



HAL
open science

Cours de bases de données - Ecole doctorale

Sylvie Damy

► **To cite this version:**

Sylvie Damy. Cours de bases de données - Ecole doctorale. Doctorat. Introduction aux bases de données relationnelles, Besançon, France. 2024, pp.275. hal-04573261

HAL Id: hal-04573261

<https://univ-fcomte.hal.science/hal-04573261v1>

Submitted on 16 May 2024

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



Distributed under a Creative Commons Attribution 4.0 International License

Introduction aux Bases de données

Ecole doctorale - Formations permanentes - Année 2023-2024

Sylvie DAMY

CHRONO
ENVIRONNEMENT



UBFC
UNIVERSITÉ
BOURGOGNE FRANCHE-COMTÉ



Objectifs du cours

Utilisateurs avertis des bases de données

Etre capable de :

- concevoir un modèle de BD « simple »
- créer la base de données
- intégrer des données
- interroger la base de données : requêtes simples ...
requêtes de comptage

Plan du cours



Qu'est-ce qu'une base de données ?



Modélisation - MCD



Modèle relationnel



SQL : Langage d'interrogation

Organisation



Séance 1 : 07 février 2024
MCD + Modélisation
relationnelle



Séance 3 : 04 mars 2024
SQL + première requêtes



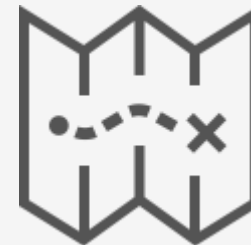
Séance 5 : 20 mars 2024
Requêtes



Séance 2 : 21 février 2024
Travail sur vos données
→ MCD



Séance 4 : 13 mars 2024
Requêtes



Salle 110K

Pour le 15 février :

Envoi de vos données ou confirmation que vous viendrez avec des données sur lesquelles nous pourrions travailler

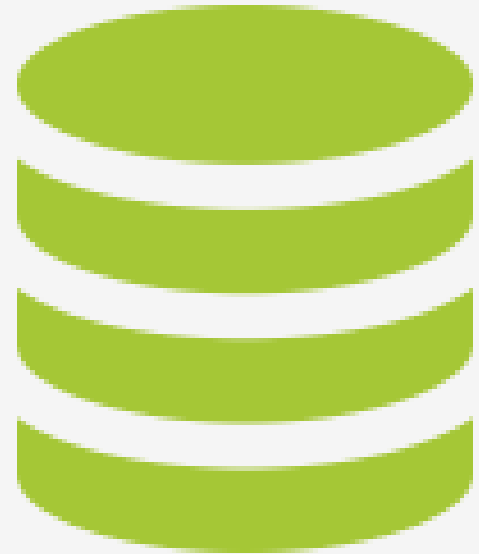


sylvie.damy@univ-fcomte.fr

Partie 1

Qu'est-ce qu'une base de données

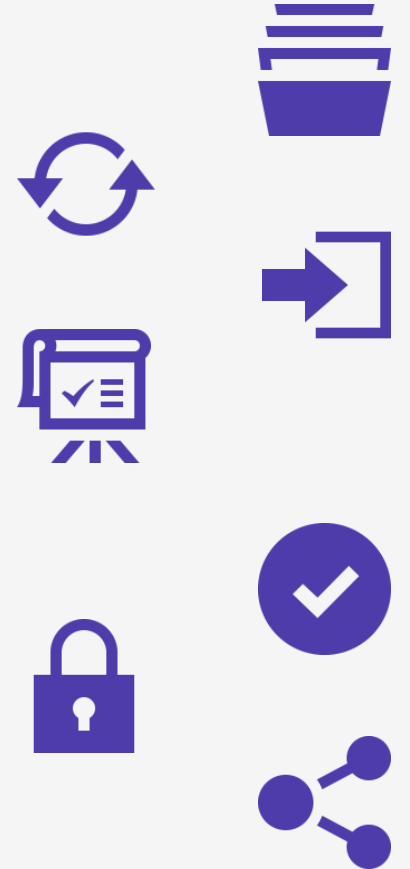
?



Pourquoi des bases de données ?

- Stockage permanent des données
- Mise à jour des données
- Accès facile
- Présentation des données

- Qualité des données
- Sécurité
- Partage



Bases de données : définitions

Ensemble d'informations modélisant les objets d'une partie du monde réel et servant de support à une application informatique

Collection **organisée, interrogeable et persistante de données** ou d'informations, concernant un thème donné pour l'entreprise



Bases de données : définitions

D'un point de vue plus informatique

Ensemble de fichiers comprenant un certain nombre d'articles formés chacun d'un type de données, avec des liens entre ces articles

Ensemble d'opérateurs permettant la recherche, le tri, la fusion ou toute autre opération de même type

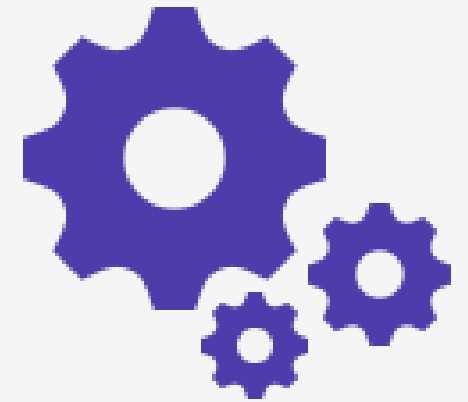
Ensemble de données et de métadonnées (données qui décrivent les données → schéma de la base)

SGBD : Système de Gestion des Bases de Données

Outil principal de gestion des bases de données

Ensemble de logiciels fournissant un environnement fournissant des applications pour :

- décrire,
- mémoriser,
- manipuler,
- traiter des ensembles de données,
- en contrôler l'accès,
- partager ces données,
- gérer des pannes, ...



Un peu de vocabulaire

Base de données

Collection organisée de données



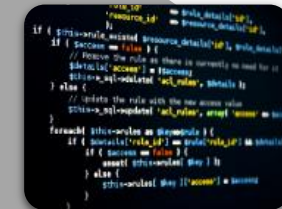
Système de gestion de base de données

Logiciel qui gère et permet l'accès aux données de la base



Application de base de données

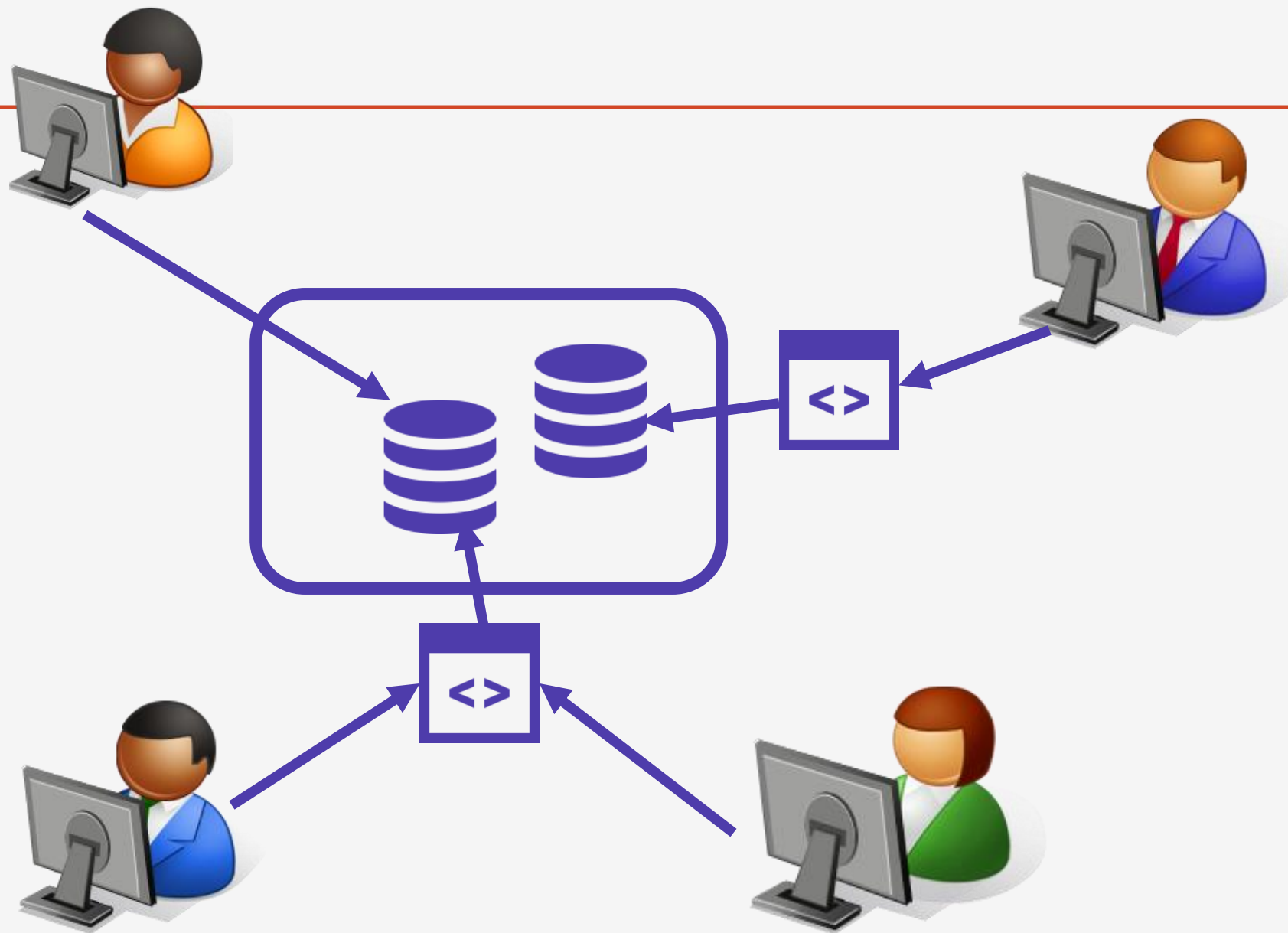
Programme qui interagit avec la base de données à certains moments de son exécution en lui adressant des requêtes (SQL)



Système d'information

Ensemble formé par la ou les bases de données, le SGBD et les applications de base de données liées à cette base de données





Votre utilisation des Bases de Données



BD versus feuille de calcul

	Base de Données	Feuille de calcul
<i>Organisation des données</i>	Liste, formulaire	Liste
<i>Qualité des données</i>	Contrôle de la cohérence	
<i>Volume d'information</i>	Non limité	Limites des tableurs
<i>Nature des informations</i>	Texte, numérique, image, fichier, ...	Chaine de caractères, numérique, booléen

BD versus feuille de calcul

	Bases de Données	Feuille de calcul
<i>Recherche d'information</i>	Requêtes → SQL	Tris
<i>Partage des informations</i>	Plusieurs utilisateurs	Un seul utilisateur
<i>Sécurité</i>	Confidentialité des données Profil d'utilisateur	

Rôles dans les bases de données

Différents types de personnes peuvent manipuler une base de données :

- Administrateurs
- Concepteurs
- Développeurs
- Utilisateurs

Rôles dans les bases de données : **Administrateurs**

Administrateur de données ou DA

- connaissance des données disponibles,
- acquisition et échange des données,
- animation du dispositif autour de la base de données.

Administrateur de bases de données ou DBA

- intégrité des données,
- sécurité et performance,
- aide au développement et au test,
- recouvrement de données et gestion des pannes,
- validation et conseil,
- migration et mise à jour



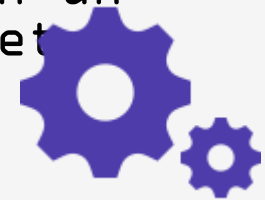
Rôles dans les bases de données : **Concepteurs**

Concepteurs logiques

- s'intéressent à l'ensemble de l'organisation à mettre en place, et en particulier à l'identification des données (entités, attributs, associations, contraintes).
- travaillent avec les utilisateurs futurs de la base de données.

Concepteurs physiques

- traitent le passage du modèle logique de données vers un modèle physique. Ils représentent le schéma logique en un ensemble de relations et de contraintes d'intégrité, et s'intéressent à la sécurité des données.
- travail qui dépend fortement du SGBD choisi



Rôles dans les bases de données

Développeurs d'applications



- réalisent des applications de bases de données qui permettent aux utilisateurs finaux d'utiliser/d'interagir avec la base de données.

Utilisateurs



- clients de la base de données. Certains ne travaillent qu'à travers le biais des applications de base de données et n'ont aucune connaissance dans le domaine des bases de données.
- autres utilisateurs, plus expérimentés, peuvent écrire leurs propres requêtes SQL pour interroger la base de données.

Rôles dans les bases de données

Rôle /Personne	Vous – Chercheur	Informaticien - Modélisation	Informaticien - Développeur	Informaticien – Administrateur
Administrateur de données	✓			
Administrateur de BD			✓	✓
Concepteurs logiques	✓	✓		
Concepteurs physiques		✓	✓	
Développeurs d'applications			✓	
Utilisateur simple	✓			
Utilisateur averti	✓			

Exemple de manipulation de bases de données



Utilisateur simple

The screenshot shows a web interface for a research data portal. At the top, there are logos for the French Republic, DAT@UBFC, and UBFC (Université Bourgogne Franche-Comté). A search bar is present with the text 'Rechercher'. Below the search bar, there are navigation links: 'Site web', 'À propos', 'Contact', 'Aide', and 'Fr En'. The main content area displays search results for '597 résultats'. The first result is titled 'Dimensionnement optimal d'une plateforme cloud distribuée autour de la terre'. It includes a small image of a globe, a list of disciplines (informatics, materials, and operational research), a collection date (since April 2022), and creators (Miguel Vasconcelos, Daniel Cordeiro, Georges Da Costa, Fanny Dufossé). The second result is 'LASPI: Détection et diagnostic des défauts de boîte de vitesses', featuring an image of an engine, disciplines in electrical and industrial engineering, a collection date of May 17, 2017, and creators Moncef Soualhi, Abdenour Soualhi, Thi-Phuong Khanh Nguyen, and Kamal Medjaher. The third result is 'Suivi par IPA des populations d'oiseaux nicheurs dans le bassin du Drugeon, Franche-Comté, France (1998-2020)', with an image of a bird, disciplines in biodiversity and ornithology, a collection date from April 1998 to June 2020, and creators Dominique Michelat and Patrick Giraudoux. On the right side, there are filters for 'Discipline' (environmental sciences, paleontology, archaeology, geosciences, physical geography, geology, ecology, limnology) and 'Structure' (OSU THETA, UFC, Chrono-environnement, uB, UTINAM, Biogéosciences, FEMTO-ST, ICB). A 'Label' filter for ZAAJ is also visible.

Exemple de manipulation de bases de données

phpMyAdmin

Recentes Préférées

BDD_COURS

- Nouvelle table
- affiche
- cinema
- distribution
- film

BD_DIU

CHAT

sdamy

Serveur: localhost > Base de données: BDD_COURS > Table: cinema

Parcourir Structure SQL Rechercher Insérer Exporter Importer Opérations

✓ Affichage des lignes 0 - 3 (total de 4, traitement en 0.0001 seconde(s).)

```
SELECT * FROM `cinema`
```

Profilage [Éditer en ligne] [Éditer] [Expliquer SQL] [Créer le code]

Tout afficher | Nombre de lignes : 25 | Filtrer les lignes : Chercher dans cette table | Trier sur l'index: Aucun(e)

+ Options

	NUMEROCINEMA	NOMCINEMA	ADRESSECINEMA	CODEPOSTALCINEMA	VILLECINEMA	TELEPHONECINEMA
<input type="checkbox"/> Éditer Copier Supprimer	1	Vox	Grande Rue	25000	Besançon	03 81 82 05 69
<input type="checkbox"/> Éditer Copier Supprimer	2	Plazza	Rue des granges	25000	Besançon	03 81 26 35 98
<input type="checkbox"/> Éditer Copier Supprimer	3	Plazza	Rue Ronchaux	21000	Dijon	03 80 26 53 25
<input type="checkbox"/> Éditer Copier Supprimer	4	Royal	Rue Villarceau	21000	Dijon	03 80 23 52 58

