Base de donneés

Alain Taquet

September 14, 2017

Infos

Infos générales

Organisation du cours

- Mail : alain.taquet at univ-lille3.fr
- Lien du cours : grappa.univ-lille3.fr/~taquet/
- 36h TDs
- 6 séances sur access
- 6 sur sql
- Contrôles : 2 contrôles

Bases de données

Système d'information

BD et Web

Site dynamique

- l'utilisateur demande une page, la demande est envoyée au serveur qui execute un programme.
- les données utiles sont extraites de la BD vers le serveur.
- un document html est construit et envoyé sur le reseau.
- le document est affiché.

Séparation entre contenu, structure et présentation.

Vente

Une entreprise de vente

- contenu : clients, produits, prix, fournisseurs, facturation, livraison ...
- applications : gestion de l'entreprise.
- manipulations : services de l'entreprise, commandes Web.
- consultation : entreprise, suivi de commandes ...

BD et SGBD

Définition

Une **BD** est un ensemble d'informations de grande taille, structuré, mémorisé sur un support permanent.
Un **SGBD** : (systeme de gestion de BD) est un outil permettant :

- de structurer
- d'insérer
- de rechercher des données spécifiques

Fonctions d'un SGBD

- Persistance
- Langage
- Partage, Fiabilité, Sécurité des données
- Gestion du/des disques
- Indépendance logique/physique

Le modèle relationnel

Domaine / Relation

- Un **domaine** est l'ensemble des valeurs que peut prendre une donnée
- Une **relation** (table) est un sous ensemble du produit des domaines
- Un **attribut** est une colonne de relation caractérisée par un nom
- Une **clé** est constituée d'un ou plusieurs attributs

Formes normales

1FN : à toute valeur de la clé correspond au plus une valeur des attributs

2FN : 1FN + un attribut ne dépend pas d'une partie de la clé

procom(**numcommande, numproduit**, quantite, *prixproduit*)

- prixproduit ne déprend que de numproduit

3FN : 2FN + les dépendances liant la clé aux attributs sont directes

produits(**numproduit**, . . . , numfournisseur, *nomfournisseur*)

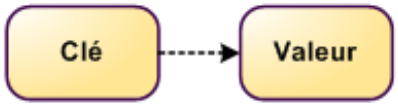
- nomfournisseur dépend de numfournisseur et pas de numproduit

Opérations

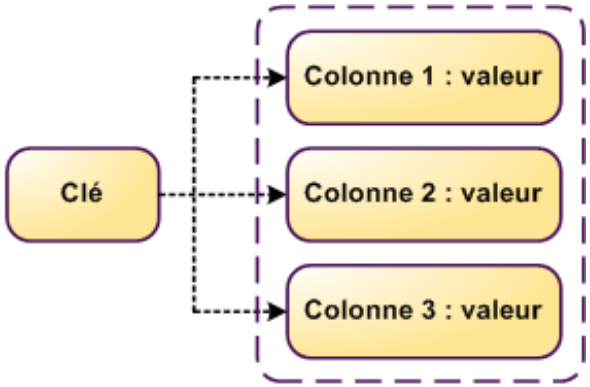
- produit
- projection (selection verticale)
- selection (selection horizontale)
- différence
- union

Les bases NoSQL

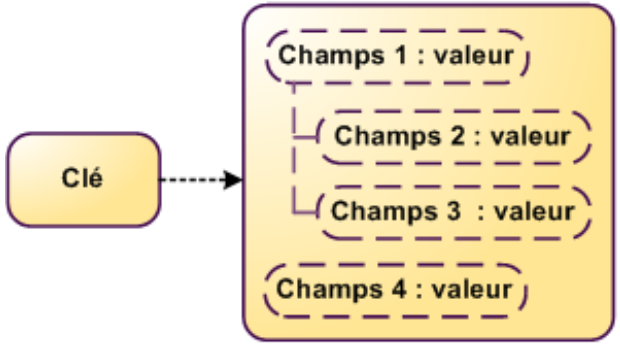
Les familles



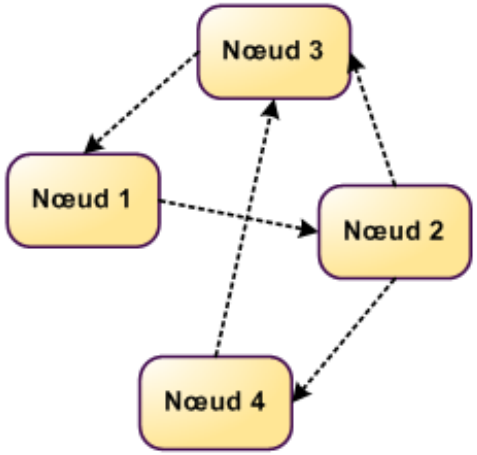
BDD Clé-Valeur



BDD Orientée colonnes



BDD Orientée document



BDD Orientée graphe

Redis

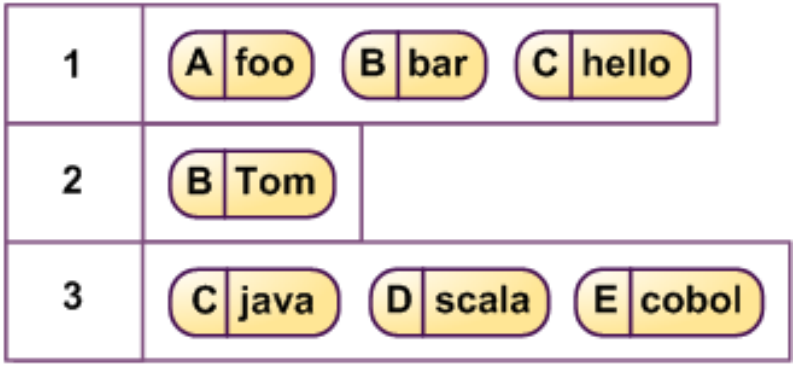
- Les bases **clef/valeur**, permettent de stocker des données sous forme d'un couple clef/valeur où la valeur peut être une chaine de caractère, un entier ou un objet sérialisé.
- LinkedIn, réseau social professionnel 400 millions de membres

