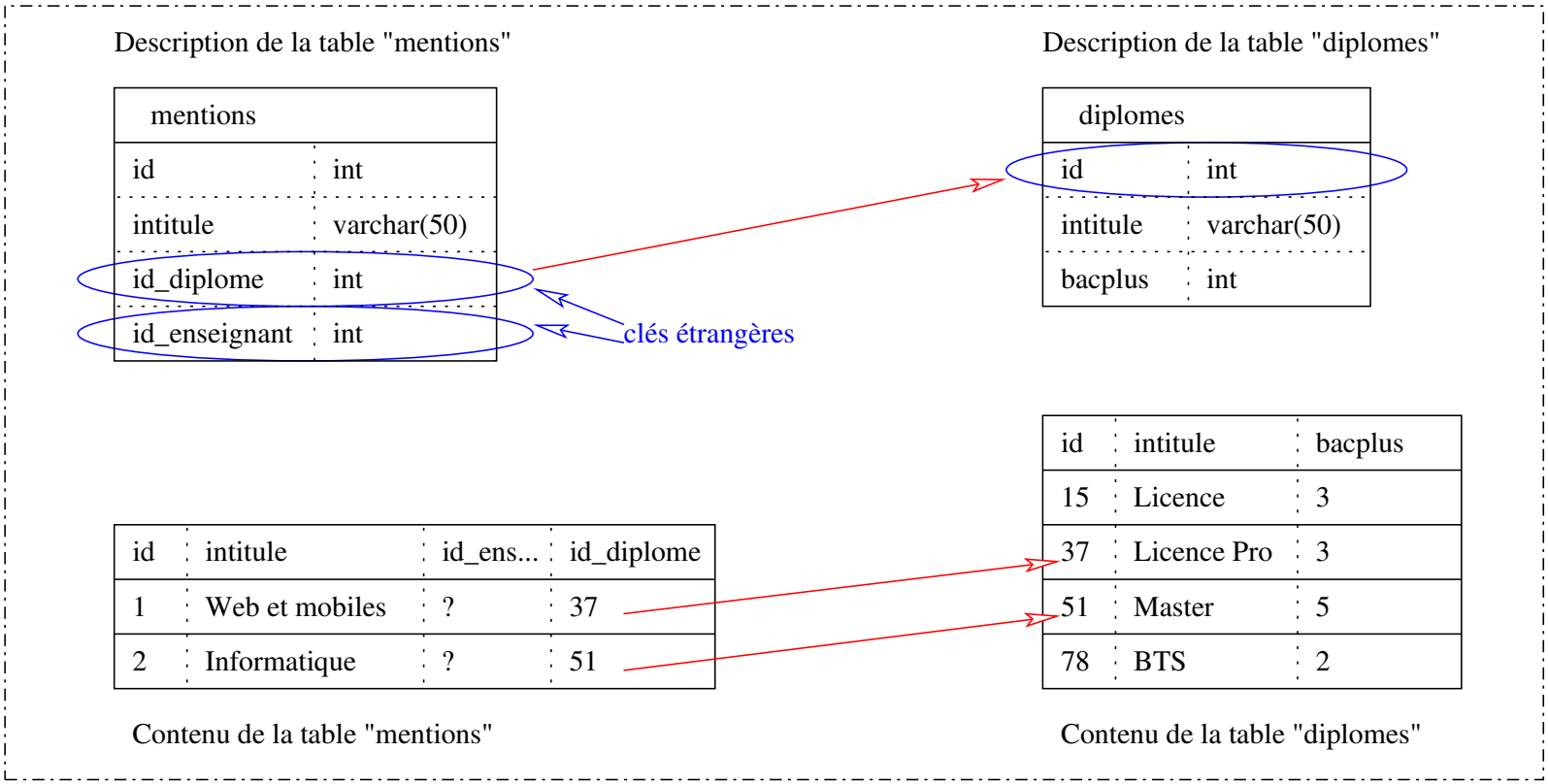


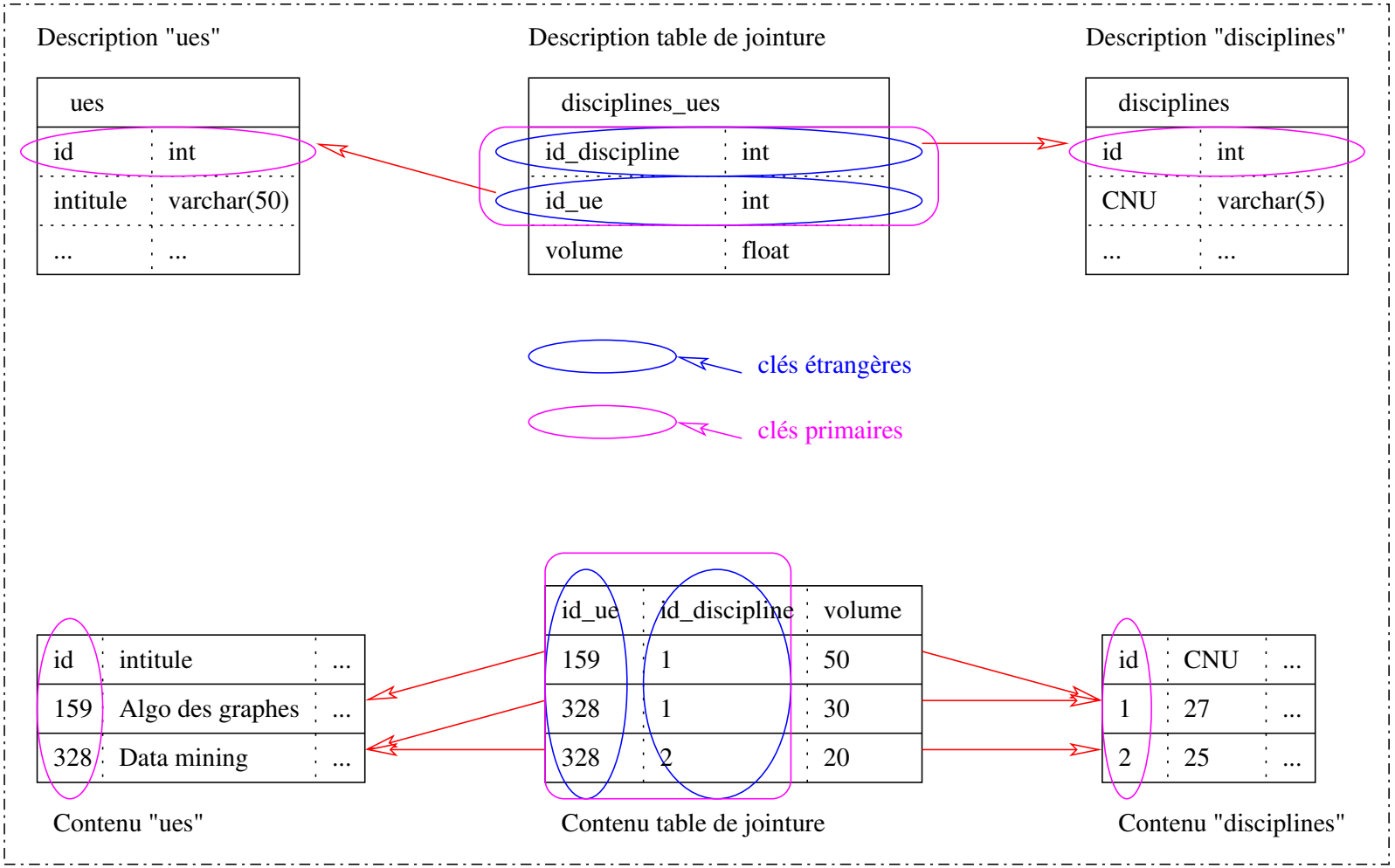
table "diplomes"
Description de la table "diplomes"
clé primaire
diplomes
id int
intitule varchar(50)
bacplus int
id intitule bacplus
1 Licence 3
2 Licence Pro 3
3 Master 5
4 BTS 2
Contenu de la table "diplomes"

- Définition :
  - une ligne d'une table est reliée à une ligne d'une autre table
  - la clé étrangère est dans la table source et désigne la clé primaire de la table cible.