Tout cocher Avec la sélection : Éditer Copier Supprimer Exporter

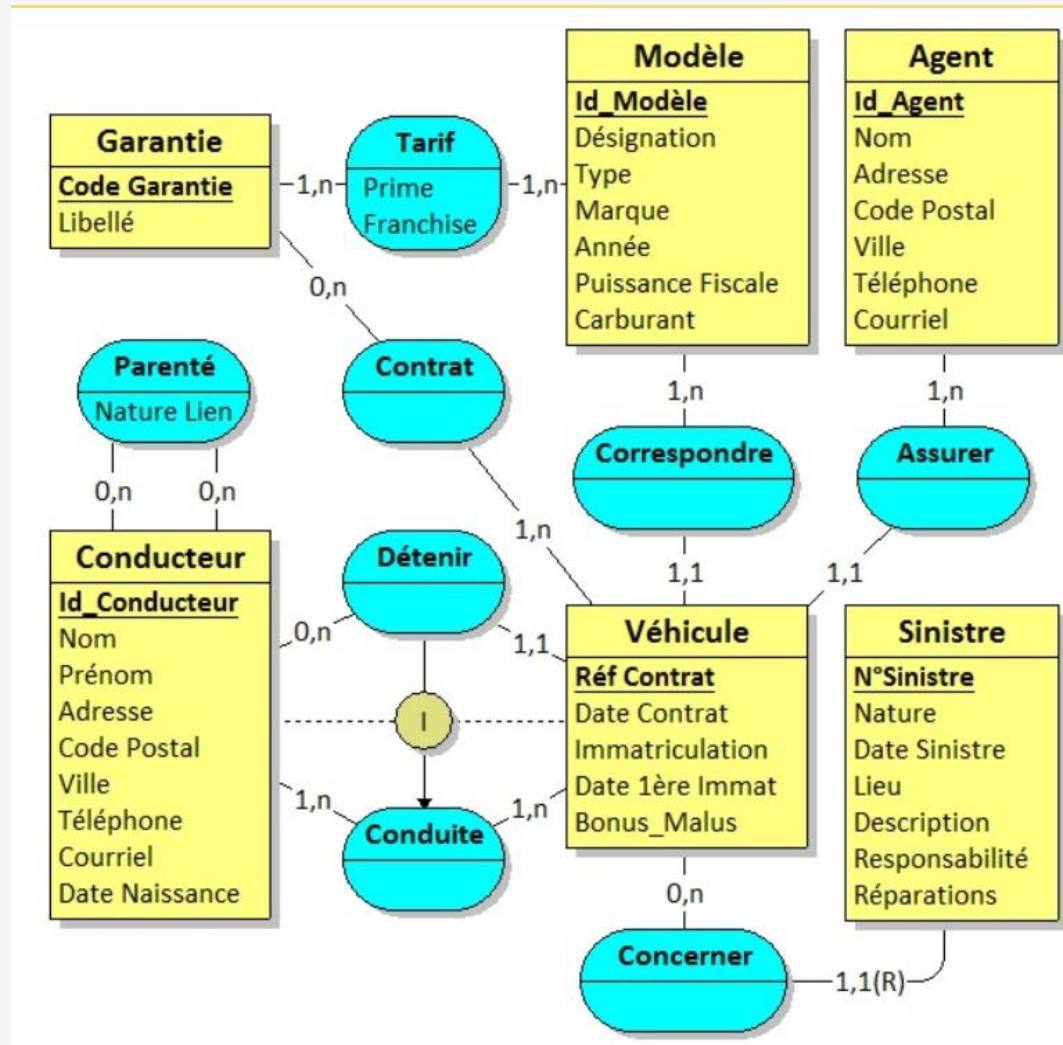


Utilisateur averti

Exemple de manipulation de bases de données

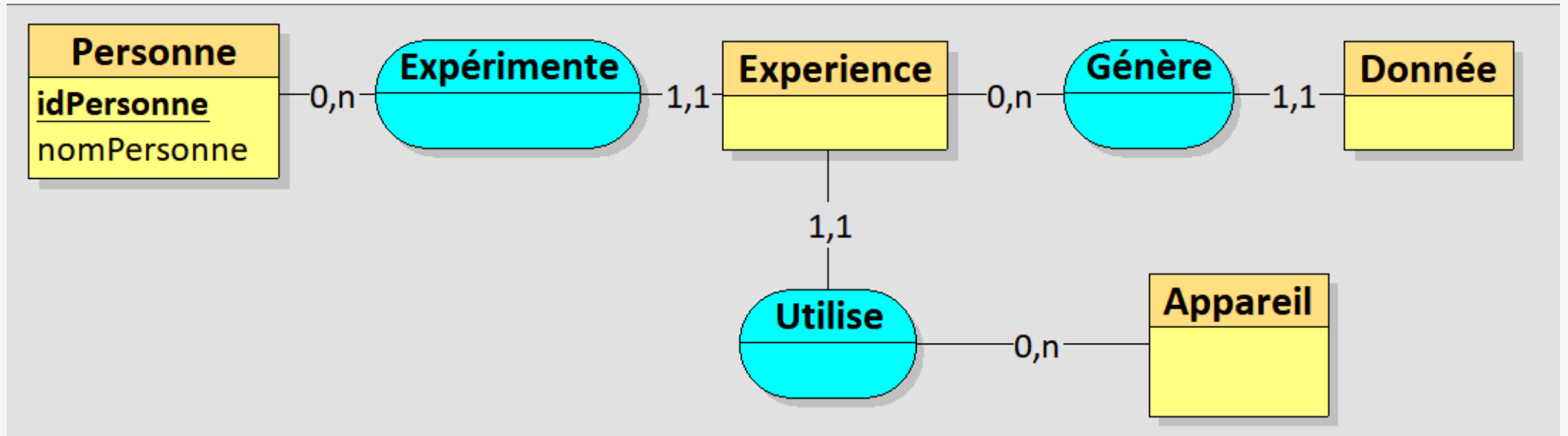


Modélisateur



Partie 2

Modélisation - MCD

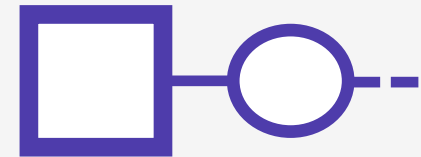


Les étapes de la conception d'une base de données

Analyse du système à modéliser

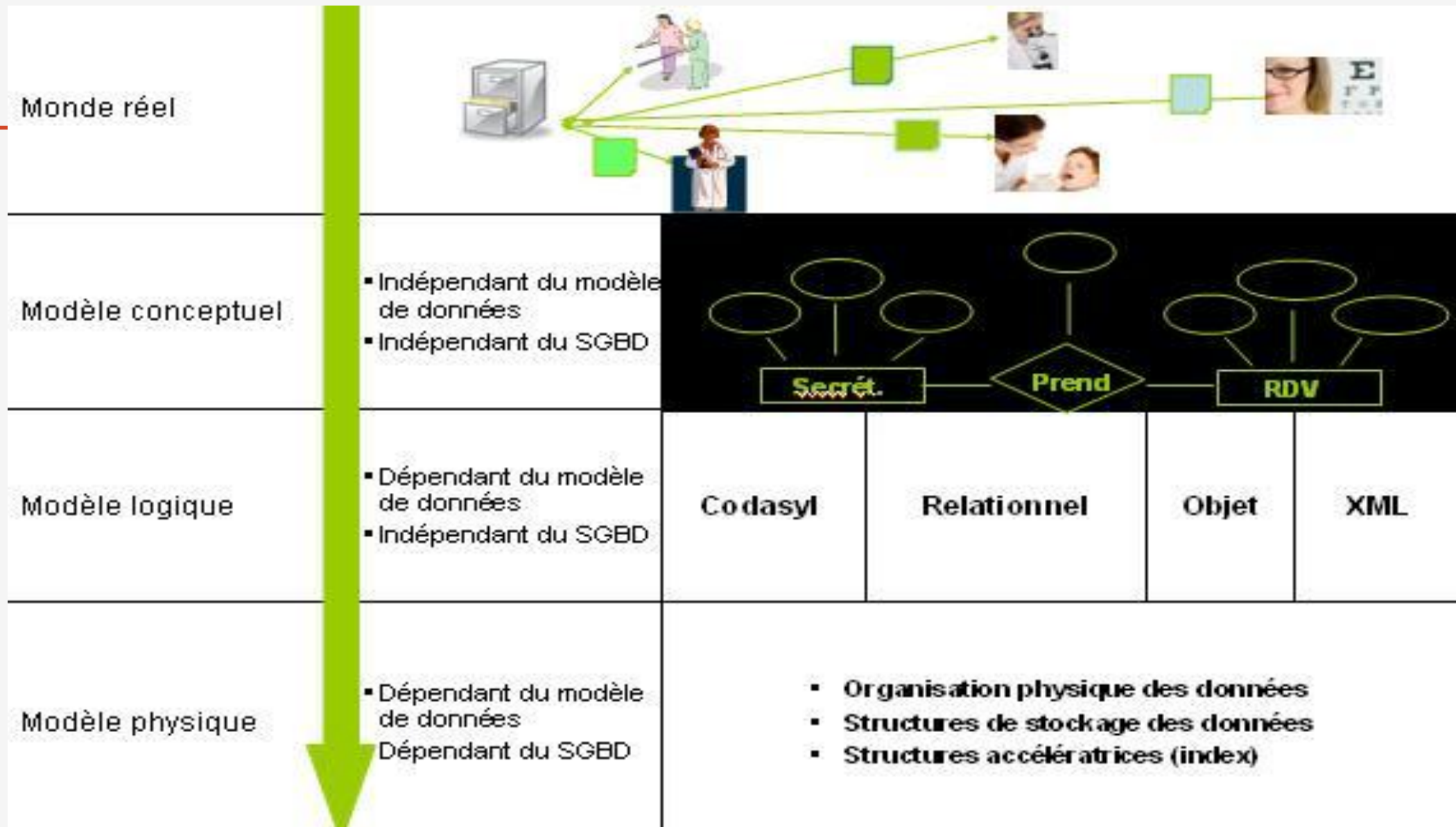


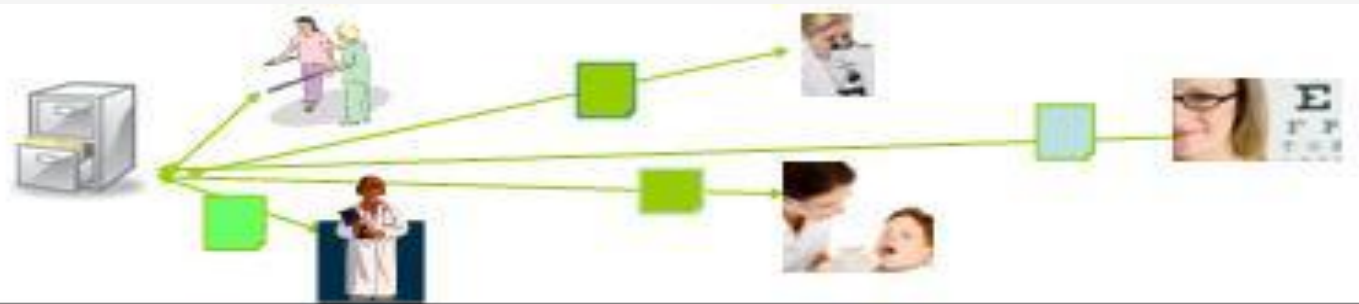
Création du modèle de données



Création de la base de données dans
un SGBD







Monde réel

Modèle conceptuel

- Indépendant du modèle de données
- Indépendant du SGBD



Merise, UML

Modèle logique

- Dépendant du modèle de données
- Indépendant du SGBD

Codasyl

Relationnel

Objet

XML

Modèle physique

- Dépendant du modèle de données
- Dépendant du SGBD

- **Organisation physique des données**
- **Structures de stockage des données**
- **Structures accélératrices (index)**



Modélisation - MCD

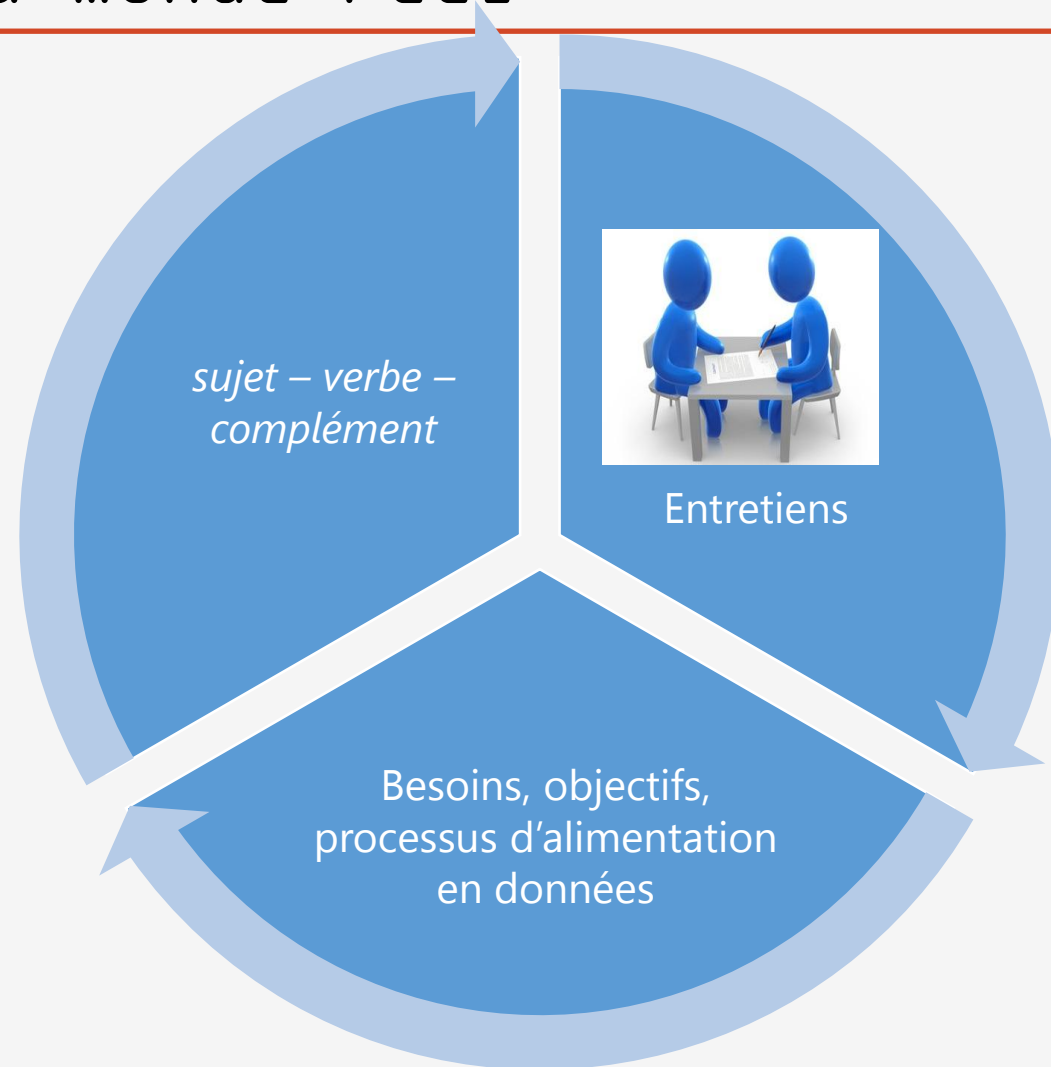
Première étape :

- Recueillir les informations puis les transcrire sous une forme facilitant le passage à un modèle de base de données (modèle relationnel)
- Utilisation du modèle entité-association (MERISE)
- Autre possibilité : UML - diagramme de classes (non vu dans ce cours)

Première description du monde réel

Comment appréhender et simplifier le monde réel pour réaliser une modélisation ?

Processus itératif

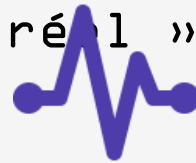


Première description du monde réel

Formulation des besoins sous forme de phrases simples :

« *sujet - verbe - complément* »

- Faire l'inventaire des objets du « monde réel »
- Décrire l'activité en termes simples
- Caractériser les liens entre les objets



Description des données, des objets et des liens

Exemple : Les gauloiseries



Faire l'inventaire des objets du monde réel :

- Ingrédient
- Potion

Décrire l'activité en termes simples :

« l'application à concevoir doit permettre de gérer l'ensemble des potions et leur composition. »

Exemple : Les gauloiseries



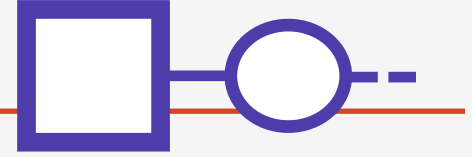
Caractériser les liens entre objets :

- Un ingrédient peut entrer dans la composition de plusieurs potions.
- Une potion peut être composée de plusieurs ingrédients...

Décrire les objets :

- Un ingrédient a un nom, un lieu de récolte
- Une potion a un nom, un effet, une durée d'effet

Modèle entité-association



Représentation graphique des informations recueillies dans un modèle : MCD ou **Modèle Conceptuel des Données**

Une **entité** est un objet du monde réel

Une **association** est un lien entre entités

Une **propriété** est une caractéristique associée à une entité ou à une association

Entité



Une *entité* est un objet spécifique du monde réel

Désigne : une personne, un objet, un concept abstrait, un événement

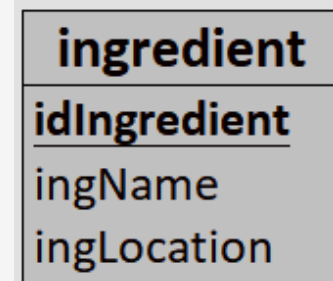
Caractérisée par son unicité

Exemples d'entités : potion magique, tomate

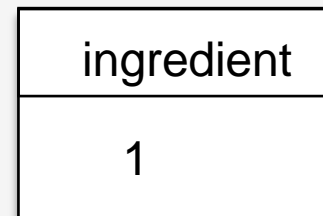
Entités ou entités types

L'ensemble des entités qui ont la même sémantique et les mêmes propriétés forment un *ensemble d'entités* appelé *entité type*

Représentation graphique d'une entité type



Représentation graphique d'une entité



Propriété ou attribut

Toute entité-type possède des propriétés, appelées *attribut*

MonEntite
<u>idEntite</u>

Exemple : nom de la personne, prénom, date de naissance, ...