Cassandra / HBase

- Les bases orientées **colonnes** ressemblent aux bases de données relationnelles : les données sont sauvegardées sous forme de lignes avec un nombre de colonnes qui peut varier d'une ligne à l'autre.
- Facebook (1 milliard par jour), Twitter, Spotify

	A	B	C	D	E
1	foo	bar	hello		
2		Tom			
3			java	scala	cobol

Organisation d'une table dans une BDD relationnelle



Organisation d'une table dans une BDD orientée colonnes

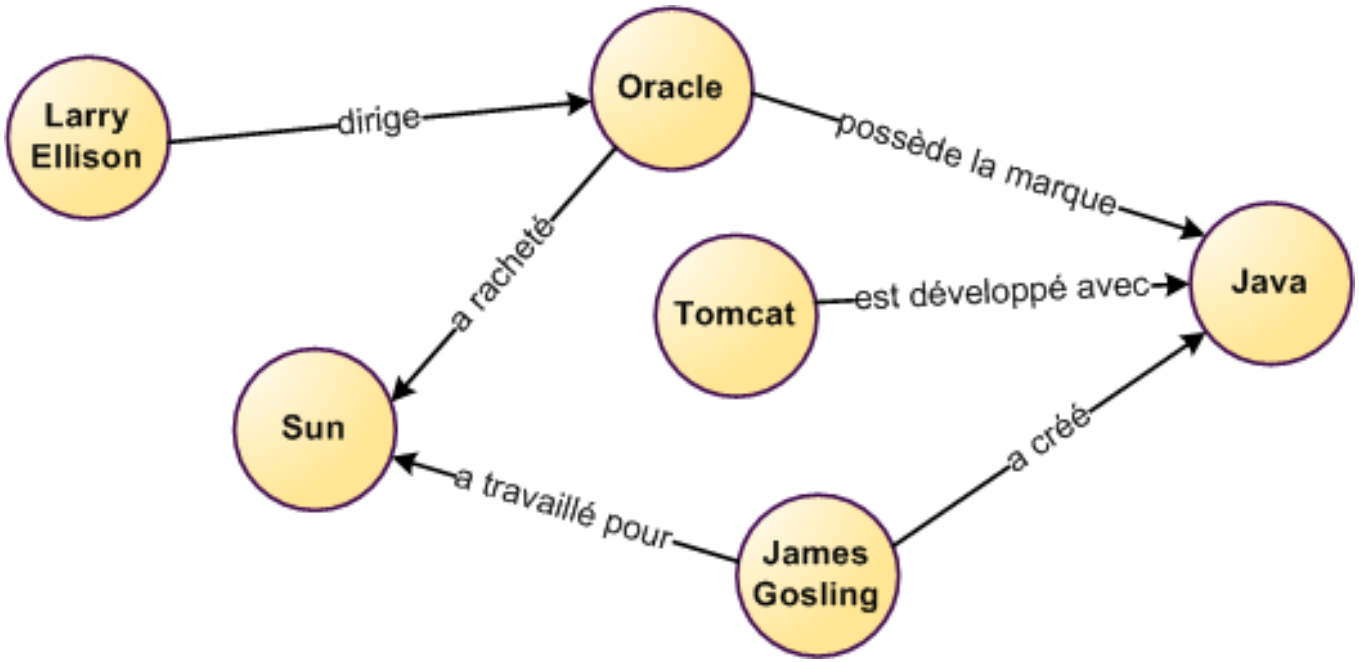
MongoDB

- Les bases orientées **document** représentent les données sous forme d'objet XML ou JSON. L'avantage est de pouvoir récupérer simplement des informations structurées de manière hiérarchique.
- eBay, foursquare

```
{"titre": "Mon post",
 "etoiles": 5,
 "commentaires": [{
    "nom": "John Doe",
    "contenu": "Super post"}, ....]
}
```

Neo4j

- Les bases orientées **graphe** représentent les données sous forme de noeud et de relation.
- Recommandation, Géo-Spatial : Walmart, LinkedIn



Remarques

- NoSQL : Not Only SQL
- Pourquoi ?
- rechercher le nombre de commentaires par utilisateur ?
- besoins spécifiques, en général les 2 systemes.

Modèle relationnel

Principe

Une enquête

Vous réalisez une enquête, vous devez stocker:

- les questions
- les réponses possibles
- la répartition des réponses en facteurs
- les réponses des sujets
- etc ..

Comment faire ?

Solution

Vous devez :

- identifier ce qui “existe” (entités) : sujets / questions etc . . .
- établir des relations entre les entités et leur type
- en déduire une implémentation

Plan :

- on voit d’abord les types de relation (1-n, n-n)
- on l’applique ensuite sur une base (médical)

Relations 1-n (père-fils)

Enoncé

Comment modéliser dans une base de donnée l'énoncé suivant:

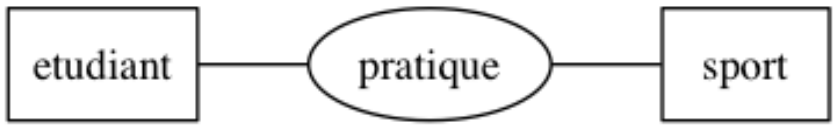
> Des étudiants pratiquent un et un seul sport

- Il faut représenter (enregistrer) les étudiants et les sports. On les appelle des **entités**.
- Le plus difficile est de représenter la **relation** entre les objets : "pratique" (le verbe dans la phrase)
  - L'étudiant E1 pratique le sport S1
  - L'étudiant E2 pratique également le sport S1

MCD (modèle conceptuel de données)

On commence par représenter graphiquement les entités et les relations.