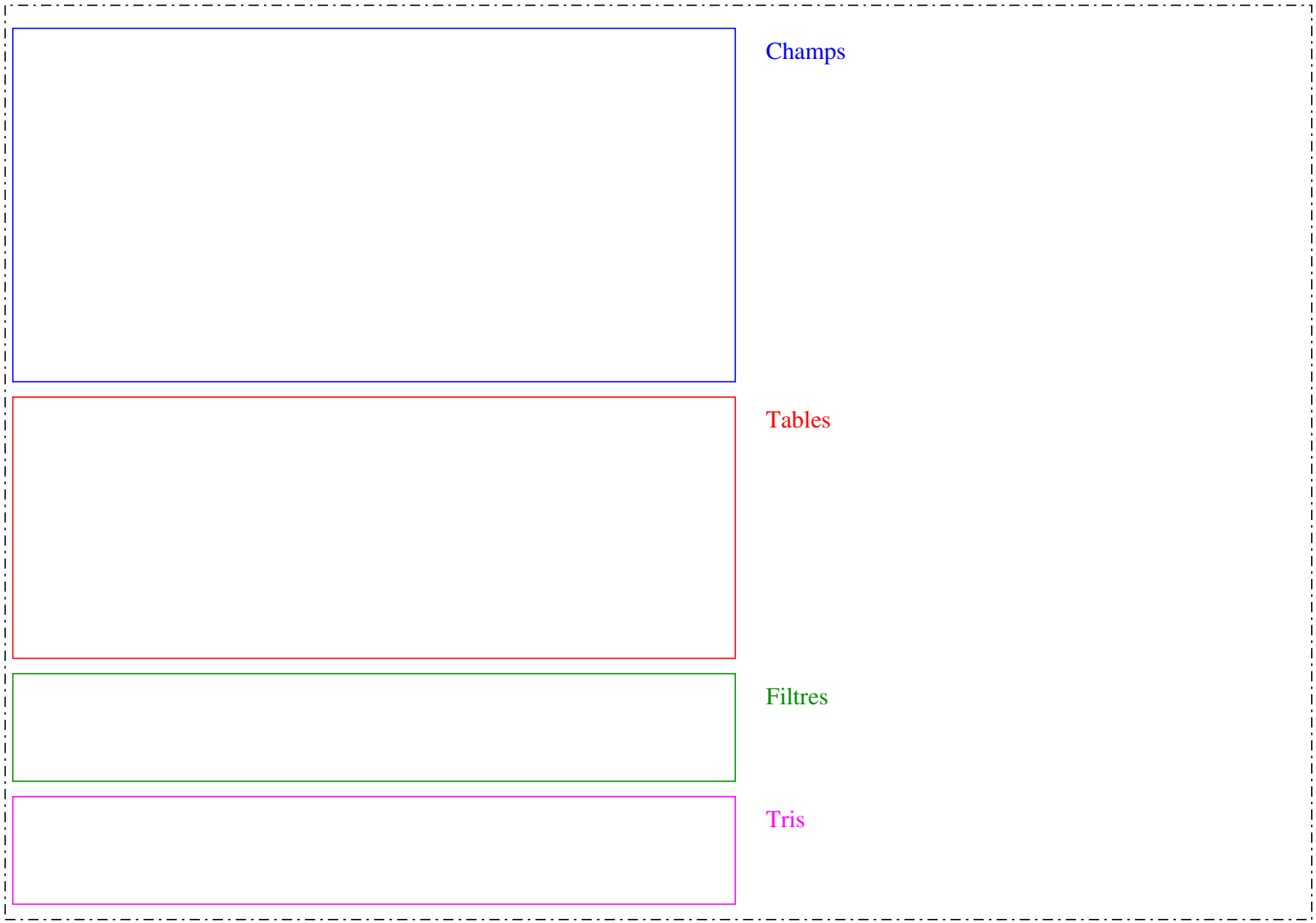
- Relation 1-1 ou 1-n ?
  - impossible de faire la différence
  - exemple 1-1 : “produits” et “notices”
  - exemple 1-n : “mentions” et “parcours”
- Cardinalité 0..1 ou 1..1 ?
  - clé étrangère à NULL
- Clés étrangères incohérentes (erreur 404 !)
  - clé étrangère pointe une donnée disparue
  - solution : gestion par le moteur (ex InnoDB)
  - intérêt : aucun risque d’incohérence
  - inconvénient : moteur moins performant