- Un attribut ne peut pas être partagé par plusieurs entités-types ou associations-types
- Un attribut est une donnée élémentaire (pas de données calculées)
- L'entité-type et ses attributs forment un ensemble cohérent

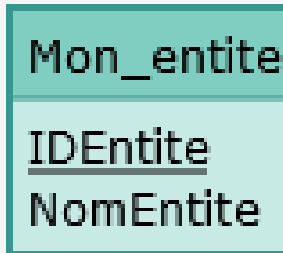
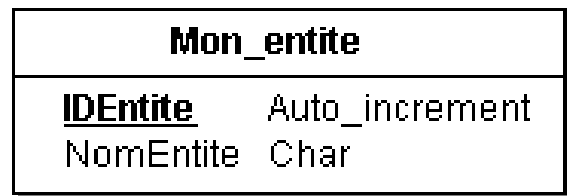
Identifiant

Toute entité-type a un *identifiant* : C'est un attribut ou ensemble d'attributs qui permet d'identifier chaque entité

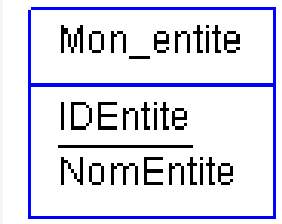
L'identifiant ne peut pas prendre la même valeur pour deux entités distinctes

Exemples d'identifiants : Numéro de sécurité sociale d'une

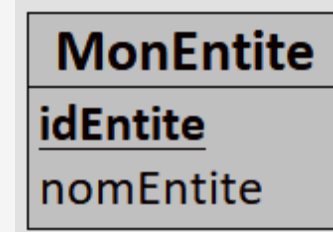
personne, D
Représentati
on graphique
de
l'identifian
t



MoCoDOnline



Analyse_SI



Looping

Identifiant

Entité-type PERSONNE

nom de personne unique →

Identifiant = nom de la personne (NomPersonne)

nom et prénom uniques →

Identifiant = (NomPersonne, PrenomPersonne)

Identifiant

Entité-type PERSONNE

Dans la pratique on évite des identifiants composés de plusieurs attributs → 1 seul attribut

ID_Personne : numéro automatique géré par le SGBD

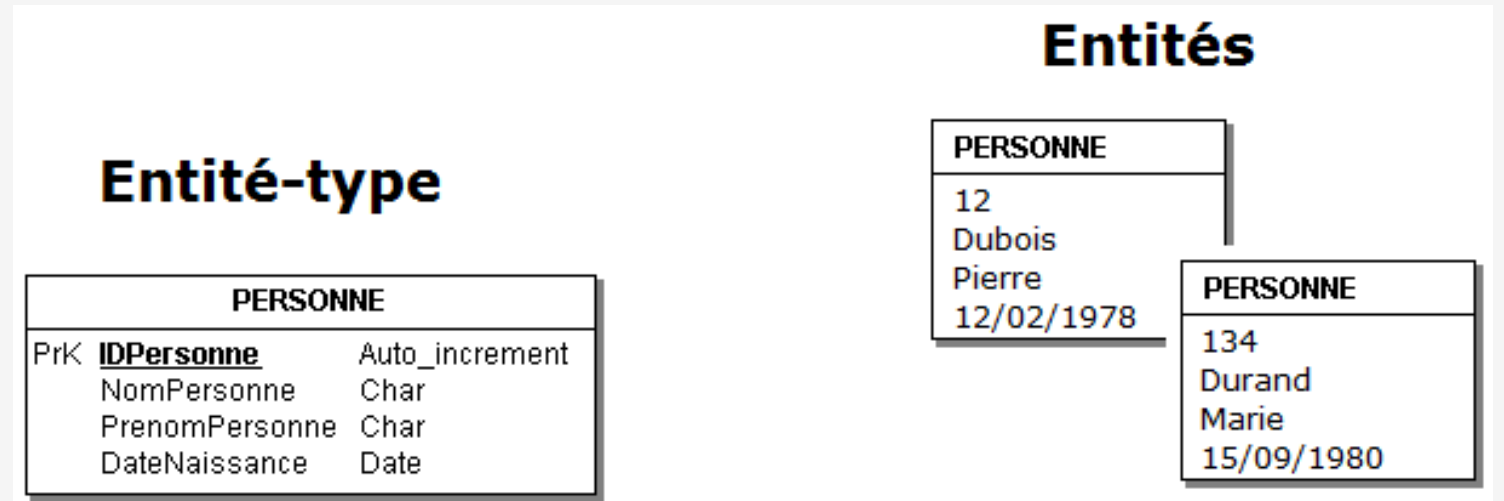
PERSONNE	
NomPersonne	Char
PrenomPersonne	Char
DateNaissance	Date



PERSONNE	
PrK <u>IDPersonne</u>	Auto_increment
NomPersonne	Char
PrenomPersonne	Char
DateNaissance	Date

Les entités : représentation graphique

Rectangle
comportant en haut
le nom de l'entité-
type et dans le bas
la liste des
attributs



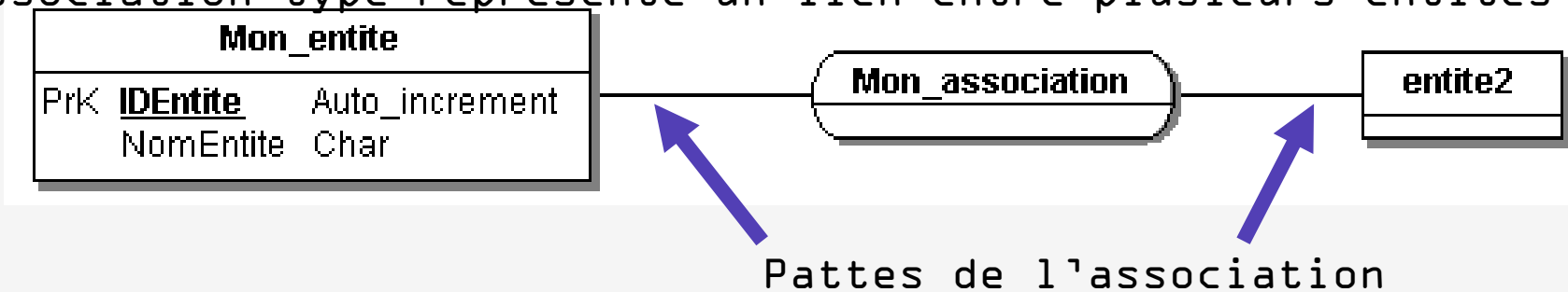
Association

Une **association** représente un lien entre plusieurs entités. Elle peut contenir des attributs

- Jean Dupont possède le livre de code ISBN 97821000720057
- Jean Dupont est marié avec Marie Dubois

Une association-type est un ensemble d'associations qui possèdent les mêmes caractéristiques

Une association-type représente un lien entre plusieurs entités-type



Cardinalité d'une association-type

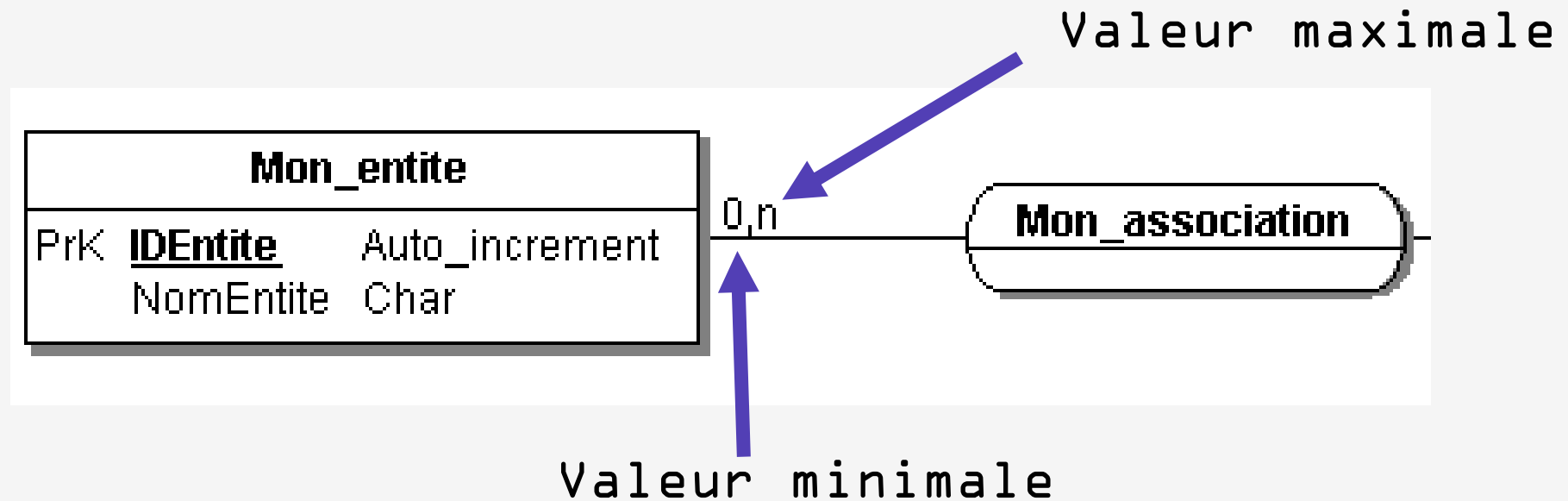
On définit une *cardinalité* sur chaque patte de l'association-type. Elle lie une entité-type et une association-type et est composée de 2 nombres :

valeur minimale : nombre minimal d'interventions d'une entité de l'entité-type dans l'association-type. Elle a pour valeur 0 ou 1

valeur maximale : nombre maximal d'interventions d'une entité de l'entité-type dans l'association-type. Elle a pour valeur 1 ou n

Cardinalité d'une association-type

La cardinalité est obligatoire pour chaque patte de l'association-type



Cardinalité d'une association-type

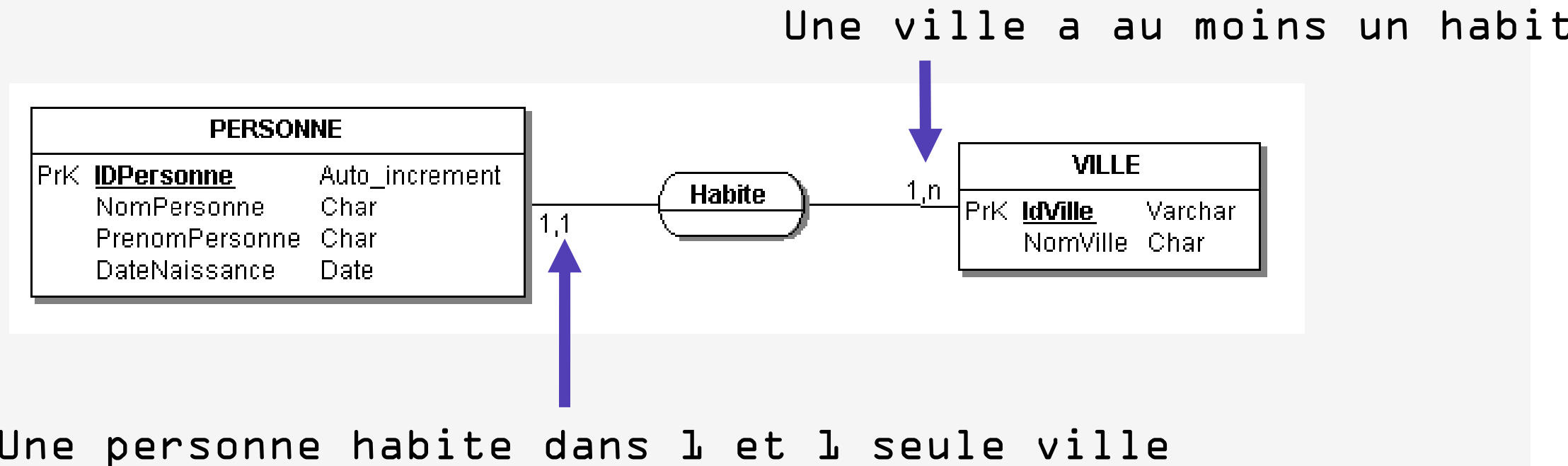
0..1 → une entité peut être impliquée dans aucune association ou une au plus

1..1 → une entité est impliquée dans une et une seule association

0..n → une entité peut être impliquée dans aucune association ou dans plusieurs (sans limite)

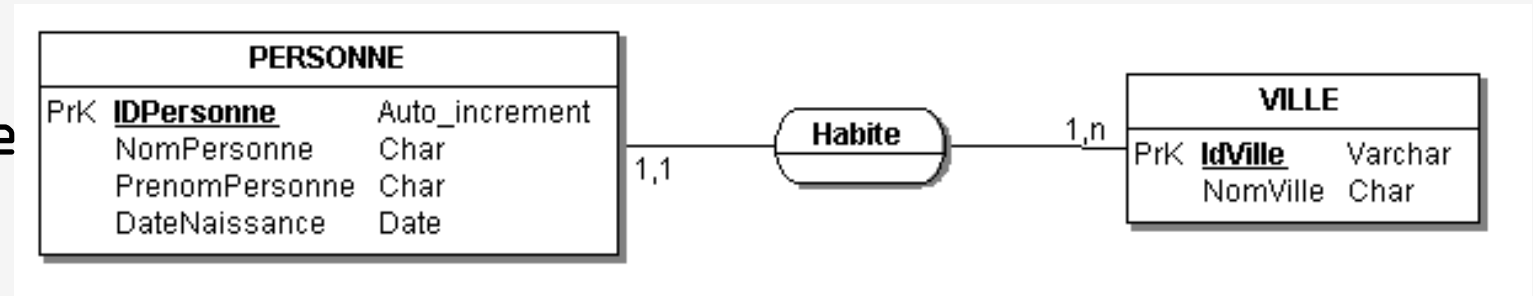
1..n → une entité est impliquée dans au moins une association

Cardinalité d'une association-type

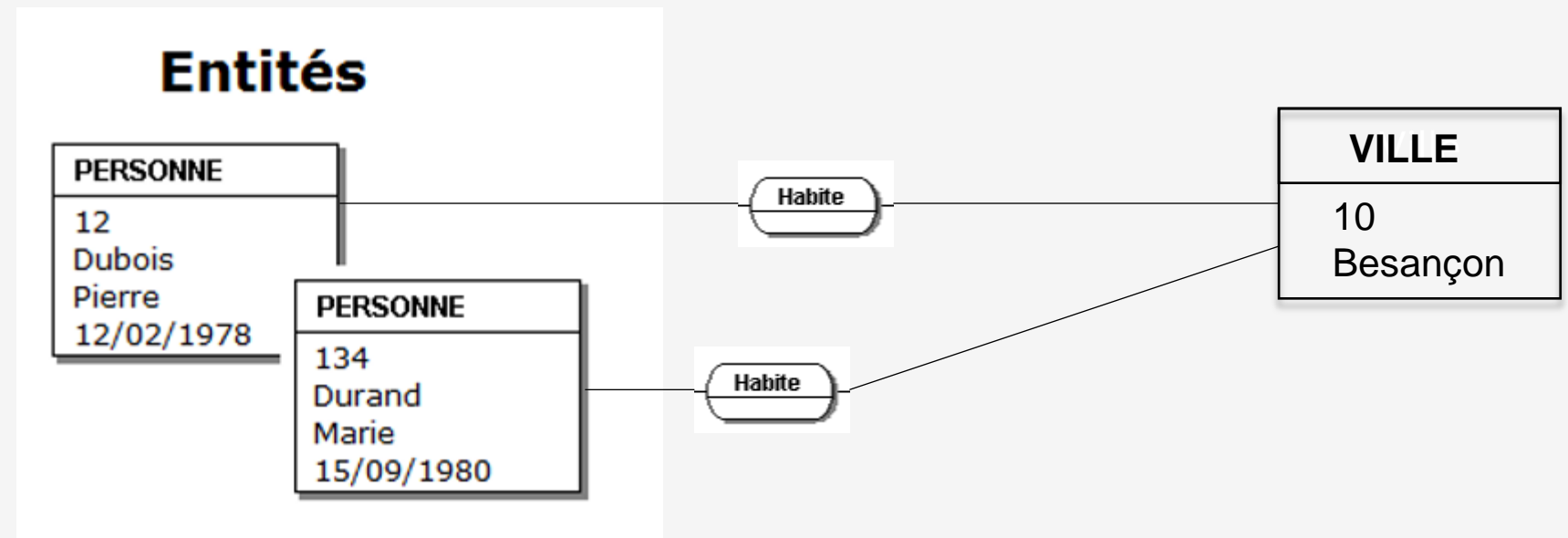


Association-type et association

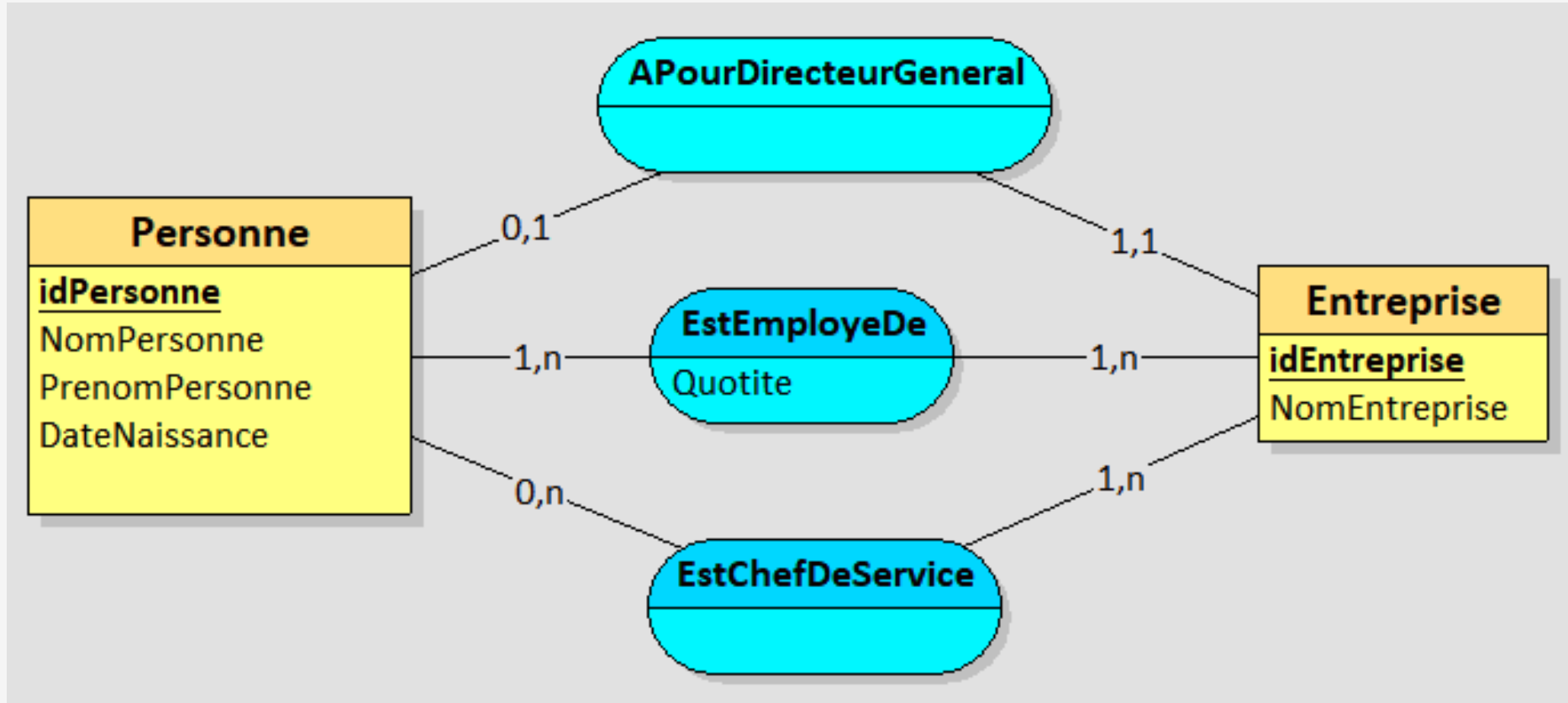
Association-type



Associations



Cardinalité d'une association-type

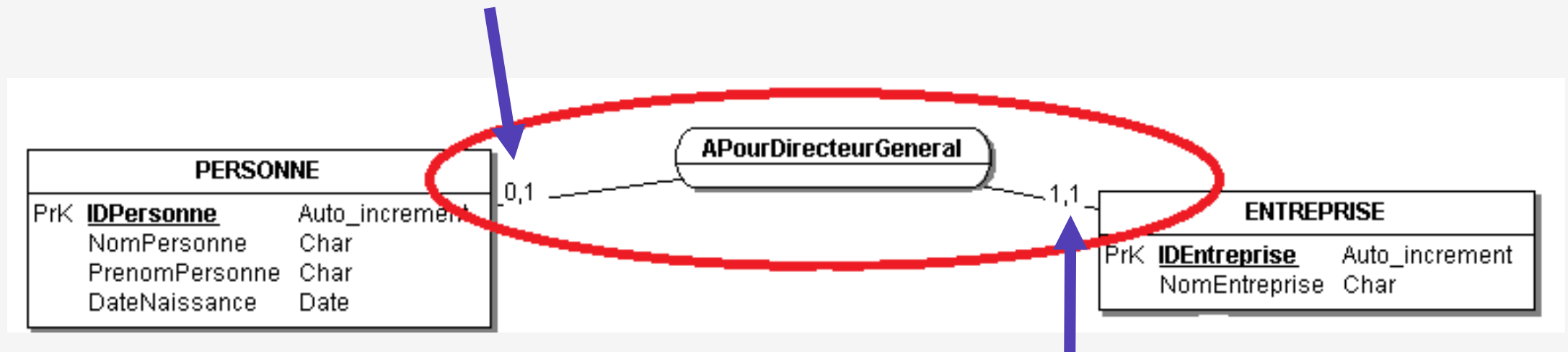


Que veulent dire ces cardinalités ?