- les entités sont représentés par des rectangles
- les relations (associations) par des ellipses

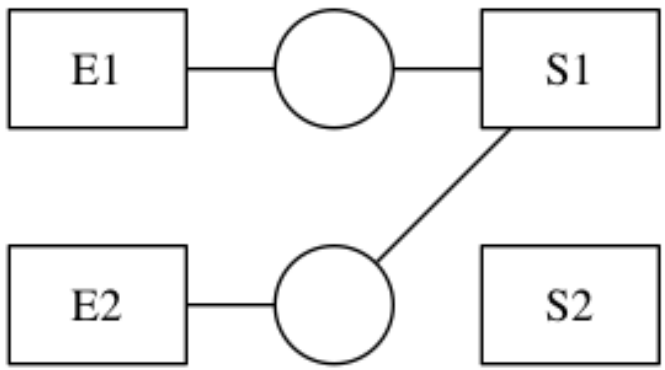


On annote ensuite le schéma avec le nombre d'associations possible qu'on appelle les cardinalités

Diagramme d'instances

Pour déterminer les cardinalités on prend un exemple, en général 2 ou 3 instances de chaque entité et on fait le lien entre les instances

- L'étudiant E1 pratique le sport S1
- L'étudiant E2 pratique également le sport S1
- Personne ne pratique le sport S2



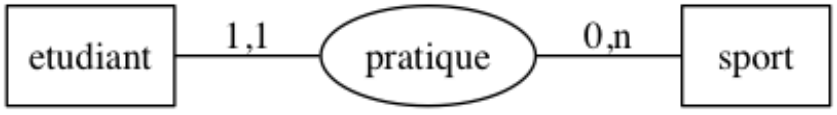
Cardinalités

On regarde ensuite le nombre d'associations (de traits) qu'il va falloir représenter :

- de étudiants vers sports :
  - un étudiant pratique au minimum 1 sport
  - un étudiant pratique au maximum 1 sport
  - la cardinalité de ce coté est 1,1 (min, max)
- de sports vers étudiants :
  - un sport est pratiqué par au minimum 0 étudiant
  - un sport est pratiqué par au maximum n étudiants
  - la cardinalité de ce coté est 0,n (min, max)

MCD annoté

On place les cardinalités sur le schéma :



- Les cardinalités minimales sont importantes pour les requêtes : si on fait une jointure entre les étudiants et les sports, S2 n'apparaitra pas
- Les cardinalités maximales sont utilisées pour la conception de la base (le MLD)

On dit qu'il s'agit d'une relation 1-n (max de chaque côté)

MLD (modèle logique de données) - entités

Il faut maintenant implémenter la base(créer en SQL ou dans access) donc implémenter les **entités** et la **relation**

Implémentation des entités

On crée une table pour chaque entité : une table étudiants et une table sports
On représente les tables avec leur nom et les différents champs entre parenthèses. La clé primaire (identifiant) est ici en gras

- étudiants (**numEtu**, nom, prénom)
- sports (**numSport**, intitulé)

MLD - relation

Implémentation de la relation

Les cardinalités maximales donnent le nombre d'associations qu'il va falloir représenter :

- du côté des étudiants (1) il n'y a qu'une seule référence à un sport
- du côté des sports (n) il y a éventuellement plusieurs références aux étudiants

On crée une nouvelle colonne du côté 1 et on y place la référence au côté n (une seule valeur)

La table etudiants

La table étudiants devient : étudiants (**numEtu**, nom, prénom, *refSport*)

numEtu	nom	prenom	*refSport*
E1	Dupont	x	S1
E2	Dupond	y	S1

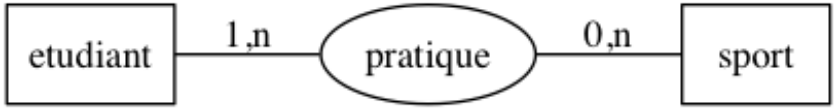
refSport est une clé étrangère et référence numSport

- E1 ne peut apparaitre qu'une seule fois (clé primaire)
- S1 apparait plusieurs fois (clé étrangère)

Relations n-n (maillées)

MCD (modèle conceptuel de données)

Des étudiants pratiquent un ou plusieurs sports

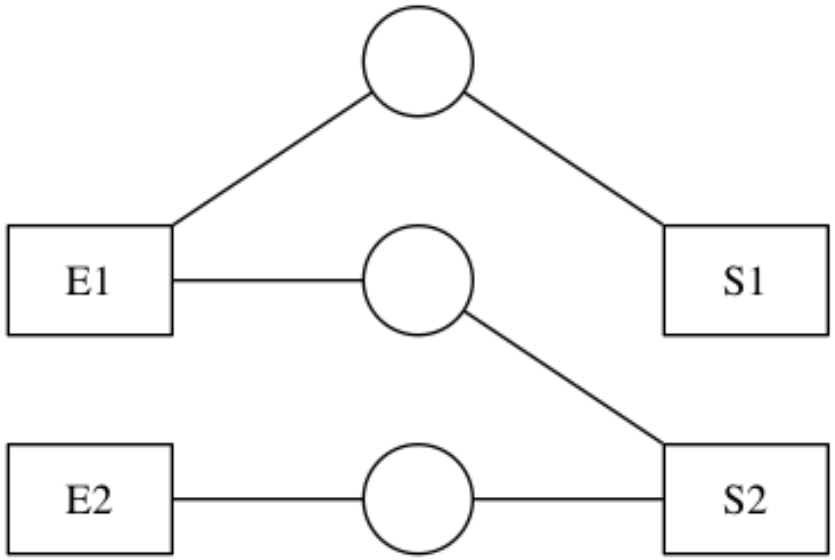


- du côté étudiants la cardinalité est 1,n
- du côté sports elle est 0,n

On dit qu'il s'agit d'une relation maillée ou n-n (max de chaque côté)