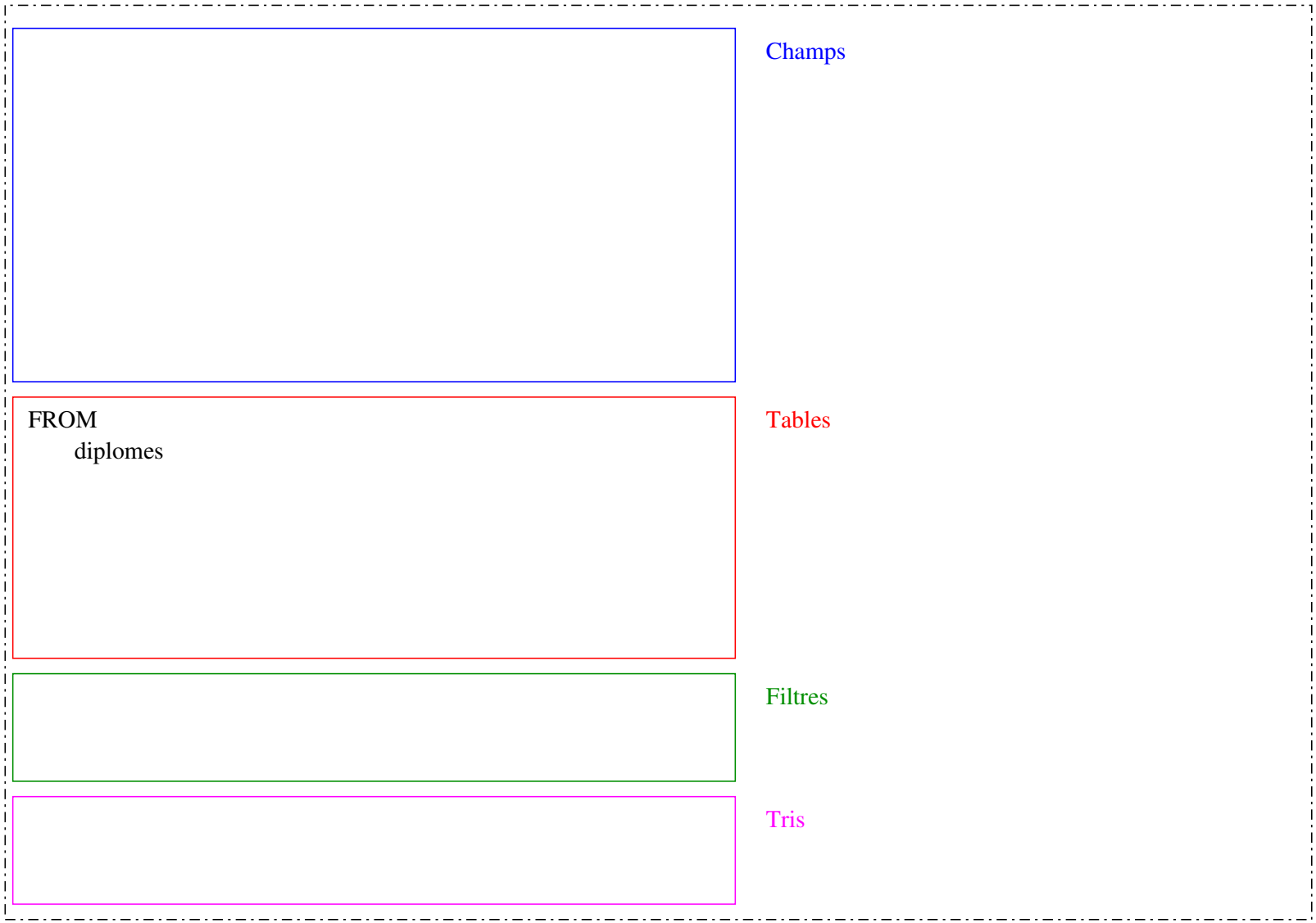
- Comment implémenter une relation n-n ?
  - exemple : lien entres parcours et UEs
  - exemple : lien entre UEs et disciplines
- Impossible de mettre la clé étrangère dans une des deux tables
- Solution : table intermédiaire dite table de jointure
  - contient tous les couples des deux tables d'origine
  - via des couples de clés étrangères
- Remarque : une table de jointure peut contenir d'autres informations, on dit que la relation est attribuée.
  - exemple : la relation entre UEs et disciplines est associée à un volume horaire
- Remarque : une table de jointure peut relier 2, 3, ... tables