Cardinalité d'une association-type

APourDirecteurGeneral :

Une personne peut être directeur général d'au plus une entreprise



Une entreprise a un et un seul directeur général

Cardinalité d'une association-type

EstEmployeDe :

Une personne travaille dans une
à plusieurs entreprises

Une entreprise a de un à n
employés



Quotite : attribut placé dans l'association type *EstEmployeDe*
Chaque valeur de cet attribut dépend d'une personne et d'une entreprise

Cardinalité d'une association-type

EstChefService :

Une personne peut être chef de 0 à n services dans des entreprises



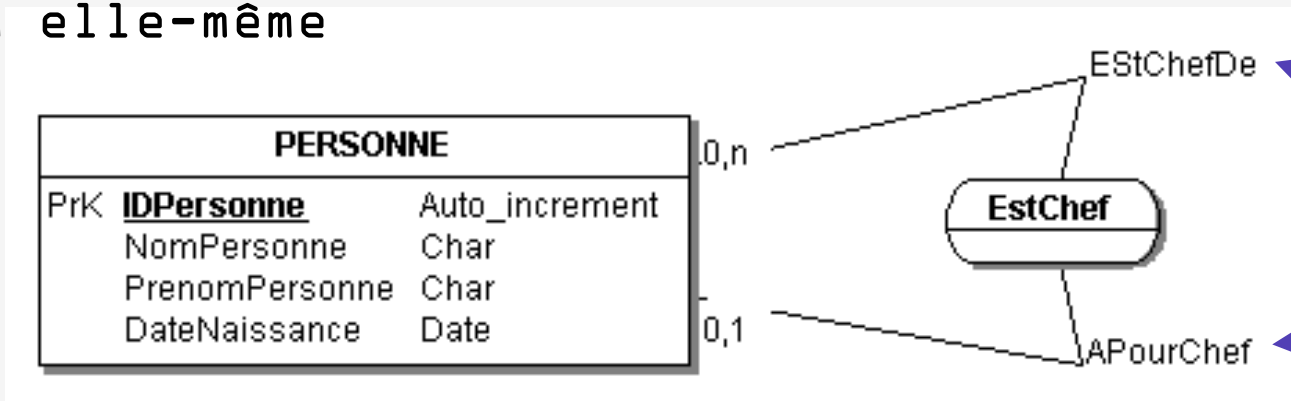
Une entreprise a 1 à n chefs de service

Associations-types particulières

Association-type binaire : associe deux entités-types



Association-type réflexive : associe une entité-type à elle-même



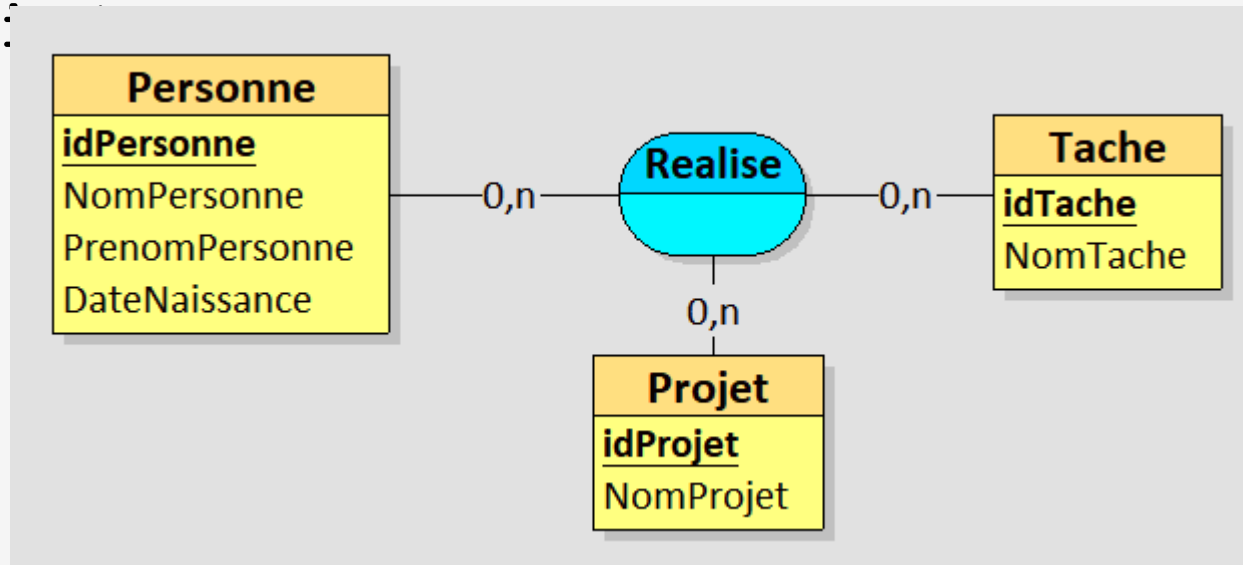
EstChefDe : une personne peut être chef de 0 à n personnes

APourChef : une personne peut avoir 0 ou 1 chef de service

Associations-types particulières

Association-type n-aire : associe plus de deux

ent:

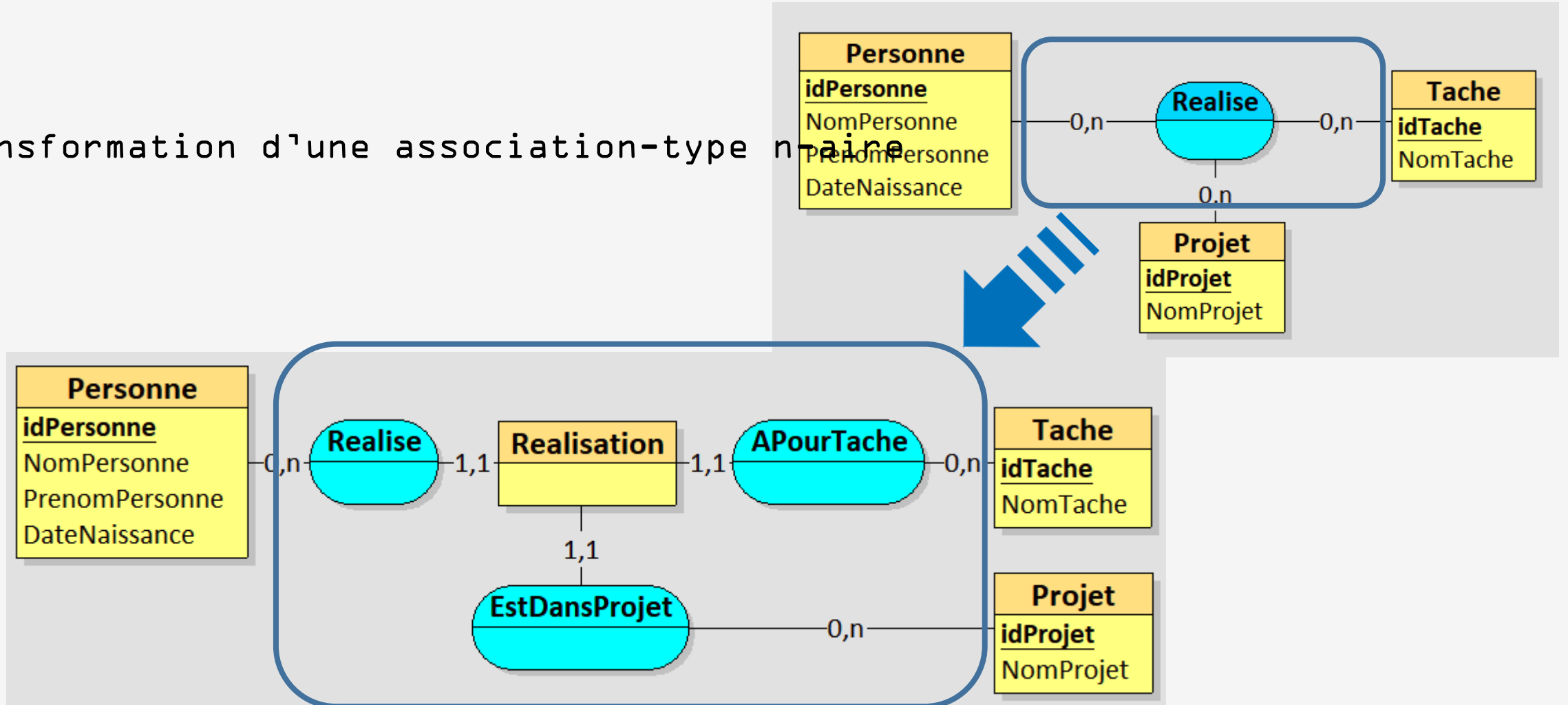


Pas de cardinalité maximale égale à 1

Une personne réalise de 0 à n tâches dans différents projets (L

Associations-types particulières

Transformation d'une association-type n-aire



Construction d'un MCD : Les étapes

Recueil des besoins :

- Documents
- Cahier des charges
- ...

Tri de l'information :

- Problèmes de
- synonymie
 - polysémie des attributs

Identification des entités-type :

- Repérer les attributs servant d'identifiant
- Une entité a une existence propre

Identification des associations-type :

- Relier les entités-type.
- Regarder les associations type de cardinalité 1,1

Vérification du modèle

- :
- Normalisation

Construction d'un MCD : Quelques conseils

Choix des noms

- Entité-type : Nom au singulier
 - Personne, Entreprise, ...
- Association-type : Verbe
 - Conjugué, à l'infinitif, avec un adverbe, ...
- Attribut : Nom au singulier qui peut être complété avec le nom de l'entité type
 - NomClient, AdresseEntreprise, ...

Choix des identifiants

- Eviter les identifiants
 - Composés de plusieurs attributs
 - Qui peuvent changer au cours du temps
 - De type chaîne de caractères
- Privilégier les identifiants de type entier pour les entités type
 - Deviendront des clés primaires de type incrément automatique (auto-incrément) dans le SGBD

Dictionnaire des données



Document qui regroupe tous les attributs de la base de données

Peut servir pour la rédaction d'un DMP !!!

Pour chaque attribut on décrit :

- **Nom** : libellé décrivant l'attribut (ou pr
- **Descriptif** : décrit à quoi sert l'attribut
- **Type** : Caractères, Numérique, Date, Booléen, ...
- **Taille** : Nombre de caractères, entier long, court, réel double, ...
- **Information supplémentaire** : si nécessaire

Informations supplémentaires :
Standard utilisé, liste de valeurs, unité utilisée, ...

NomPays, Nom du pays, Chaîne de car., 50 CodePays, Code ISO 3166 du pays, Chaîne de caractères, 3

Exemple : Les gauloiseries



Nom	Descriptif	Type	Taille	Informations supplémentaires
idPotion	Identifiant de la potion	Increment auto		Peut-on générer automatiquement les identifiants de potion ou existe-t-il déjà une façon de les identifier ?
poName	Nom de la potion	car	50	Si on a déjà des listes de valeurs regarder la longueur des noms de potions
poEffect	Effet de la potion	car	50	
poDuration	Durée d'effet de la potion	entier		Durée en minutes

Exemple : Les cinémas

Nom	Descriptif	Type	Taille	Informations supplémentaires
NumCinema	Identifiant du cinéma	Increment auto		Peut-on générer automatiquement les identifiants du cinéma ou existe-t-il déjà une façon de les identifier ?
NomCinema	Nom du cinéma	car	100	Si on a déjà des listes de valeurs regarder la longueur des noms de cinémas
VilleCinema	Commune sur laquelle le cinéma est situé	car	100	Voir si il existe des listes de villes de référence intéressantes
AdresseCinema	Rue et autres éléments complémentaires	car	200	A-t-on besoin de décomposer l'adresse ?
TelephoneCinema	Numéro de téléphone du cinéma	car Num	10 Entier	On considère qu'il n'y a qu'un seul numéro de tel. par cinéma. A gérer plutôt en caractères qu'en numérique
CodePostal	Code postal	Num	Entier	Attention aux entiers (>= 32767)

Exemple : Les gauloiseries



Nous souhaitons manipuler les informations permettant de décrire des potions et leurs ingrédients.

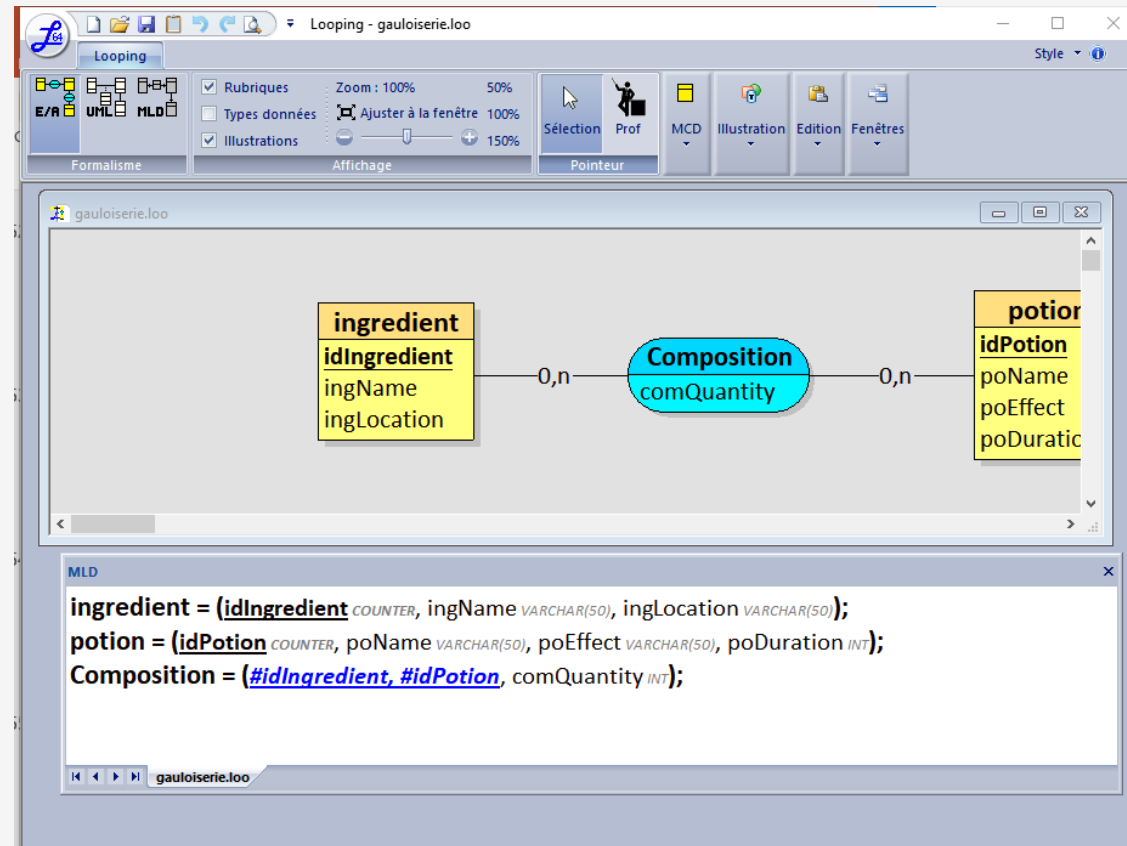
Une potion a un nom,
un effet et une
durée

Un ingrédient a un
nom et un lieu de
récolte

Un ingrédient peut apparaître dans la composition de plusieurs potions et une potion peut avoir dans sa composition plusieurs ingrédients.