Diagramme d'instances

Comment représenter le fait que E1 pratique S1 **et** S2 ?
Il y a maintenant plusieurs références à placer de chaque côté :



MLD

Les tables représentant les entités ne changent pas :

- étudiants (**numEtu**, nom, prénom)
- sports (**numSport**, intitulé)

On crée une nouvelle table etuSport pour représenter la relation.

On place dans cette table les clés primaires des entités en relation :

- etuSport (*refEtu*, *refSport*)

refEtu et *refSport* sont indépendemment des clés étrangères.

La table etuSport

- etuSport (**refEtu, refSport**)

La clé primaire de la nouvelle table est constituée de l'ensemble des clés étrangères

refEtu	**refSport**
E1	S1
E1	S2
E2	S2

- refEtu contient plusieurs fois E1
- refSport contient plusieurs fois S2
- le couple (E1, S2) ne peut apparaitre qu'une seule fois

Exemple : Base Médical

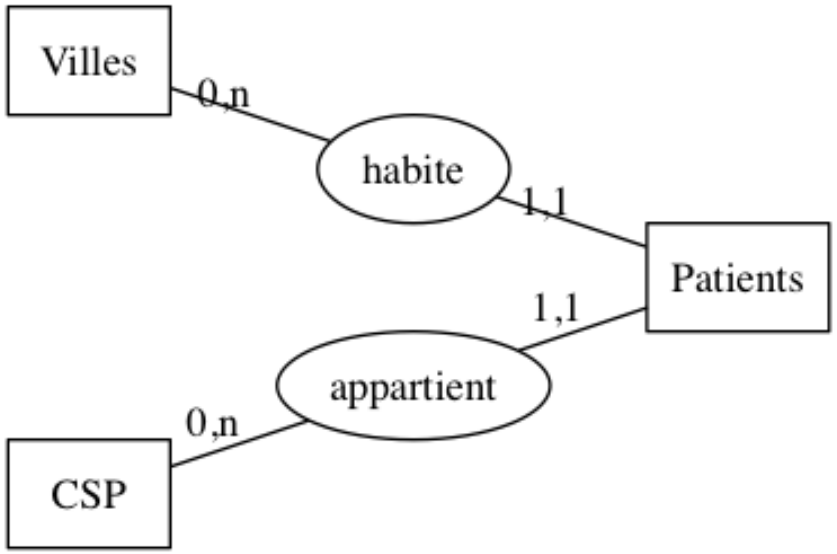
MCD

La base

- Des patients habitent une et une seule ville et ont une et une seule catégorie socio-professionnelle (CSP).
- Ces patients passent des tests dans des centres de contrôle.
- Les tests donnent lieu à un résultat +/-

Villes et CSP

Les relations 1-n entre patients et villes / CSP sont évidentes



Passation d'un test

Les patients passent des tests dans des centres de contrôle. La relation se fait entre les 3 entités :

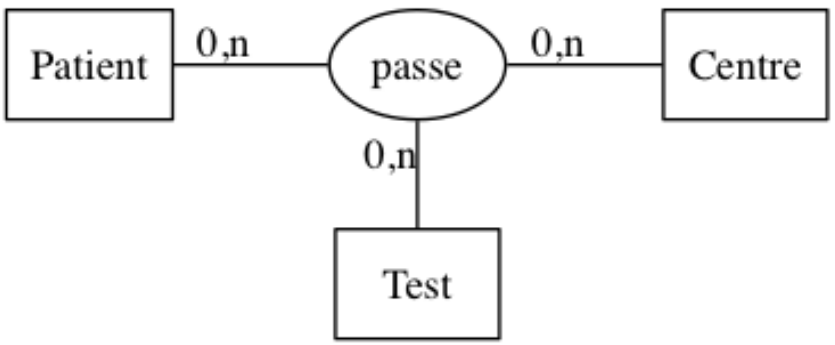
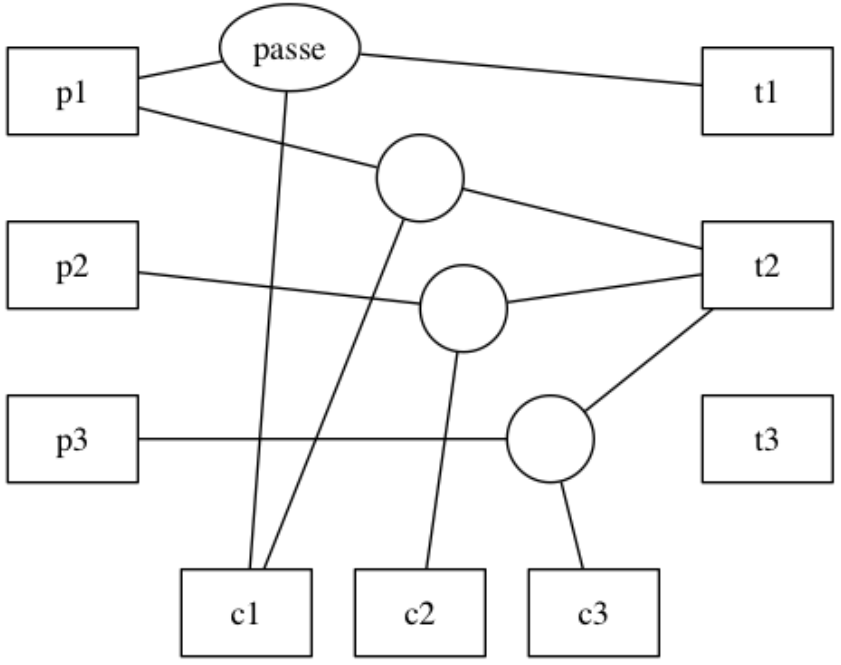