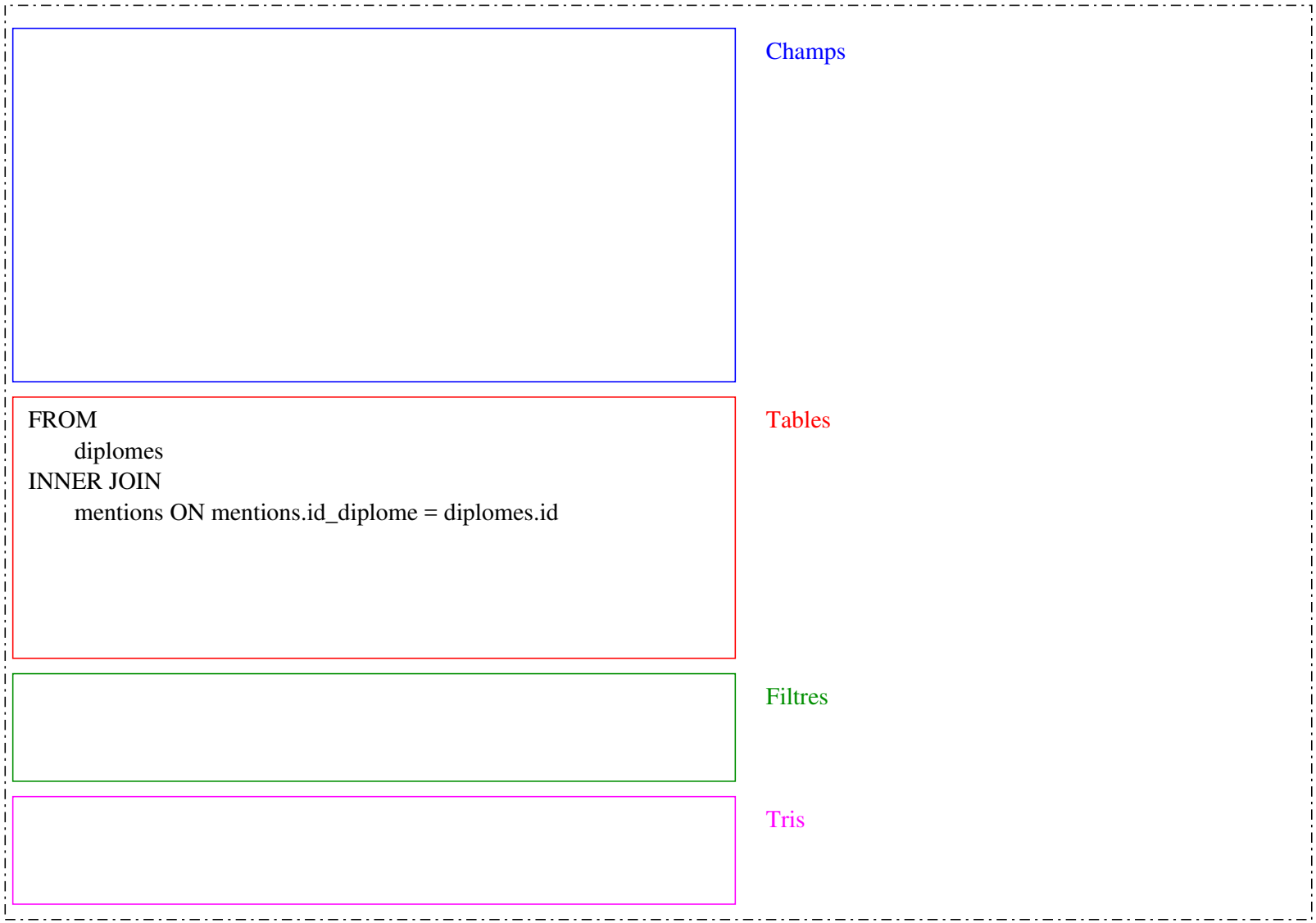

Description "ues"
Description table de jointure
Description "disciplines"
ues
id int
intitule varchar(50)
... ...
disciplines_ues
id_discipline int
id_ue int
volume float
disciplines
id int
CNU varchar(5)
... ...
clés étrangères
clés primaires
id intitule ...
159 Algo des graphes ...
328 Data mining ...
id_ue id_discipline volume
159 1 50
328 1 30
328 2 20
id CNU ...
1 27 ...
2 25 ...
Contenu "ues"
Contenu table de jointure
Contenu "disciplines"

Plan → A - Bases de Données

- Requête d'interrogation (SELECT)
  - quelles tables ?
  - quels champs ?
  - comment relier les tables entre elles ?
  - comment filtrer ? (exemple : UEs de plus de 30 heures)
  - comment trier ? Sur quels champs ? Croissant ou décroissant ?
  - comment limiter la taille des réponses
- Requête de création (INSERT)
  - ajout d'enregistrement(s) dans les tables
- Requête de modification (UPDATE)
  - modification d'enregistrement(s) déjà existant(s)


Champs
Tables
Filtres
Tris


Champs
FROM
diplomes
Tables
Filtres
Tris


Champs
FROM
diplomes
INNER JOIN
mentions ON mentions.id_diplome = diplomes.id
Tables
Filtres
Tris

Champs

Tables

```
FROM
    diplomes
INNER JOIN
    mentions ON mentions.id_diplome = diplomes.id
INNER JOIN
    enseignants ON enseignants.id = mentions.id_enseignant
INNER JOIN
    disciplines ON disciplines.id = enseignants.id_discipline
```

Filtres

Tris

```
SELECT
    diplomes.id,
    diplomes.intitule,
```

Champs

```
FROM
    diplomes
INNER JOIN
    mentions ON mentions.id_diplome = diplomes.id
INNER JOIN
    enseignants ON enseignants.id = mentions.id_enseignant
INNER JOIN
    disciplines ON disciplines.id = enseignants.id_discipline
```

Tables

Filtres

Tris

```
SELECT
    diplomes.id,
    diplomes.intitule,
    mentions.id,
    mentions.intitule,
    enseignants.id,
    enseignants.nom,
    enseignants.prenom,
    disciplines.id,
    disciplines.CNU,
    disciplines.nom_section
```

Champs

```
FROM
    diplomes
INNER JOIN
    mentions ON mentions.id_diplome = diplomes.id
INNER JOIN
    enseignants ON enseignants.id = mentions.id_enseignant
INNER JOIN
    disciplines ON disciplines.id = enseignants.id_discipline
```

Tables

Filtres

Tris

```
SELECT
    diplomes.id,
    diplomes.intitule,
    mentions.id,
    mentions.intitule,
    enseignants.id,
    enseignants.nom,
    enseignants.prenom,
    disciplines.id,
    disciplines.CNU,
    disciplines.nom_section
```