Construction du MCD

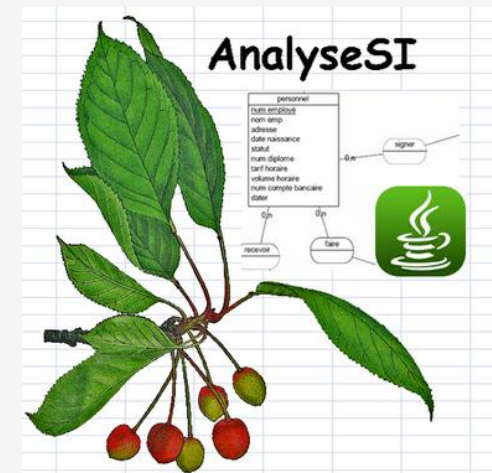
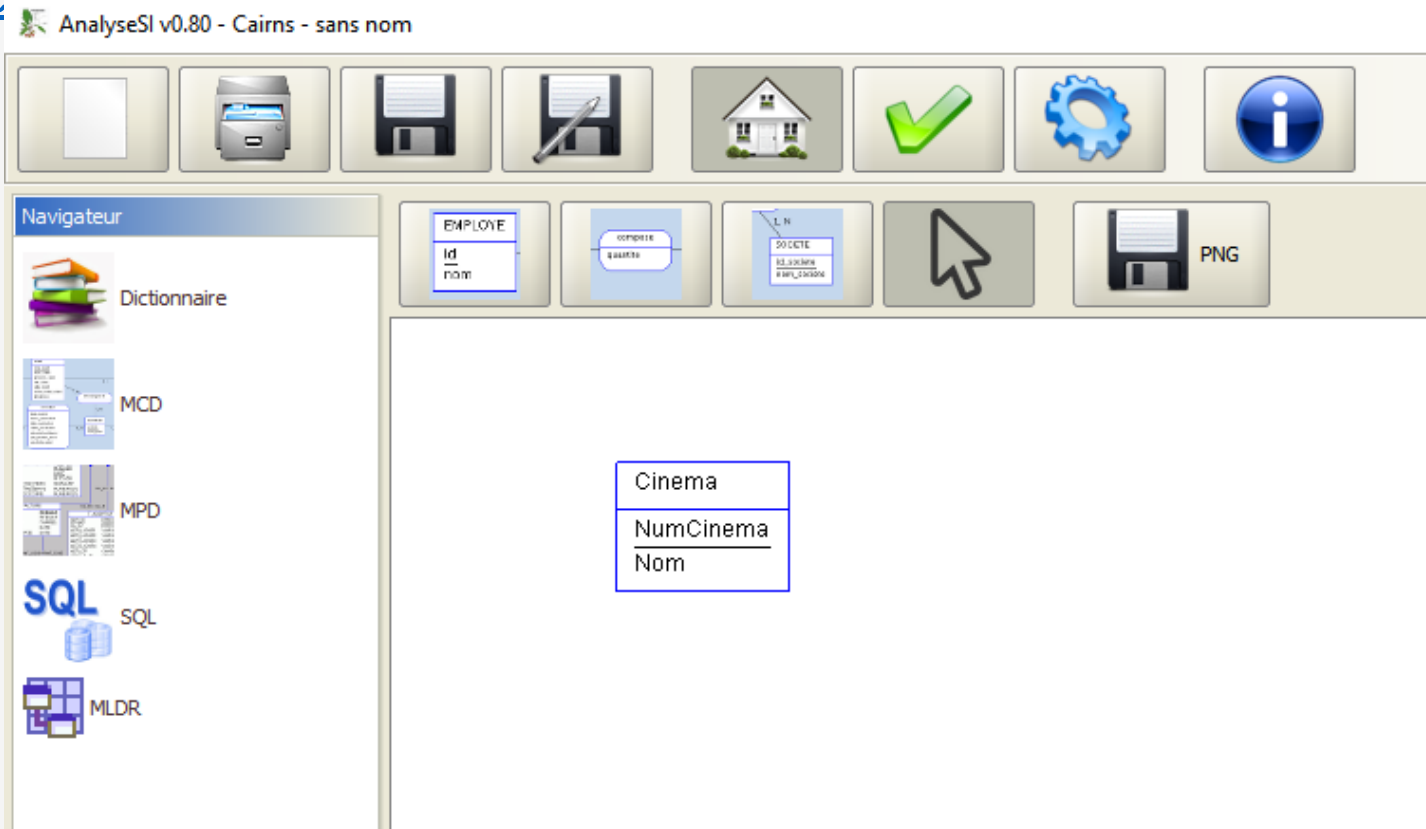
Looping : Outil libre <https://www.looping-mcd.fr/#>



Construction du MCD

AnalyseSI : outil libre

<https://launchpad.net/analyse-si/download>



Construction du MCD

Mocodo : outil en ligne

<http://www.mocodo.net/>



The screenshot shows the Mocodo online tool interface. At the top, the logo 'MOCODOonline' is displayed. Below it are navigation tabs: 'Entrée', 'Options', and 'Retouches'. The main content area is titled 'Animaux' and contains the following text-based MCD:

```
PEUT VIVRE DANS, 1N ESPÈCE, 1N ENCLOS: nb. max. congénères
ENCLOS: num. enclos
OCCUPE, 1N ANIMAL, 1N PÉRIODE, 1N ENCLOS
PÉRIODE: date début, _date fin

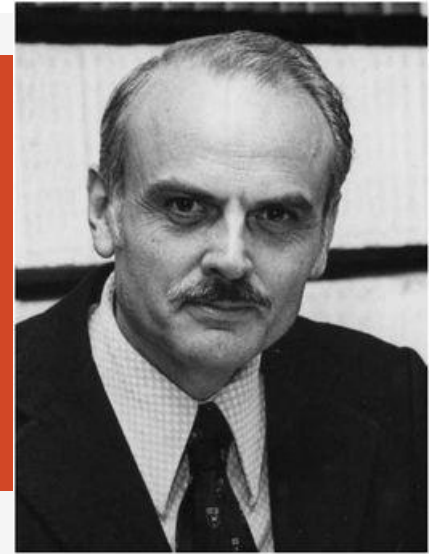
ESPÈCE: code espèce, libellé
DF, 0N ESPÈCE, _11 ANIMAL
ANIMAL: nom, sexe, date naissance, date décès
A MÈRE, 01 ANIMAL, 0N> [mère] ANIMAL

PEUT COHABITER AVEC, 0N ESPÈCE, 0N [commensale] ESPÈCE: nb. max. commensaux
:
A PÈRE, 0N ANIMAL, 0N> [père présumé] ANIMAL
:
```

Below the text is a diagrammatic representation of the MCD. It shows entities: 'ESPÈCE' (code espèce, libellé), 'ANIMAL' (nom, sexe, date naissance, date décès), 'ENCLOS' (num. enclos), 'PÉRIODE' (date début, date fin), 'A MÈRE', and 'A PÈRE'. Relationships are shown as boxes: 'PEUT VIVRE DANS' (nb. max. congénères), 'OCCUPE', 'PEUT COHABITER AVEC' (nb. max. commensaux), and 'DF'. Cardinalities and constraints are indicated on the lines connecting these elements.

Partie 3

Modèle relationnel



Edgar Codd

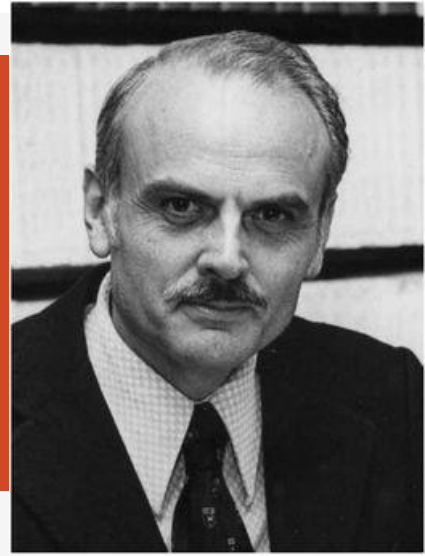
Définitions

Intégrité des données

Passage du MCD au modèle
relationnel

Partie 3

Modèle relationnel



Edgar Codd


Définitions

Intégrité des données

Passage du MCD au modèle
relationnel

La théorie relationnelle

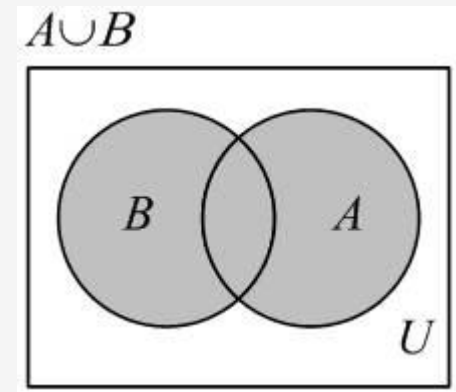
A relational model of data for large shared data banks

Author:  [E. F. Codd](#) [Authors Info & Claims](#)

Communications of the ACM, Volume 13, Issue 6 • June 1970 • pp 377–387 • <https://doi.org/10.1145/362384.362685>

Approche basée sur la théorie mathématique des ensembles

Aucune
modification
révolutionnaire

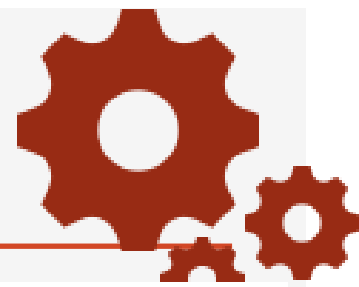


~~Définitions~~

Base de données

Quesako ?

Connexion au serveur de bases de données



votre login est sdamy

Votre mot de passe pour l'accès à phpmyadmin est: ?????

Accédez a [Phpmyadmin](#)

<https://coursbdd.univ-fcomte>



**Solution réservée
aux membres de
l'uFC**

phpMyAdmin

Bienvenue dans phpMyAdmin

Lingue - *Language*

Français - French

Connexion ⓘ

Utilisateur :

Mot de passe :

Exécuter

Connexion au serveur de bases de données



Il faut installer MySQL ou MariaDB sur votre machine

<https://www.youtube.com/watch?v=trPjbiGRwLw>



Pour les autres
membres d'UBFC

Consulter une base de données



1^{ère} étape : création de votre BD

Importez le fichier `gauloiseries.sql` (que je vous ai envoyé par mail) sous phpmyadmin

Structure SQL Rechercher Requête Exporter **Importer** Plus

Importation dans la base de données « sdamy »

Fichier à importer :

Le fichier peut être compressé (gzip, bzip2, zip) ou non.
Le nom du fichier compressé doit se terminer par **.[format].[compression]**. Exemple : **.sql.zip**

Parcourir les fichiers : Aucun fichier sélectionné. (Taille maximale : 2 048kio)

Il est également possible de glisser-déposer un fichier sur n'importe quelle page.

Jeu de caractères du fichier : ▼

Consulter une base de données



Qui-a-t-il dedans ?

<ul style="list-style-type: none">+ CHAT- sdamy<ul style="list-style-type: none">Nouvelle table+ composition+ ingredient+ potion	<table border="1"><thead><tr><th>Table</th><th>Action</th></tr></thead><tbody><tr><td><input type="checkbox"/> composition</td><td>★ Parcourir Structure Rechercher Insérer Vider Supprimer</td></tr><tr><td><input type="checkbox"/> ingredient</td><td>★ Parcourir Structure Rechercher Insérer Vider Supprimer</td></tr><tr><td><input type="checkbox"/> potion</td><td>★ Parcourir Structure Rechercher Insérer Vider Supprimer</td></tr><tr><td>3 tables</td><td>Somme</td></tr></tbody></table>	Table	Action	<input type="checkbox"/> composition	★ Parcourir Structure Rechercher Insérer Vider Supprimer	<input type="checkbox"/> ingredient	★ Parcourir Structure Rechercher Insérer Vider Supprimer	<input type="checkbox"/> potion	★ Parcourir Structure Rechercher Insérer Vider Supprimer	3 tables	Somme
Table	Action										
<input type="checkbox"/> composition	★ Parcourir Structure Rechercher Insérer Vider Supprimer										
<input type="checkbox"/> ingredient	★ Parcourir Structure Rechercher Insérer Vider Supprimer										
<input type="checkbox"/> potion	★ Parcourir Structure Rechercher Insérer Vider Supprimer										
3 tables	Somme										

Consulter une base de données



Consultez le contenu de la table potion

Qui-a-t-il dedans ?

idPotion	poName	poEffect	poDuration
1	Magic potion	Makes you strong	30
2	Happiness Potion	Makes people happy	120
3	Invisibility Potion	Makes you invisible	10
4	Anger potion	Makes you angry	5

Définitions de base

Table : Une table, R , sur un ensemble de domaines D_1, D_2, \dots, D_n , est constituée de 2 parties :
l'en-tête : ensemble fixé d'attributs.
le corps : ensemble de t-uplets.

T-uplet : Un t-uplet correspond à une ligne

Attribut : Un attribut correspond à une colonne

Table

idPotion	poName	poEffect	poDuration	Entête
1	Magic potion	Makes you strong	30	
2	Happiness Potion	Makes people happy	120	Corps
3	Invisibility Potion	Makes you invisible	10	
4	Anger potion	Makes you angry	5	T- uplet

Attrib
ut

Définitions

Domaine : Ensemble fini ou infini de valeurs dans lequel des attributs puisent leurs valeurs

Cardinalité : Nombre de t-uplets

Degré : Nombre d'attributs

Définitions

Domaine : Ensemble fini ou infini de valeurs dans lequel des attributs puisent leurs valeurs

Cardinalité : Nombre

poName, poEffect :
chaîne
de caractères

idPotion, poDuration
entiers

idPotion	poName	poEffect	poDuration
1	Magic potion	Makes you strong	30
2	Happiness Potion	Makes people happy	120
3	Invisibility Potion	Makes you invisible	10
4	Anger potion	Makes you angry	5

Degré : Nombre d'attributs

Définitions

Domaine : Ensemble fini
dans lequel des attributs

Cardinalité =
Degré = 4

idPotion	poName	poEffect	poDuration
1	Magic potion	Makes you strong	30
2	Happiness Potion	Makes people happy	120
3	Invisibility Potion	Makes you invisible	10
4	Anger potion	Makes you angry	5

Cardinalité : Nombre de t-uplets

Degré : Nombre d'attributs

Définitions

Clé : Groupe d'attributs dont la valeur permet d'identifier de manière unique tout t-uplet de la table.

Schéma relationnel : Ensemble de relations qui forment la base de données (noms différents)

Définitions

Clé : Groupe d'attributs dont la valeur permet d'identifier de manière unique tout t-uplet de la table.

Schéma relationnel :
forment la base de do

Clé primaire
idPotion

idPotion	poName	poEffect	poDuration
1	Magic potion	Makes you strong	30
2	Happiness Potion	Makes people happy	120
3	Invisibility Potion	Makes you invisible	10
4	Anger potion	Makes you angry	5



Exemple de table : potion


Une potion a un identifiant, un nom, un effet et une durée d'effet

Les attributs de cette table sont :

idPotion, poName, poEffect et poDuration

idPotion	poName	poEffect	poDuration
1	Magic potion	Makes you strong	30
2	Happiness Potion	Makes people happy	120
3	Invisibility Potion	Makes you invisible	10
4	Anger potion	Makes you angry	5

La potion 4, appelée « Anger potion », rend furieux pendant 5 minutes



```
potion(*idPotion, poName, poEffect,  
poDuration)
```




Exemple de table : potion

Une potion a un identifiant, un nom, un effet et une durée d'effet

Les attributs de cette table sont :

idPotion, poName, poEffect et poD

idPotion	poName	poEffect	poDuration
1	Magic potion	Makes you strong	30
2	Happiness Potion	Makes people happy	120
3	Invisibility Potion	Makes you invisible	10
4	Anger potion	Makes you angry	5

`potion(*idPotion, poName, poEffect,
poDuration)`

Nom de la table

Clé de la table

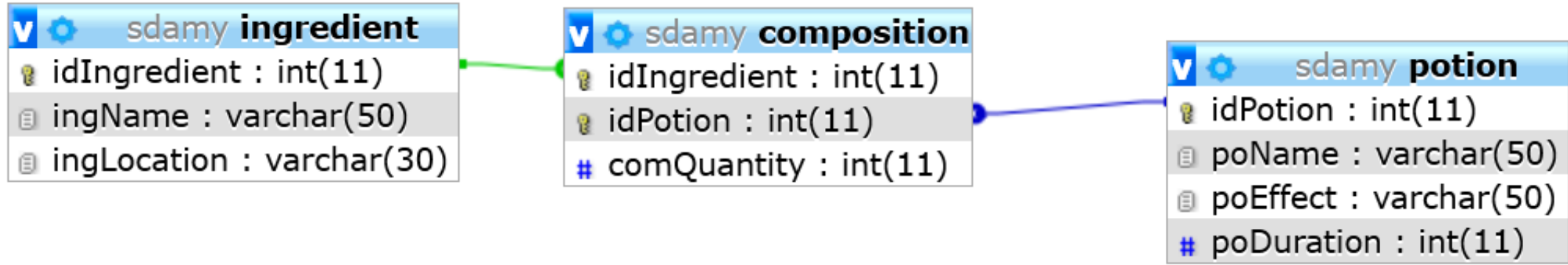
Tables potion et ingredient



idPotion	poName	poEffect	poDuration
1	Magic potion	Makes you strong	30
2	Happiness Potion	Makes people happy	120
3	Invisibility Potion	Makes you invisible	10
4	Anger potion	Makes you angry	5

idIngredient	ingName	ingLocation
1	water	anywhere
2	mistletoe	tree
3	Apple	Apple tree
4	Beer	hostel
5	Honey	forest
6	Lemon	Lemon tree
7	Raspberry	Garden
8	green salad	Garden

Schéma relationnel Gauloiseries



```
potion(*idPotion, poName, poEffect,  
poDuration)
```

```
ingredient(*idIngredient, ingName,  
ingLocation)
```

Définitions

Propriétés des tables

Propriétés des relations

Définition mathématique de la notion de relation

- Pas de duplication de t-uplets
- Les t-uplets ne sont pas ordonnés
- Les attributs ne sont pas ordonnés
- Les valeurs des attributs sont atomiques
- Les valeurs d'un attribut font toutes parties d'un même domaine

Pas de duplication de t-uplets

Ensemble des t-uplets d'une table = ensemble au sens mathématique

Une même valeur ne peut pas apparaître deux fois dans un ensemble.