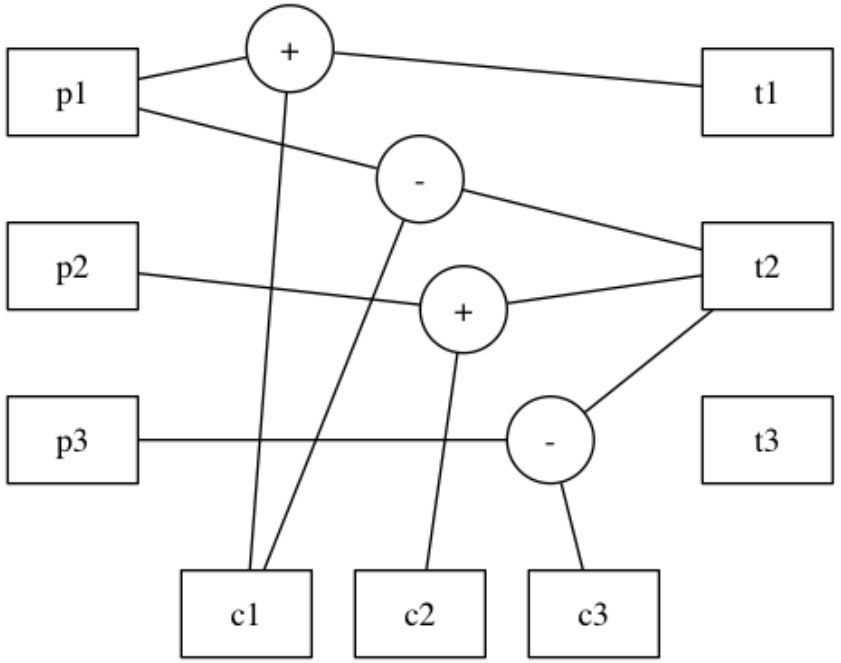
Diagramme d'instances

- Une association particulière est déterminée par l'ensemble des 3 clés primaires des entités. Par exemple (p1,c1,t1)
- Ou placer le résultat des tests ?



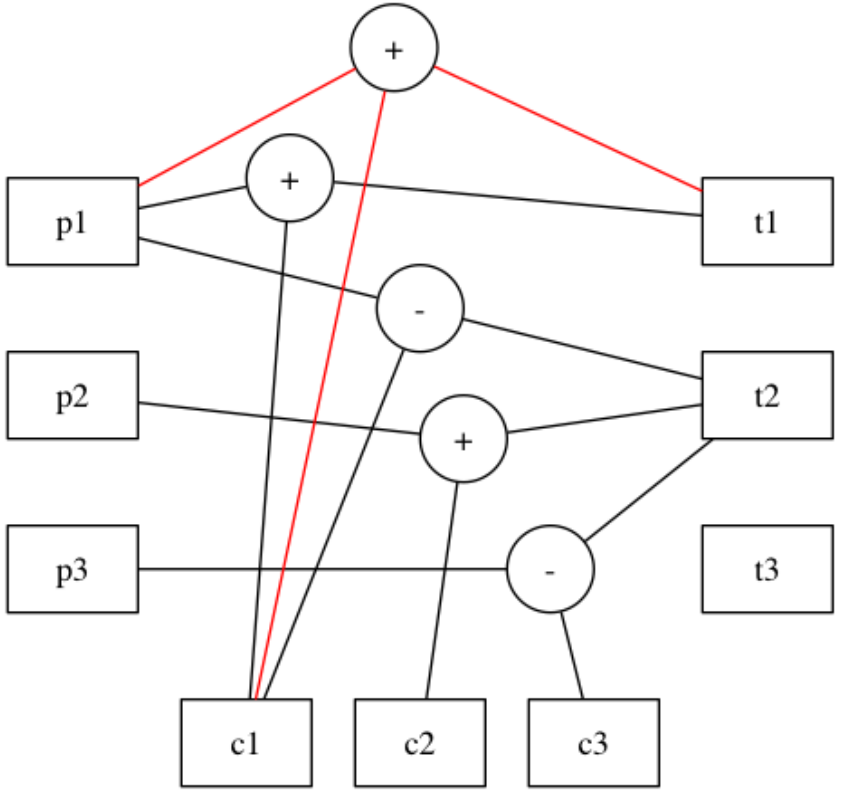
Propriétés des relations

- Le résultat du test ne peut pas être placée dans les entités
- C'est une propriété de la relation



Problème

Le patient P1 doit repasser le test T1 dans le même centre et obtient le même résultat. Les 2 associations sont déterminées par les mêmes tuples (p1,c1,t1) !