Champs

```
FROM
    diplomes
INNER JOIN
    mentions ON mentions.id_diplome = diplomes.id
INNER JOIN
    enseignants ON enseignants.id = mentions.id_enseignant
INNER JOIN
    disciplines ON disciplines.id = enseignants.id_discipline
```

Tables

```
WHERE
        disciplines.CNU = '27'
    OR disciplines.CNU = '25'
```

Filtres

Tris

```
SELECT
    diplomes.id,
    diplomes.intitule,
    mentions.id,
    mentions.intitule,
    enseignants.id,
    enseignants.nom,
    enseignants.prenom,
    disciplines.id,
    disciplines.CNU,
    disciplines.nom_section
```

Champs

```
FROM
    diplomes
INNER JOIN
    mentions ON mentions.id_diplome = diplomes.id
INNER JOIN
    enseignants ON enseignants.id = mentions.id_enseignant
INNER JOIN
    disciplines ON disciplines.id = enseignants.id_discipline
```

Tables

```
WHERE
        disciplines.CNU = '27'
    OR disciplines.CNU = '25'
```

Filtres

```
ORDER BY
    diplomes.intitule ASC,
    mentions.intitule ASC
```

Tris

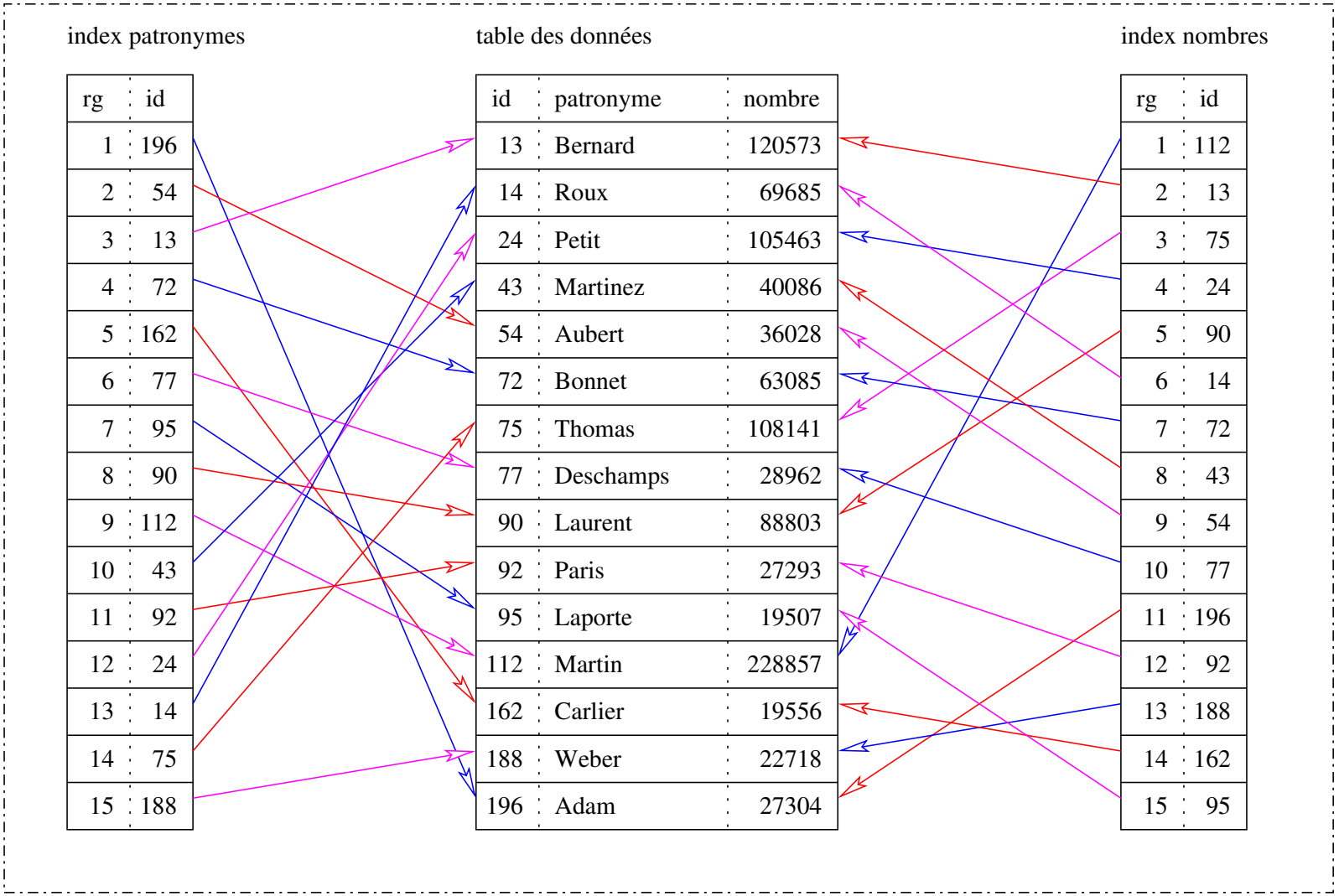
- Redondance
  - cas 1 : dupliquer une donnée (ex : éviter une recherche sur une table distante)
  - cas 2 : stocker un calcul (ex : solde d'un compte)
  - intérêt : gain de temps en lecture
  - inconvénient : perte de temps de mise à jour, risque d'incohérence
- À éviter dans la mesure du possible
- À justifier systématiquement