→ Toute table possède une
clé

Ajout de données dans potion



idPotion	poName	poEffect	poDuration
1	Magic potion	Makes you strong	30
2	Happiness Potion	Makes people happy	120
3	Invisibility Potion	Makes you invisible	10
4	Anger potion	Makes you angry	5

Colonne	Type	Fonction	Null	Valeur
idPotion	int(11)			1
poName	varchar(50)			Magic potion
poEffect	varchar(50)			Make you strong
poDuration	int			

Ignorer

Colonne	Type
idPotion	int
poName	varchar

Erreur

Requête SQL : [Éditer](#)

```
INSERT INTO `potion` (`idPotion`, `poName`, `poEffect`,
```

MySQL a répondu : ?

#1062 - Duplicata du champ '1' pour la clef 'PRIMARY'

On ne peut pas avoir 2 t-uplets avec la même valeur

Les t-uplets ne sont pas ordonnés

Ensemble des t-uplets d'une table = Ensemble au sens mathématique.

Les t-uplets ne sont pas ordonnés.

→ Il n'existe pas de 2^o t-uplet dans une table, pas plus que de t-uplet suivant.



Table potion

idPotion	poName	poEffect	poDuration
1	Magic potion	Makes you strong	30
2	Happiness Potion	Makes people happy	120
3	Invisibility Potion	Makes you invisible	10
4	Anger potion	Makes you angry	5
5	test	test	2



idPotion	poName ▲ 1	poEffect	poDuration
4	Anger potion	Makes you angry	5
2	Happiness Potion	Makes people happy	120
3	Invisibility Potion	Makes you invisible	10
1	Magic potion	Makes you strong	30
5	test	test	2

Les attributs ne sont pas ordonnés

L'en-tête d'une table = ensemble d'attributs.

Comme pour les t-uplets, les attributs ne sont pas ordonnés.

→ Les attributs sont référencés par un nom et non par une position dans l'en-tête.

Table potion



idPotion

poName

poEffect

poDuration



poEffect

idPotion

poDuration

poName

Les valeurs des attributs sont atomiques

Les domaines contiennent des valeurs atomiques

Dans une table à l'intersection d'une ligne et d'une colonne il ne peut y avoir qu'une seule valeur

→ table normalisée : 1^o forme normale

table potion



*idPotion	poName	poEffect	poDuration
1, 2	Magic potion	Makes you strong	30
3	Invisibility Potion	Makes you invisible	10, 20
4	Anger potion	Makes you angry, Makes people happy	15

Cit_ID	Cit_Author	Year	Citation
1	Sitting Bull	1880	Lorsque la dernière goutte d'eau sera polluée, le dernier animal chassé et le dernier arbre coupé, l'homme blanc comprendra que l'argent ne se mange pas


Domaine unique

Dans une table les valeurs possibles d'un attribut sont données par son domaine

Pas d'association de plusieurs domaines à un même attribut (pas d'union de type)

Domaine unique : table potion



#	Nom	Type	Interclassement	Attributs	Null	Valeur par défaut	Commentaires	Extra
1	idPotion 	int(11)			Non	<i>Aucun(e)</i>		AUTO_INCREMENT
2	poName	varchar(50) utf8_bin			Non	<i>Aucun(e)</i>		
3	poEffect	varchar(50) utf8_bin			Non	<i>Aucun(e)</i>		
4	poDuration	int(11)			Non	<i>Aucun(e)</i>		



Domaines

poDuration ne peut pas prendre la valeur 'ND' ou '??' si l'utilisateur ne la connaît pas. C'est

un entier.

~~Définitions~~

Propriétés des
schémas relationnels

Propriétés du schéma relationnel

Noms des tables et des attributs uniques :

- Schéma de base de données = Ensemble de tables
- Table = Ensemble d'attributs

Il n'est pas possible d'avoir deux fois le même élément dans un ensemble

- Référencement ambigu

Propriétés du schéma relationnel

Une table a un nom unique dans le schéma relationnel

Un attribut a un nom unique dans une table

Propriétés du schéma relationnel



On ne peut pas avoir 2 tables portant le nom « potion »

On ne peut pas avoir 2 attributs dans la table « potion »
portant le nom « poName »

Propriétés du schéma relationnel



Créer une nouvelle table « potion »

Structure SQL Rechercher Requête Exporter Importer Opérations Plus

Nom de table: Ajouter colonne(s)

Structure						
Nom	Type	Taille/Valeurs*	Valeur par défaut	Interclassement	Attributs	Null
<input type="text" value="idpotion1"/>	<input type="text" value="INT"/>	<input type="text"/>	<input type="text" value="Aucun(e)"/>	<input type="text"/>	<input type="text"/>	<input type="checkbox"/>

Erreur

MySQL a répondu :

La table potion existe déjà !

Base de données « Gauloiseries »



```
potion(*idPotion, poName, poEffect, poDuration)
```

```
ingredient(*idIngredient, ingName, ingLocation)
```

```
composition(*idingredient, idPotion,  
comQuantity)
```

Compréhension du modèle

Pouvez-vous donner la recette de la potion magique ?

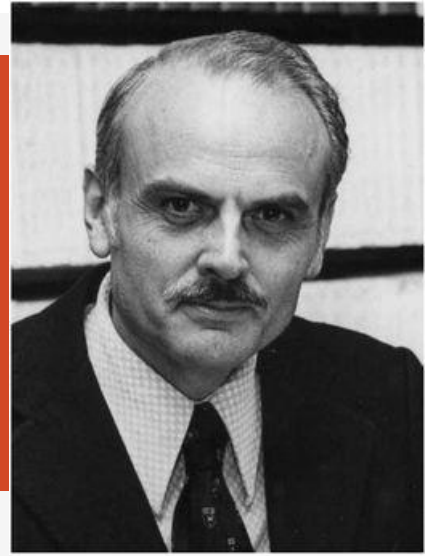
idPotion	poName	poEffect	poDuration
1	Magic potion	Makes you strong	30
2	Happiness Potion	Makes people happy	120
3	Invisibility Potion	Makes you invisible	10
4	Anger potion	Makes you angry	5
5	test	test	2

idIngredient	idPotion	comQuantity
1	1	2
2	1	1
4	4	10
5	1	1

idIngredient	ingName	ingLocation
1	water	anywhere
2	mistletoe	tree
3	Apple	Apple tree
4	Beer	hostel
5	Honey	forest
6	Lemon	Lemon tree
7	Raspberry	Garden
8	green salad	Garden

Partie 3

Modèle relationnel



Edgar Codd

Définitions

Intégrité des données

Passage du MCD au modèle
relationnel

~~Intégrité des données~~

Intégrités des
données ?

Intégrité des données relationnelles

Une base de données contient une configuration particulière des valeurs de données

Certaines configurations de valeurs n'ont pas de sens

→ Définition d'une base de données étendue pour inclure certaines contraintes dites d'intégrité

Intégrité des données

Contraintes d'intégrité : règles qui permettent au SGBD de conserver automatiquement la cohérence de la base de données



Intégrité des données relationnelles

Comment ?

- en vérifiant les données lors de leur chargement,
- en vérifiant les données lors de toute modification (saisie, mise à jour),
- en répercutant certaines mises à jour entre tables,
- en gérant les références entre tables.

Les 4 règles d'intégrité des données

Contraintes d'intégrité

:

- Unicité de clé
- Contraintes de domaine
- Contraintes de références
- Intégrité des entités

~~Intégrité des données~~

Notions préliminaires
: clé, lien, NULL

Les clés relationnelles

Clé : Ensemble d'attributs de la table qui identifie chaque t-uplet de la table

- Super-clé
- Clé candidate
- Clé primaire
- Clé étrangère



Super-clé

Une **super-clé** pour une table R est un sous-ensemble de l'ensemble des attributs de R qui identifie de façon unique chaque t -uplet de R

Une super-clé possède la Propriété d'unicité :

Il n'existe pas deux t -uplets distincts de R ayant la même valeur pour cet ensemble d'attributs

Super-clés de potion



```
potion(idPotion, poName, poEffect, poDuration)
```



Deux potions ne peuvent pas avoir le même `idPotion` → `(idPotion)` est une super-clé de potion.



Deux potions ne peuvent pas avoir le même `idPotion` et le même nom → `(idPotion, poName)` est une super-clé de potion.

Les clés candidates

Une clé candidate pour une table R_1 est une super-clé irréductible

Propriété d'irréductibilité : Un ensemble d'attributs K est irréductible si aucun sous-ensemble strict de K n'est une super-clé

Clés candidates de potion



`potion(idPotion, poName, poEffect,
poDuration)`

`(idPotion)` est une super-clé de `potion` :

Clé réduite à un seul attribut →

Irréductible

→ Clé candidate



`(idpotion, poName)` est une super-clé de
`potion` : Non irréductible → Non clé
candidate



Les clés candidates

Les clés candidates fournissent un **mécanisme d'adressage** dans la table, car elles permettent de repérer un t-uplet dans

idpotion est la clé de potion



Les clés primaires

Une clé primaire est une clé choisie arbitrairement parmi les clés candidates

Il est possible d'avoir plusieurs clés candidates pour une table

→ Introduction de la notion de clé primaire, qui elle, est unique pour une table donnée

Clé primaire de potion



Ajoutez dans la table potion le t-uplet :

- `idPotion = 10`
- `poName = Magic potion`
- `poEffect = Makes you strong`
- `poDuration = 30`

<code>idPotion</code>	<code>poName</code>	<code>poEffect</code>	<code>poDuration</code>
1	Magic potion	Makes you strong	30
2	Happiness Potion	Makes people happy	120
3	Invisibility Potion	Makes you invisible	10
4	Anger potion	Makes you angry	5
5	test	test	2



`idPotion` différents => T-uplets différents
pour le SGBD

Clé primaire de potion



Ajoutez dans la table potion le t-uplet :

- `idPotion = 1`
- `poName = Laughing potion`
- `poEffect = Makes you laugh`
- `poDuration = 50`

<code>idPotion</code>	<code>poName</code>	<code>poEffect</code>	<code>poDuration</code>
1	Magic potion	Makes you strong	30
2	Happiness Potion	Makes people happy	120
3	Invisibility Potion	Makes you invisible	10
4	Anger potion	Makes you angry	5
5	test	test	2

`idPotion` égaux => Mêmes T-uplets pour le SGBD



Clé primaire de potion



idPotion	poName	poEffect	poDuration
1	Magic potion	Makes you strong	30
2	Happiness Potion	Makes people happy	120
3	Invisibility Potion	Makes you invisible	10
4	Anger potion	Makes you angry	5

The screenshot shows a database management tool interface with a menu bar (Parcourir, Structure, SQL, Rechercher, Insérer, Exporter, Importer) and a table structure view. The table has columns: idPotion (int(11)), poName (varchar(50)), poEffect (varchar(50)), and poDuration (int(11)). Below the table structure, there is a checkbox for 'Ignorer' and another table structure view. An error dialog box is overlaid on the right, titled 'Erreur', with the following content:

Requête SQL : [Éditer](#)

```
INSERT INTO `potion` (`idPotion`, `poName`, `poEffe
```

MySQL a répondu : [?](#)

#1062 - Duplicata du champ '1' pour la clef 'PRIMARY'

On ne peut pas avoir 2 t-uplets avec la même valeur de la clé idPotion

Les clés étrangères

Certains attributs ou ensemble d'attributs peuvent apparaître dans plusieurs relations

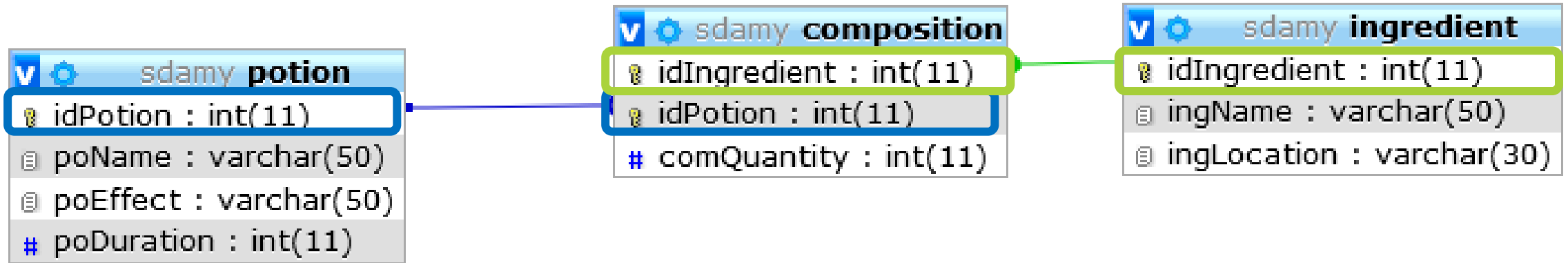
Une clé étrangère est un ensemble d'attributs d'une relation S qui est clé candidate d'une relation C

On parle de lien entre les relations S et C



Les clés étrangères

Certains attributs ou ensemble d'attributs peuvent apparaître dans plusieurs tables



Saisie de données dans composition



Essayez de saisir des informations telles que :

- `idPotion = 2, idIngredient = 1 et comQuantity = 2`
- `idPotion = 15, idIngredient = 1 et ComQuantity = 1`
- `idPotion = 2, idIngredient = 12 et ComQuantity = 1`

Que se passe-t-il ?

Base de données Gauloiseries



idPotion	poName	poEffect	poDuration
1	Magic potion	Makes you strong	30
2	Happiness Potion	Makes people happy	120
3	Invisibility Potion	Makes you invisible	10
4	Anger potion	Makes you angry	5
5	test	test	2

idIngredient	ingName	ingLocation
1	water	anywhere
2	mistletoe	tree
3	Apple	Apple tree
4	Beer	hostel
5	Honey	forest
6	Lemon	Lemon tree
7	Raspberry	Garden
8	green salad	Garden

idIngredient	idPotion	comQuantity
1	1	2
2	1	1
4	4	10
5	1	1

Les clés relationnelles

Soit C une table (cible), une **clé étrangère** dans C est un sous-ensemble $EtrC$, de l'ensemble des attributs de C , tel que :

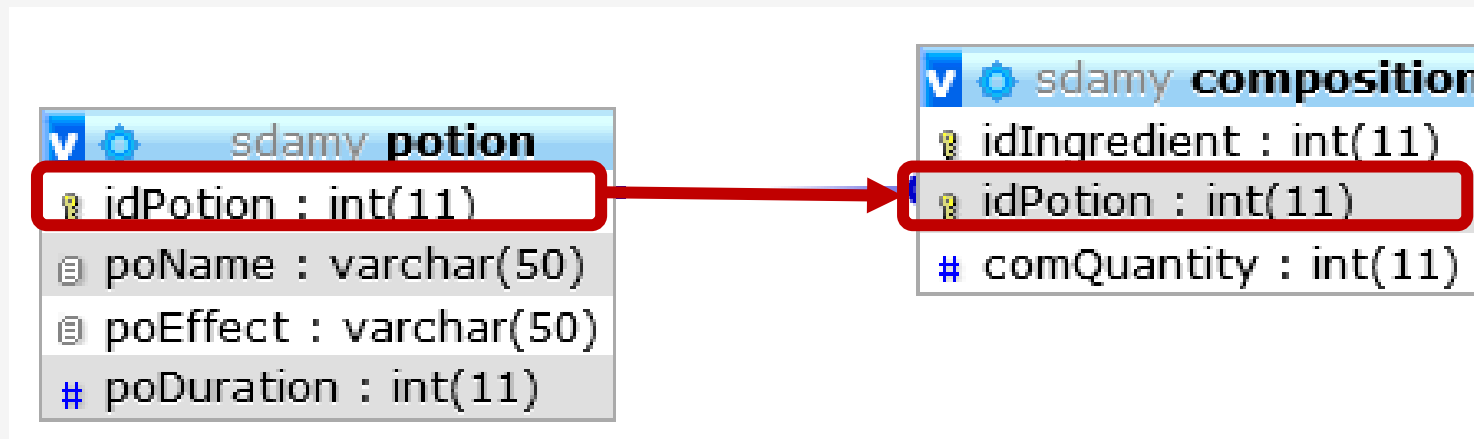
- Il existe une table S (source) avec une clé candidate $CleS$,
- et à tout instant, chaque valeur de $EtrC$ dans la table C , est identique à une valeur de $CleS$ dans la table S .

On définit un lien entre les tables S (ource) et C (ible).