Solution

Il faut ajouter que la passation d'un test s'effectue à une date donnée

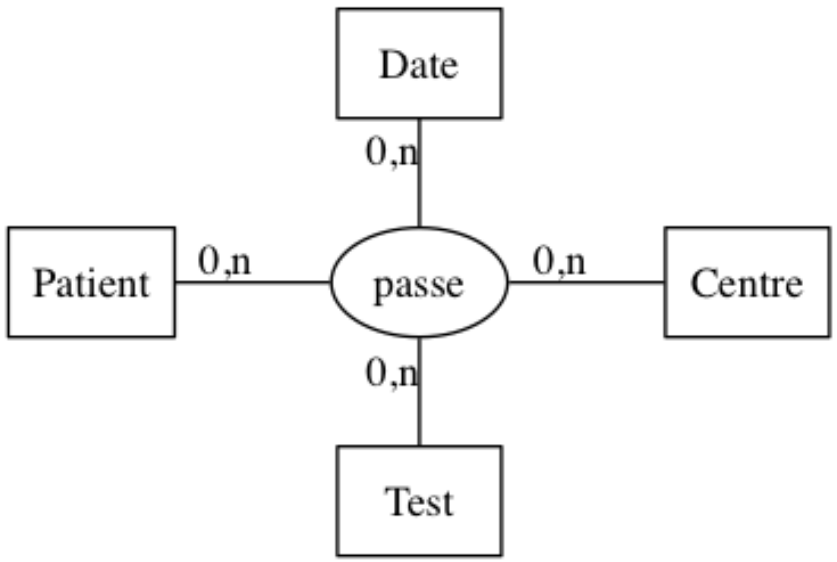
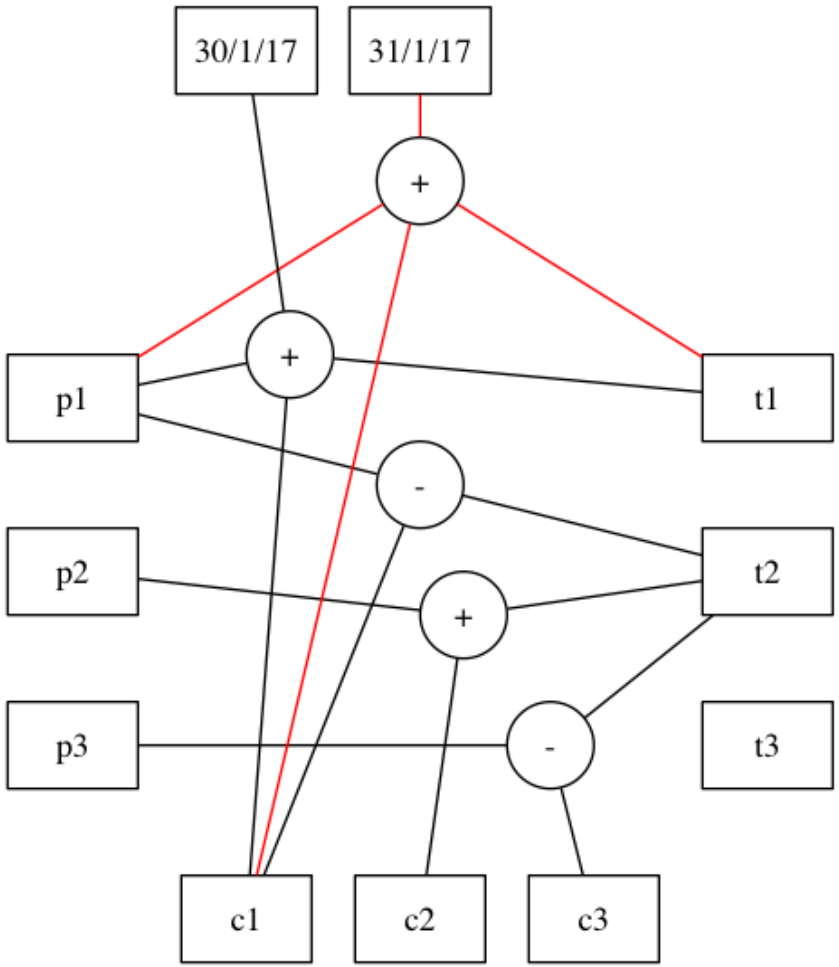


Diagramme d'instance final

La clé de la date étant la date on ne crée pas de nouvelle table représentant les instances de Date



MLD

Les tables

- Les entités
  - postal (**codePostal**, ville)
  - catSocio (**numCsp**, libelleCsp)
  - patients (**numeroSS**, nom, . . . , *codePostal*, *numCsp*)
  - tests (**codeTest**, intitulé, tarif)
  - centres (**codeCentre**, laboratoire)
- La relation
  - passation (**refPatient, refTest, refCentre, date**, resultat)

La table passation

refPatient	**refTest**	**refCentre**	**date**	résultat
P1	T1	C1	30/1/17	+
P1	T1	C1	31/1/17	+
P1	T2	C1	x	-
P2	T2	C2	x	+
P3	T2	C3	x	-

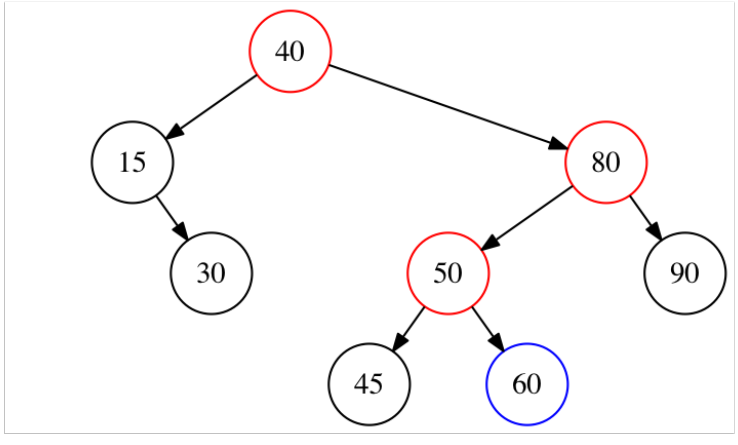
L'ensemble des 4 clés étrangères constitue la clé primaire, résultat est une propriété et ne fait pas partie de la clé