- Définition : code SQL stocké dans la base de données
- Exécuté sur événement
  - sur la mise à jour d'un enregistrement
  - sur un effacement
  - ...
- Exemple
  - contraintes (ex : un âge est positif)
  - redondance : un changement dans une table est automatiquement répercuté sur la donnée redondante
  - champs automatiques (date de dernière modification, nombre de modifications, ...)

Plan → A - Bases de Données

- Exemple : table non triée avec :
  - nom de personne
  - nombre d'habitants avec ce nom
- Recherche d'un nom
  - coûteux : recherche linéaire
- Solution : gérer une table triée ?
  - pourquoi pas !
- Recherche sur un nombre d'occurrences
  - trier la table ?
  - non, plus coûteux qu'une recherche linéaire
- On ne veux pas favoriser une colonne par rapport aux autres
  - solution : index

- Exemple : patronymes et population
- Tri sur des colonnes
  - celles fréquemment utilisées
  - sans modifier la table visée
    $\Rightarrow$ tables annexes (et internes au SGBDR)
- Avantages
  - recherche (select) rapide : complexité logarithmique
- Inconvénients
  - perte de place
  - modifications de la table ralenties
- Hypothèse
  - nbr consultations $>$ nbe mises à jour
  - souvent vérifiée



rg	id
1	196
2	54
3	13
4	72
5	162
6	77
7	95
8	90
9	112
10	43
11	92
12	24
13	14
14	75
15	188

id	patronyme	nombre
13	Bernard	120573
14	Roux	69685
24	Petit	105463
43	Martinez	40086
54	Aubert	36028
72	Bonnet	63085
75	Thomas	108141
77	Deschamps	28962
90	Laurent	88803
92	Paris	27293
95	Laporte	19507
112	Martin	228857
162	Carlier	19556
188	Weber	22718
196	Adam	27304

rg	id
1	112
2	13
3	75
4	24
5	90
6	14
7	72
8	43
9	54
10	77
11	196
12	92
13	188
14	162
15	95

- Utilisation de structures de données (et d'algorithmes) complexes
  - tables de hashage
  - arbres : binaires, n-aires
  - B-arbre
  - combinaisons !
- c'est un métier

Plan → A - Bases de Données

- Accès concurrents
- Conflit d'une seule requête
  - géré par le SGBD
  - transparent pour le programmeur
- Ensemble de requêtes inter-dépendantes
  - notion de verrouillage de tables
  - géré par l'utilisateur qui verrouille les tables
  - géré par le SGBD qui bloque les accès à ces tables pour les autres utilisateurs
- Exemple : mise à jour d'un solde bancaire
  - lecture du solde existant dans la table
  - calcul du nouveau solde
  - modification de la table avec le nouveau solde

- Architecture logicielle complexe
  - installation d'un serveur
  - petit système d'exploitation
  - lourdeur de fonctionnement
  - exemples : Oracle, MySQL/MariaDB, PostgreSQL, ...
- Et si mono-utilisateur ?
  - architecture simplifiée
  - gain de performance
  - exemple : SQLITE
  - exemples de clients : Firefox, gestion des contacts d'un mobile, ...

- Pouvoir annuler une opération ... ou plusieurs !
- Solution : garder une sauvegarde du ou des états antérieurs de la base de données
- Coût
  - éventuellement prohibitif (exemple : vider une table)
  - restrictions : undo à un niveau, quelques opérations non réversibles
- Autre solution
  - à la demande
    - début commandes réversibles
    - quelques requêtes
    - éventuellement décision d'annuler
    - quelques requêtes
    - fin commandes réversibles
- Solution connexes
  - dumps réguliers de la base (i.e. gestion des sauvegardes)

- Que faire si le serveur plante au milieu d'une requête ?
  - une seule requête, exemple : mise à jour d'un enregistrement
  - ensemble de requêtes, exemple : mise à jour du solde bancaire
- Solution : système de journalisation
  - les différentes étapes d'avancement sont consignées dans un journal
  - les modifications sont effectuées dans une table clone
  - le résultat final est recopié dans la table "officielle"
  - requêtes multiples liées : cf. undo
  - (note : les partitions d'un disque peuvent être journalisées)
- Reprise après panne
  - rejouer le scénario journalisé
- Défaillance d'un disque
  - redondance de données (RAID)
  - sauvegardes incrémentale, totale.