Les clés étrangères

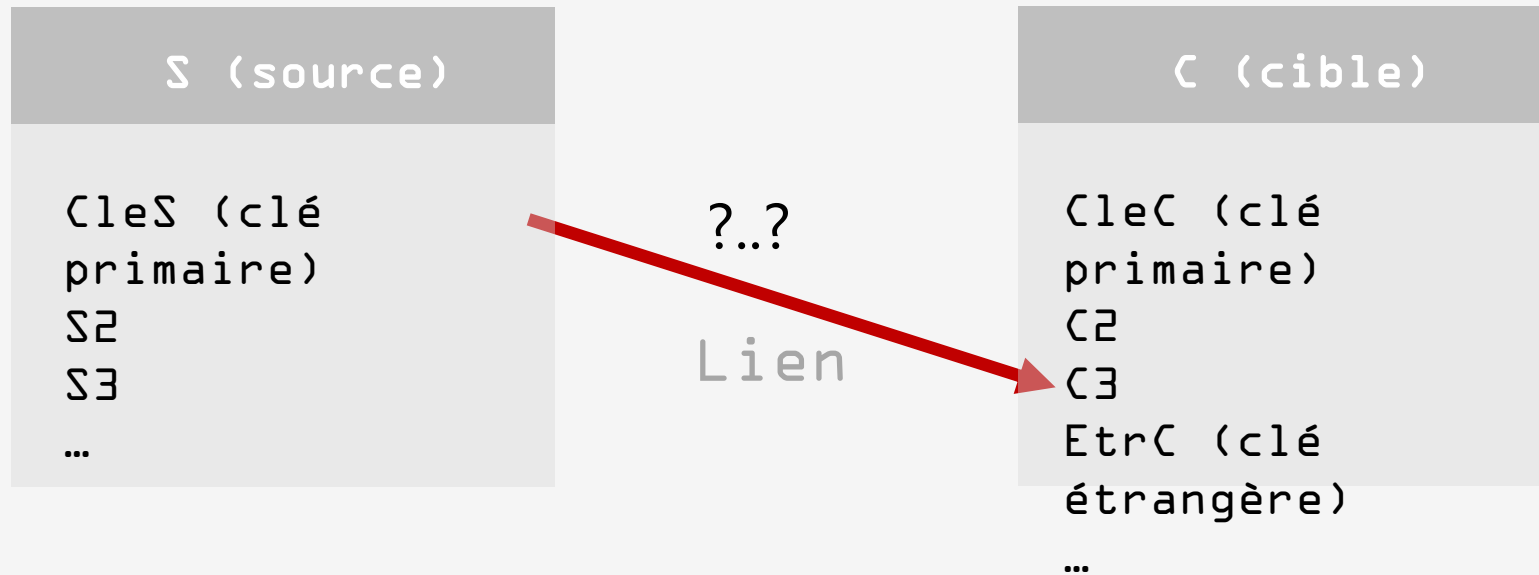


Certains attributs ou ensemble d'attributs peuvent apparaître dans plusieurs tables



Source → Cible

Représentation d'un lien



$S(\text{CleS}, S1, \dots) \llbracket \text{CleS} \rrbracket (? .. ?) \rightarrow \llbracket \text{EtrC} \rrbracket$
 $R(C1, \dots, \text{EtrC}, \dots)$

Les clés relationnelles

Différents types de liens selon le nombre de fois où la valeur de la clé primaire peut apparaître en tant que clé étrangère dans la relation cible

Lien 1..n : une valeur de la clé candidate peut apparaître 0 à n fois en tant que clé étrangère

Lien 1..1 : une valeur de la clé candidate ne peut apparaître qu'une et une seule fois en tant que clé étrangère

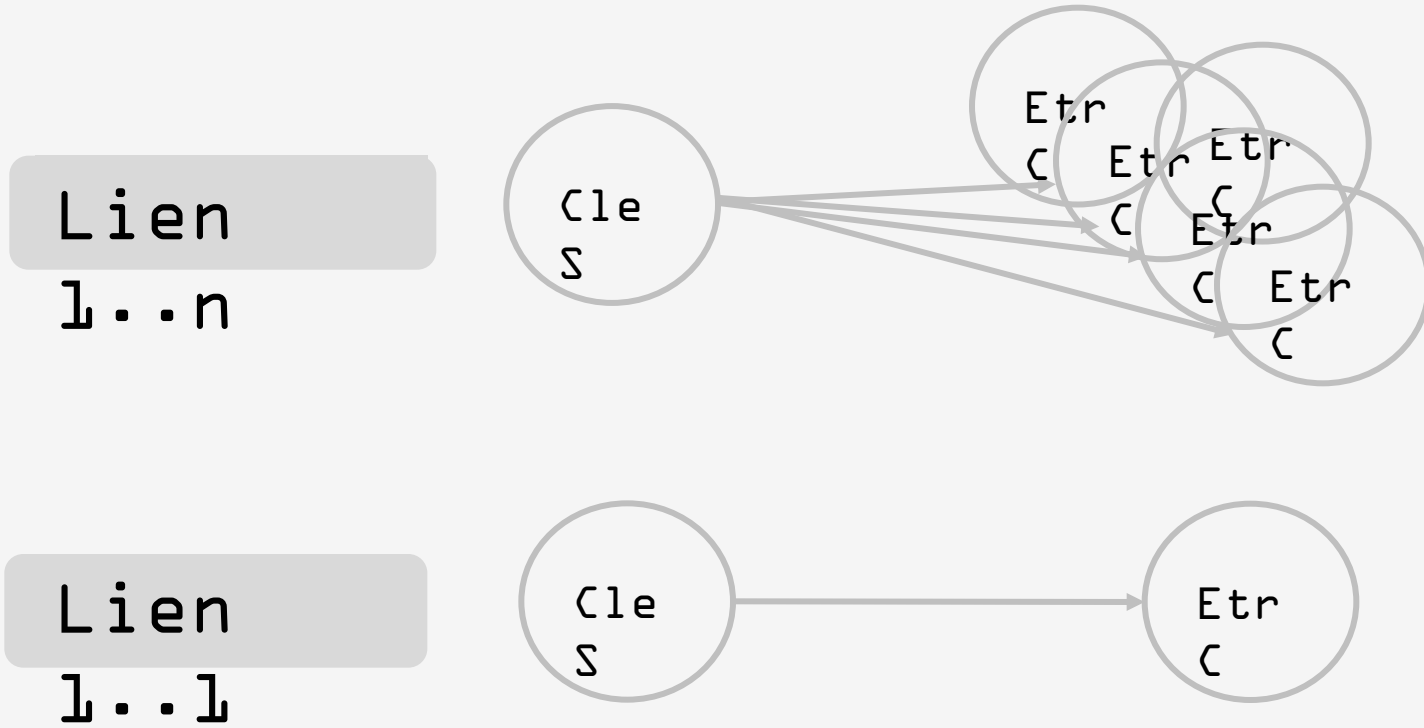
Les clés relationnelles

Différents types de liens selon le nombre de fois où la valeur de la clé primaire peut apparaître en tant que clé étrangère dans la relation cible

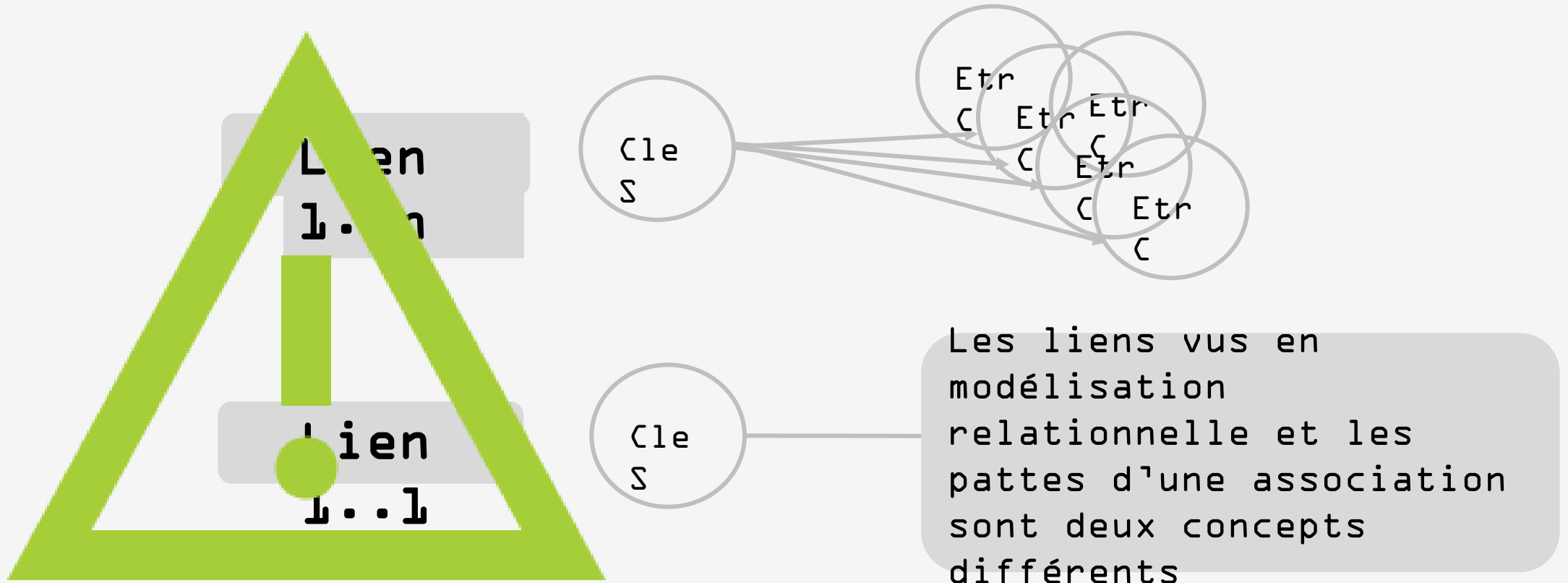
Lien 1..n : une valeur de la clé candidate peut apparaître 0 à n fois en tant que clé étrangère

Lien 1..1 : une valeur de la clé candidate ne peut apparaître qu'une et une seule fois en tant que clé étrangère

Les clés relationnelles



Les clés relationnelles



Définition de lien entre relations

Un lien est toujours défini d'une clé primaire vers une clé étrangère.

J'ai une valeur dans **une** relation (clé primaire) elle peut être référencée dans une autre table (clé étrangère).

Lien entre ingredient et composition



v	sdamy	composition
🔑		idIngredient : int(11)
🔑		idPotion : int(11)
#		comQuantity : int(11)

v	sdamy	ingredient
🔑		idIngredient : int(11)
📄		ingName : varchar(50)
📄		ingLocation : varchar(30)

Lien entre ces tables : Quel sens ?

Lien entre ingredient et composition



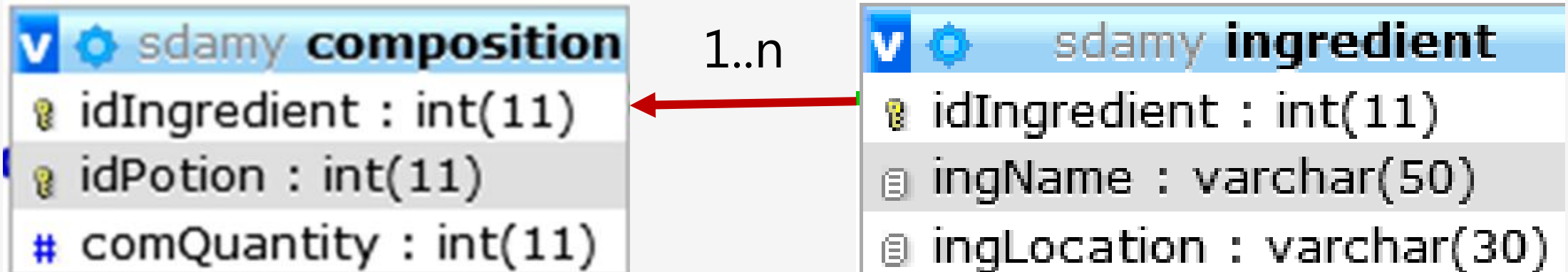
v	sdamy	composition
🔑		idIngredient : int(11)
🔑		idPotion : int(11)
#		comQuantity : int(11)

v	sdamy	ingredient
🔑		idIngredient : int(11)
📄		ingName : varchar(50)
📄		ingLocation : varchar(30)



Quel type de lien : 1..1 ou 1..n
?

Lien entre ingredient et composition



Type de lien : 1..n

Valeur NULL

Pas toujours possible de renseigner complètement toutes les données d'une base de données

→ Introduction de la valeur NULL

NULL : valeur conventionnelle introduite dans une relation lorsque que la valeur d'un des attributs est inconnue ou inapplicable.

Représente l'absence de valeur

Valeur NULL

Conception de la base de données : on précise si un attribut est autorisé ou non à prendre la valeur NULL

- Si l'attribut peut prendre la valeur NULL, cette valeur lui sera attribuée automatiquement par le SGBD, dès que l'utilisateur ne précisera pas la valeur de l'attribut
- Dans le cas contraire le SGBD rejettera tout t-uplet où la valeur de l'attribut ne serait pas renseignée

Valeur NULL



Conception de la base de données : on précise si un attribut est autorisé ou non à prendre la valeur NULL

Nom	Type	Taille/Valeurs*	Valeur par défaut	Interclassement	Attributs	Null	Ajuster les privilèges
poName	VARCHAR	50	Aucun(e)	utf8_bin		<input type="checkbox"/>	<input type="checkbox"/>
poEffect	VARCHAR	50	NULL	utf8_bin		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

Ajout de donnée



Colonne	Type	Fonction	Null	Valeur
idPotion	int(11)	<input type="text"/>		<input type="text"/>
poName	varchar(50)	<input type="text"/>		<input type="text"/>
poEffect	varchar(50)	<input type="text"/>	<input checked="" type="checkbox"/>	<input type="text"/>
poDuration	int(11)	<input type="text"/>	<input checked="" type="checkbox"/>	<input type="text"/>

idPotion	poName	poEffect	poDuration
1	Magic potion	Makes you strong	30
2	Happiness Potion	Makes people happy	120
3	Invisibility Potion	Makes you invisible	10
4	Anger potion	Makes you angry	5
12		NULL	NULL

Intégrité des données

Règles d'intégrité

L'unicité de clé

Première règle d'intégrité, basée sur les notions

- de clé candidate
- et de clé primaire

Règle de l'unicité de clé

Règle de l'unicité de clé : dans une base de données, toutes les relations doivent posséder une clé unique, appelée clé primaire.

Permet de mettre en place des mécanismes de contrôle, qui vérifient que deux informations identiques ne peuvent pas être présentes dans une relation de la base de données

→ La clé primaire ne peut prendre deux fois la même valeur



Clé primaire de potion

idPotion	poName	poEffect	poDuration
1	Magic potion	Makes you strong	30
2	Happiness Potion	Makes people happy	120
3	Invisibility Potion	Makes you invisible	10
4	Anger potion	Makes you angry	5

The screenshot shows a database management tool interface with a menu bar (Parcourir, Structure, SQL, Rechercher, Insérer, Exporter, Importer) and a table structure view. The table has columns: idPotion (int(11)), poName (varchar(50)), poEffect (varchar(50)), and poDuration (int(11)). The 'idPotion' column is marked as the primary key. An 'Insérer' dialog is open, showing the following values: idPotion: 1, poName: Laughing potion, poEffect: makes you laugh, poDuration: 50. A red error message box is overlaid on the dialog, stating: 'Erreur', 'Requête SQL : Éditer', 'INSERT INTO `potion` (`idPotion`, `poName`, `poEffe', and 'MySQL a répondu : #1062 - Duplicata du champ '1' pour la clef 'PRIMARY''.

On ne peut pas avoir 2 t-uplets avec la même valeur de la clé idPotion

Contraintes de domaine

Chaque attribut d'une relation prend ses valeurs dans un domaine (ensemble)

Domaines : types de base tels que entier, réel, chaîne de caractères, ...

Contrainte de domaine : contrainte d'intégrité qui définit la propriété que doit vérifier toute valeur d'un attribut d'une relation donnée

Contraintes de domaine

Contrainte traitée par le SGBD lors :

- de la saisie
- de la mise à jour d'information

pour vérifier la cohérence des données

Mise en œuvre de cette contrainte pas claire
dans MySQL

Contraintes de domaine dans composition



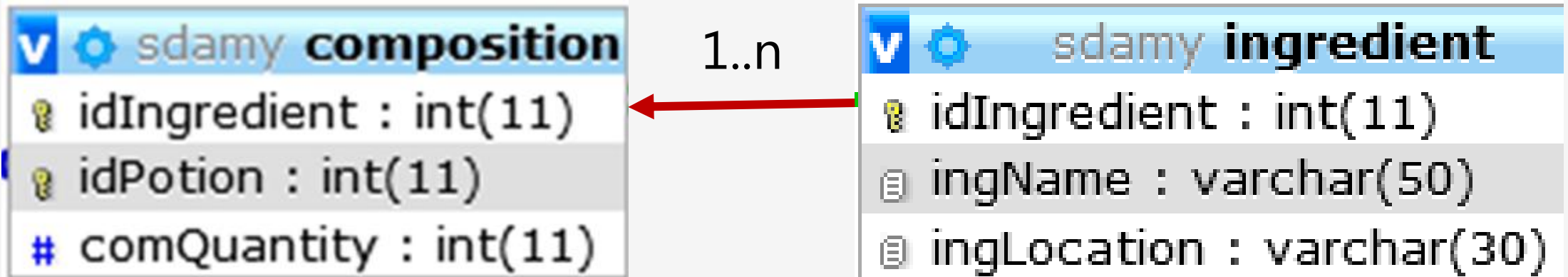
```
comQuantity :  
    domaine = entier compris entre 0  
et 100
```

Mise en œuvre de cette contrainte non opérationnelle
dans MySQL

Les contraintes de références

Permettent au SGBD de gérer automatiquement la présence de **données référencées** dans différentes relations de la base de données

Lien entre ingredient et composition



Le t-uplet $(l, l, l0)$ permet de dire que l'ingrédient l intervient dans la potion l en quantité = $l0$
On ne peut pas utiliser dans une potion, un ingrédient qui n'existe pas

L'intégrité référentielle

Intégrité référentielle : la base de données ne doit pas contenir de valeurs de clés étrangères non unifiables

Si une clé étrangère prend une valeur alors cette valeur existe dans la table source référencée par le lien

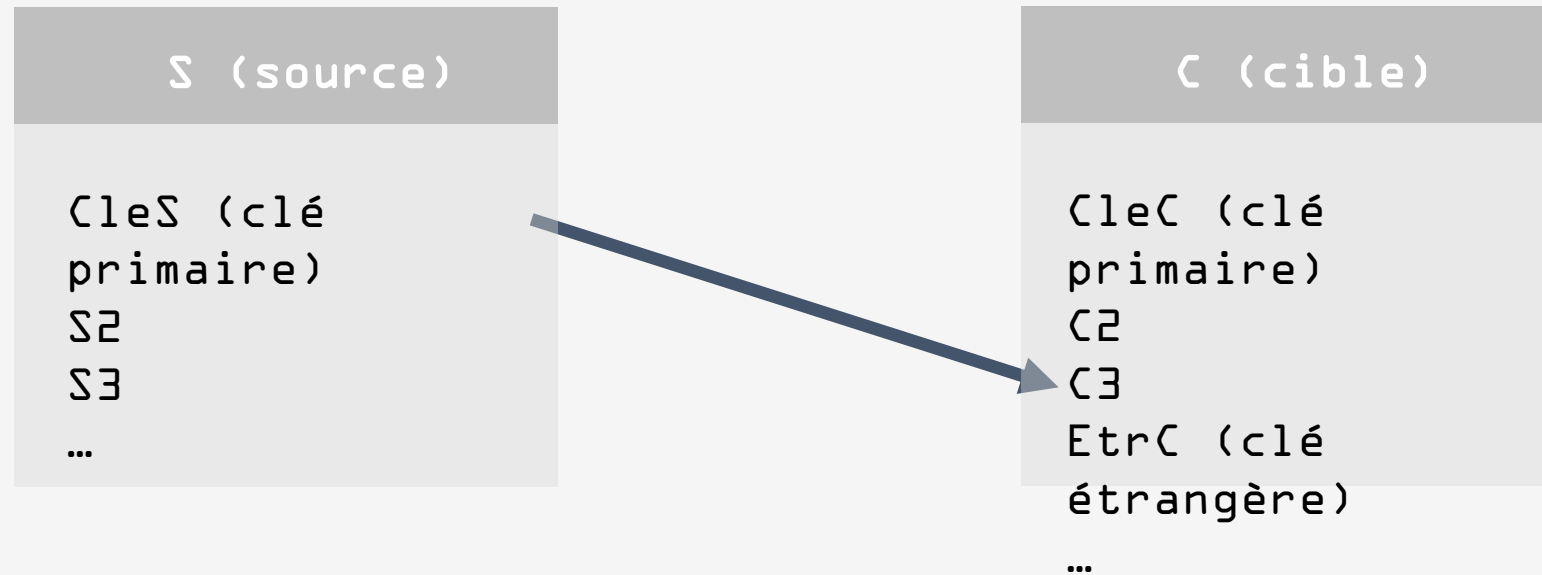
L'intégrité référentielle

Le SGBD doit gérer les situations où un utilisateur réalise des opérations qui invalident cette règle :

- 1^o possibilité : refuser toute modification de cette nature,
- 2^o possibilité : accepter l'opération, et prendre en compte un certain nombre d'opérations supplémentaires nécessaires pour garantir la cohérence des données.