Champs indexés

Arbre équilibré

- Lorsqu'on indexe un champ la BD crée un arbre qui le représente
- On accepte une différence de +/- 1 entre les feuilles de l'arbre
- La clé primaire est toujours indexée

- à partir d'un noeud :
- ce qui est $<$ est à gauche
- ce qui est $>$ à droite

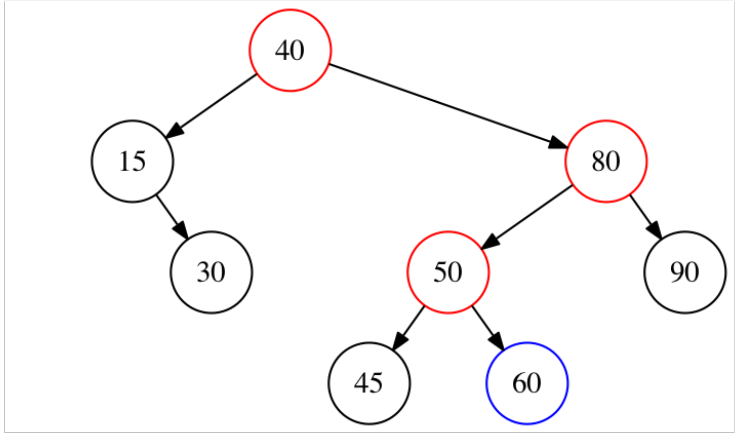


Recherche d'une valeur

à chaque comparaison on divise le nombre de recherche par 2

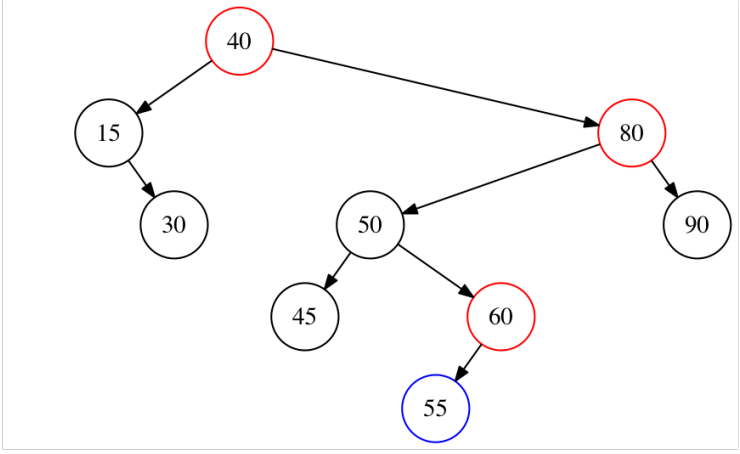
On recherche 60 :

- on part de 40 (la racine)
- on suit le chemin :
  - 40 -> 80
  - 80 -> 50
  - 50 -> 60

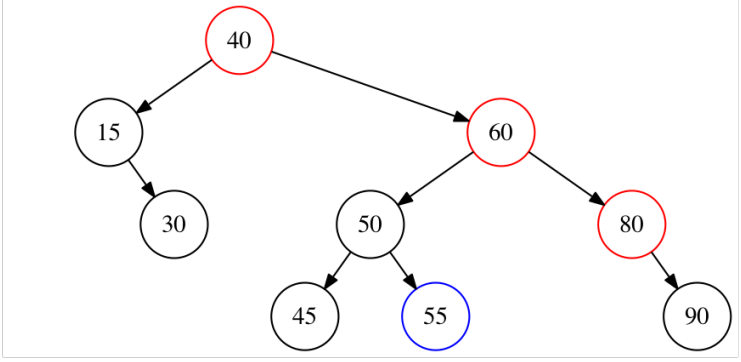


Insertion d'une valeur

- Lorsqu'on insère 55 l'arbre n'est plus équilibré
- Il faut le restructurer donc déplacer des sous arbres



Non équilibré



Restructuration

Conclusion

- Les recherches dans un champ indexé sont rapides
- Les insertions et suppressions demandent beaucoup de calcul
- On indexe uniquement les champs sur lesquels on fait beaucoup de recherches