- Principe
  - les tables sont réparties sur plusieurs machines
- Pourquoi ?
  - trop grande masse de données
  - volonté de traitement parallèle
  - exemple : référencements de Google
- Inconvénients
  - SGBD très complexe
  - comment gérer la cohérence des données, les pannes, ... ?
- Stratégie de répartition
  - 1) une table par machine ?
  - 2) chaque table est fragmentée sur plusieurs machines ?
- Choix
  - solution 2
  - intérêt : interrogation d'une table traitée en parallèle
  - difficulté : agglomérer toutes les réponses partielles

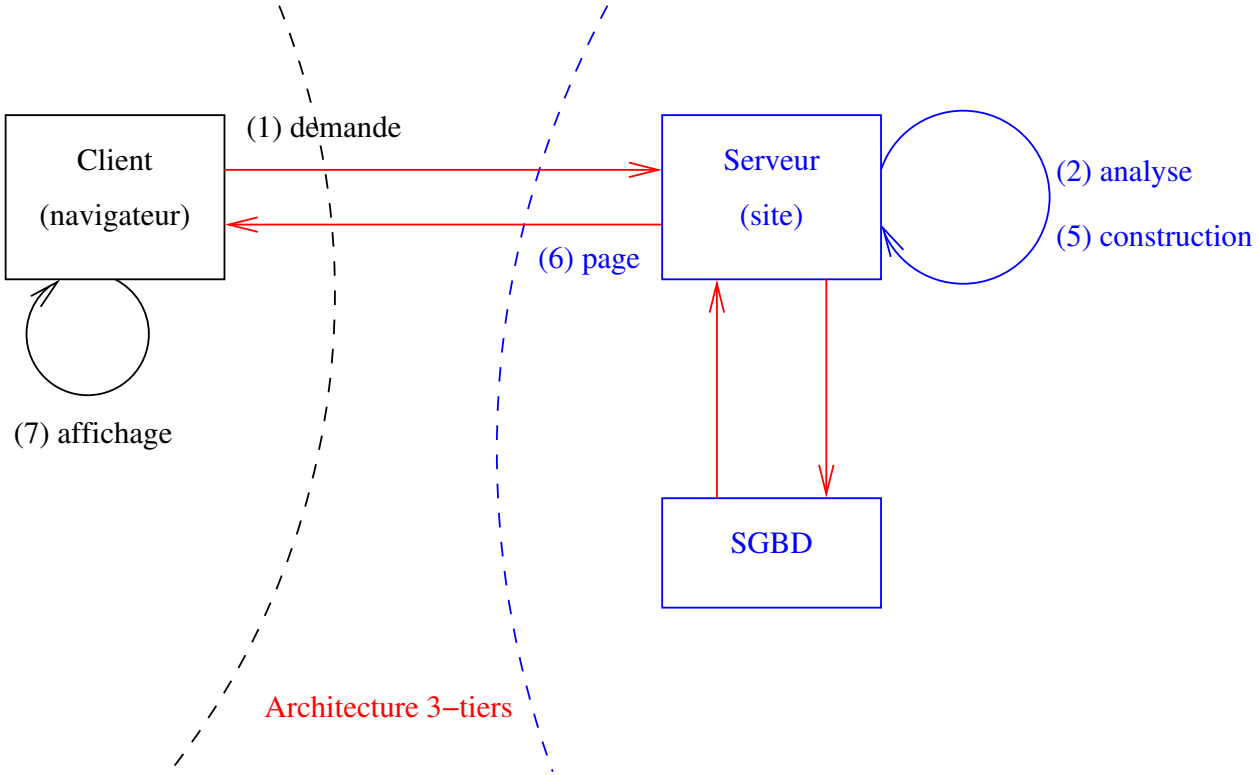
  → stratégie d'Amazon pour répartir les livres, disques durs multi-plateaux

Plan → A - Bases de Données

- vues (tables virtuelles)
- notion de trigger et procédure stockée
- optimisation des requêtes
- BD NoSQL (Not only SQL)
- BD orientées objets
- moteurs de stockage
- administration d'une BD
- Data mining $\rightarrow$ intelligence artificielle
- ...

Plan → A - Bases de Données

- Architecture n-tiers
  - les services sont répartis sur plusieurs machines



- Serveur de la base attaquable isolément
  - non visible de l'extérieur (IP privée)

- Empêcher les intrusions
  - fermer à clé, ne pas noter le mot de passe sur un post-it, ...
  - pare-feu
  - mises à jour des logiciels/systèmes
  - se protéger des injections SQL (ou autres : javascript, ...)
  - protocole https
  - limiter l'accès au serveur à des machines identifiées
- Limiter les dégats
  - mots de passe stockés en crypté
  - ne pas stocker les numéros de CB
  - ... ou alors sur un second serveur protégé (notion de double pare-feu)