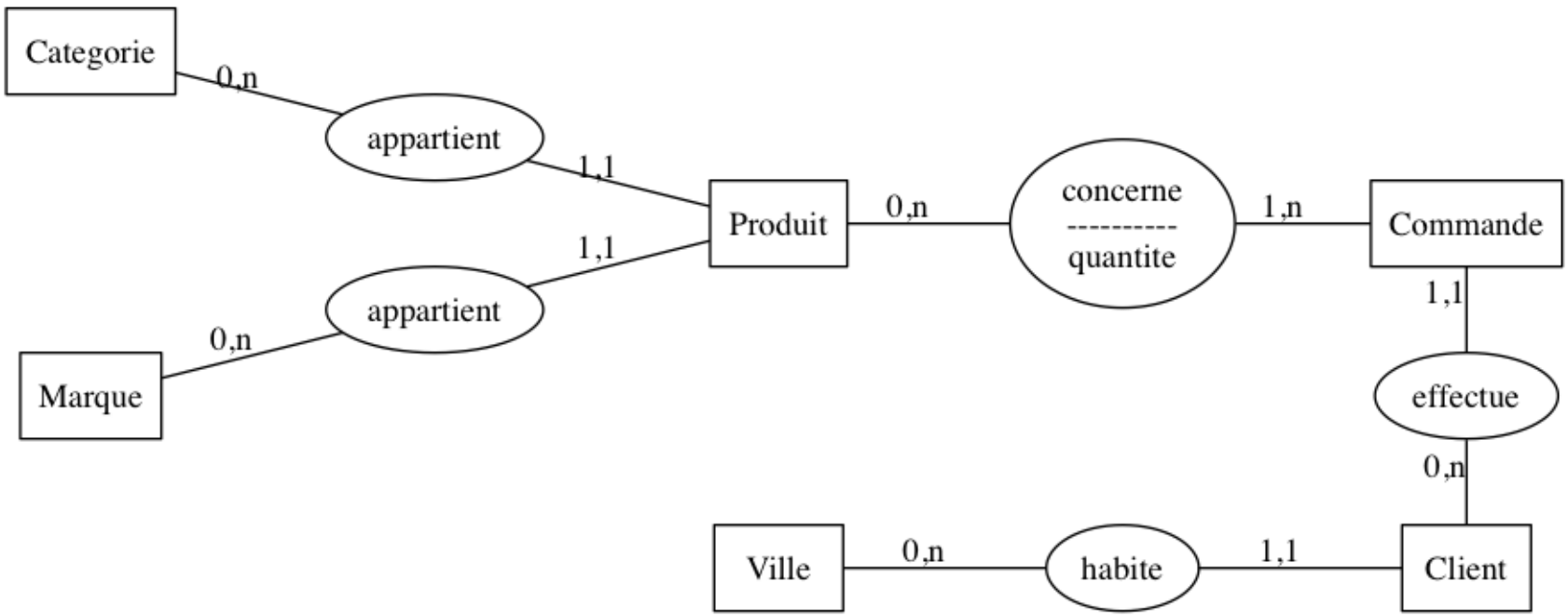
La base du TD

La base Sport

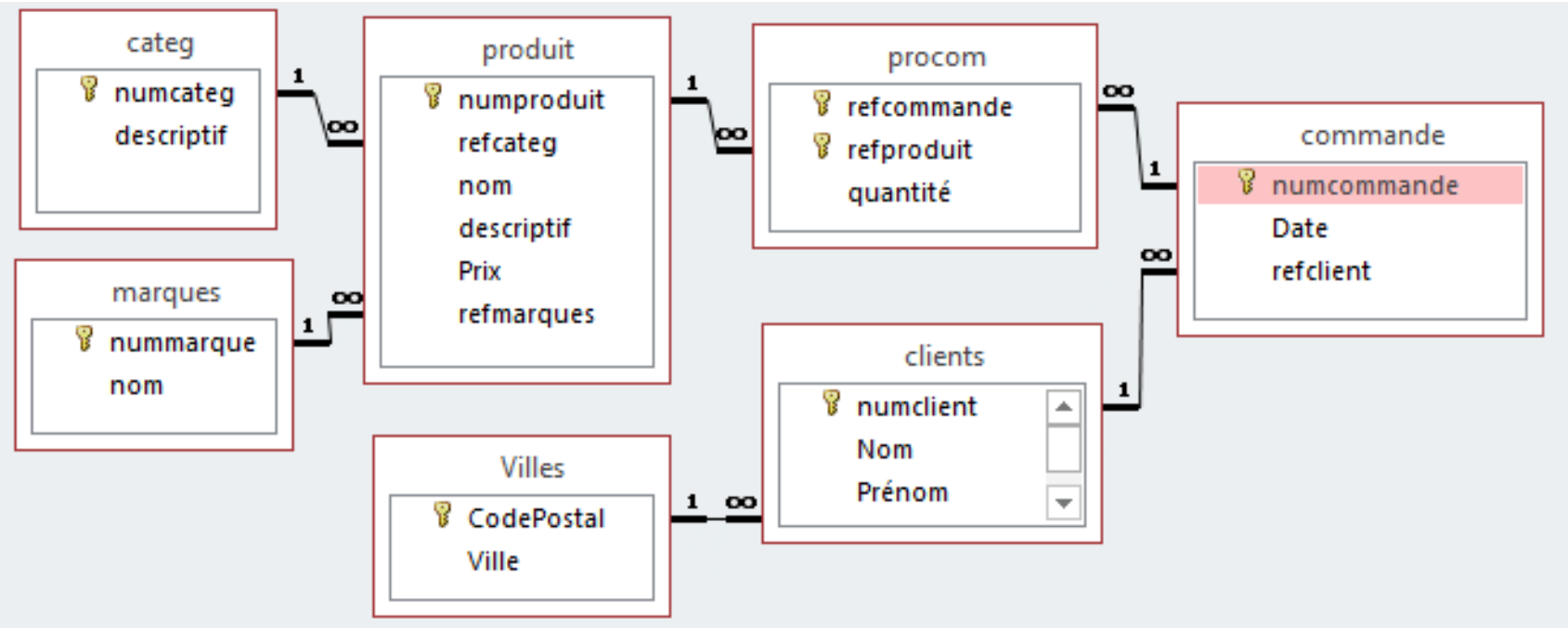
Description

- Un magasin de sport souhaite informatiser les commandes effectuées par ses clients.
- Les produits ont un code, un nom, un descriptif et un prix et sont classés selon des marques et des catégories.
- Les clients ont un nom, un prénom et habitent une ville référencée à partir du code postal.
- Les clients effectuent des commandes d'un ou plusieurs produits en quantité variable.
- Plusieurs commandes peuvent être faite par le même client à une même date.

MCD


Categorie
0,n
appartient
1,1
Produit
0,n
concerne
quantite
1,n
Commande
1,1
appartient
1,1
0,n
Marque
effectue
0,n
Ville
0,n
habite
1,1
Client

MLD


categ
numcateg
descriptif
produit
numproduit
refcateg
nom
descriptif
Prix
refmarques
procom
refcommande
refproduit
quantité
commande
numcommande
Date
refclient
marques
nummarque
nom
clients
numclient
Nom
Prénom
Villes
CodePostal
Ville
1
∞