Les règles de mises à jour permettent de définir la stratégie de prise en compte de ce type d'opérations

L'intégrité référentielle



A quel moment intervient l'intégrité référentielle ?

Intégrité référentielle et saisie



Saisie de la valeur d'une clé étrangère:

- Contrôle permettant de savoir si la valeur existe dans la table référencée
- Réalisé immédiatement

v	⚙	sdamy	composition
🔑			idIngredient : int(11)
🔑			idPotion : int(11)
#			comQuantity : int(11)

v	⚙	sdamy	ingredient
🔑			idIngredient : int(11)
📄			ingName : varchar(50)
📄			ingLocation : varchar(30)



Lors de la saisie d'un t-uplet dans composition, le SGBD vérifie qu'il existe dans ingredient un t-uplet avec la même valeur de idIngredient

Les règles de mises à jour

Opérations de modification ou suppression de données

RESTRICTED ou NO ACTION : opération de mise à jour "restreinte" au cas où aucune référence à cette valeur n'existe dans les autres relations en lien avec la relation de base, dans les autres cas elle est interdite.

CASCADE : opération de mise à jour réalisée en "cascade" dans les autres relations.

Restricted ou No action



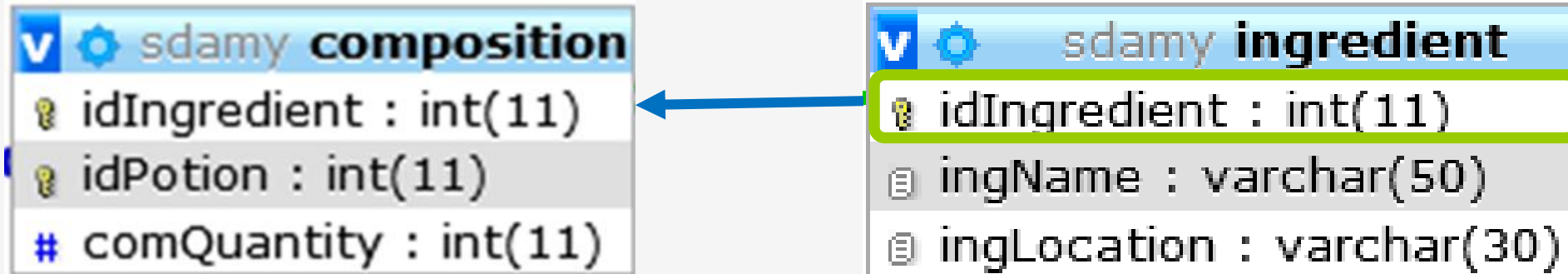
sdamy composition	
🔑	idIngredient : int(11)
🔑	idPotion : int(11)
#	comQuantity : int(11)

sdamy ingredient	
🔑	idIngredient : int(11)
📄	ingName : varchar(50)
📄	ingLocation : varchar(30)



RESTRICTED ou NO ACTION : opération de mise à jour "restreinte" au cas où aucune référence à cette valeur n'existe dans les autres tables en lien avec la table de base, dans les autres cas elle est interdite

Cascaded



CASCADE : opération de mise à jour réalisée en "cascade" dans les autres tables.

Les clés primaires et NULL

Règle d'intégrité des données : les attributs appartenant à la clé primaire d'une table ne sont pas autorisés à prendre la valeur NULL.

Les clés étrangères et NULL






Toute valeur d'une clé étrangère doit contenir une valeur unifiable.

Cette définition peut être étendue pour prendre en compte la valeur NULL.






Le concepteur de la BD peut préciser si oui ou non une clé étrangère peut prendre la valeur NULL. (Sauf, si celle-ci fait partie de la clé primaire)

Les clés primaires et NULL

Règle d'intégrité des données : les attributs appartenant à la clé primaire d'une table ne sont pas autorisés à prendre la valeur NULL.

  sdamy ingredient
 idIngredient : int(11)
 ingName : varchar(50)
 ingLocation : varchar(30)

L'attribut idIngredient ne peut pas prendre la valeur NULL

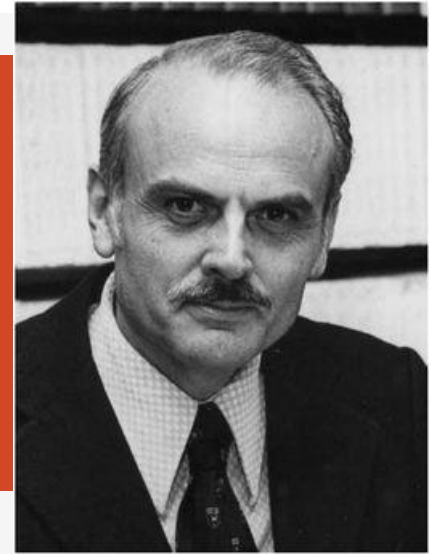
  sdamy composition
 idIngredient : int(11)
 idPotion : int(11)
 comQuantity : int(11)

L'attribut idIngredient ne peut pas prendre la valeur NULL

L'attribut idPotion ne peut pas prendre la valeur NULL

Partie 3

Modèle relationnel



Edgar Codd

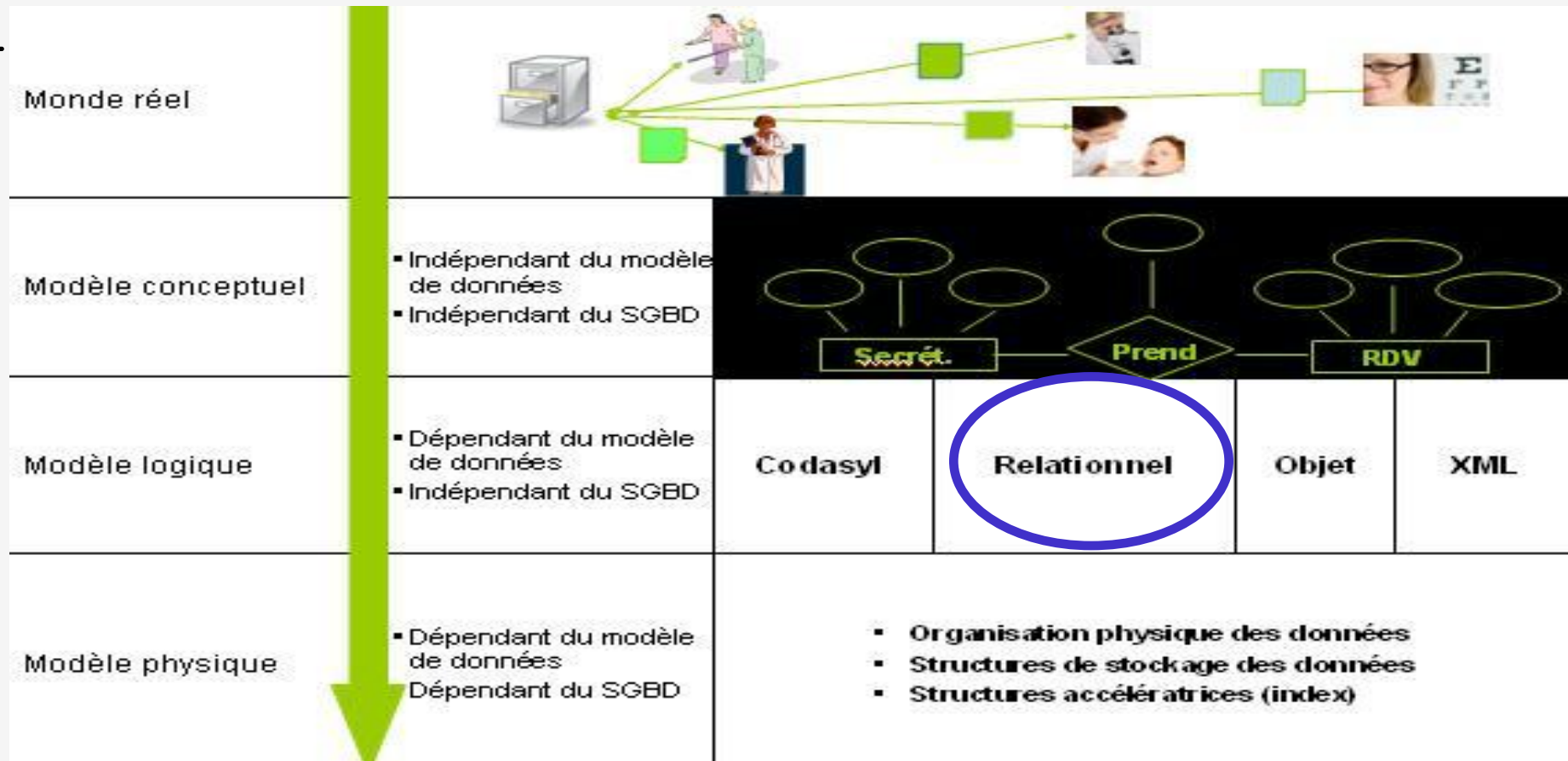
Définitions

Intégrité des données

Passage du MCD au modèle
relationnel

Passage du MCD au modèle relationnel

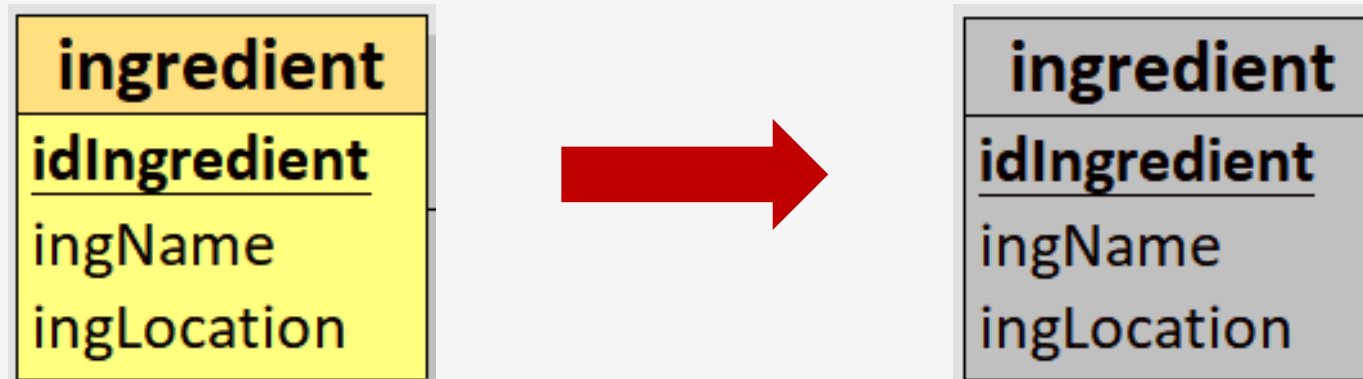
Modèle logique de données ou MLD : Modèle
relati



Les règles de transformation : Conversion d'une entité-type

Conversion d'une entité-type

- Chaque entité-type du MCD est représentée par une relation dont la clé est l'identifiant de l'entité
- Chaque attribut de l'entité-type devient un attribut de la relation



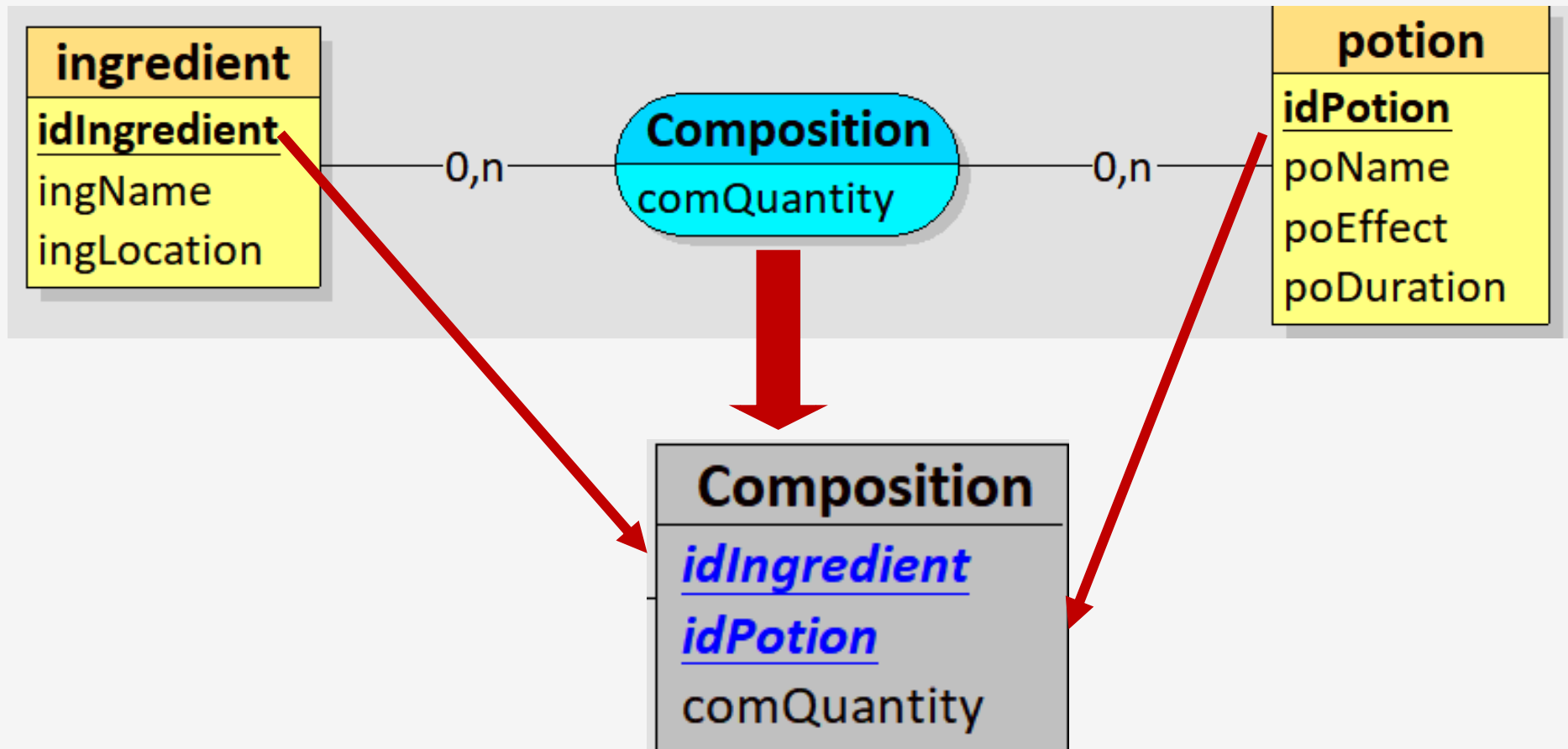
Les règles de transformation : Conversion d'une association-type

Chaque patte a une cardinalité maximale égale à n

- Devient une relation ou table
- Chaque attribut de l'association-type devient un attribut de la table
- Si l'association-type contient un identifiant il devient la clé primaire de la table
- Sinon la clé est formée de la concaténation des identifiants des entités-types qui interviennent dans l'association-type

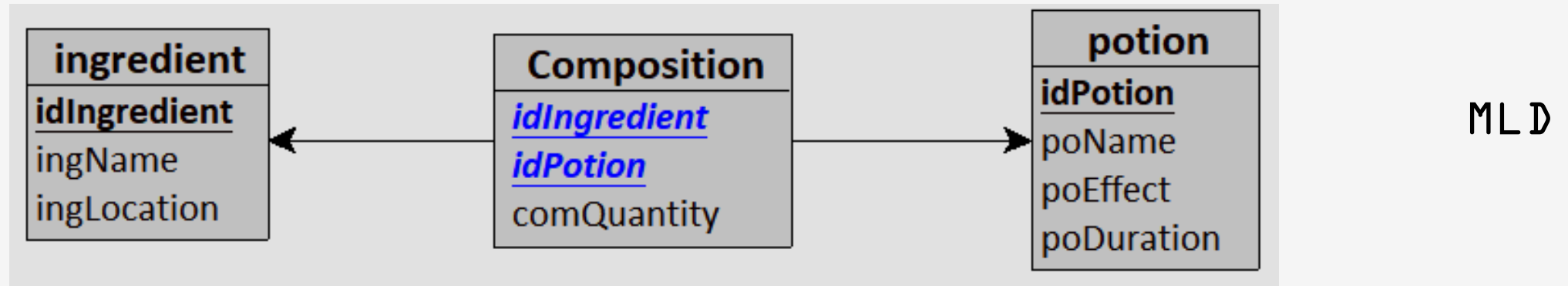
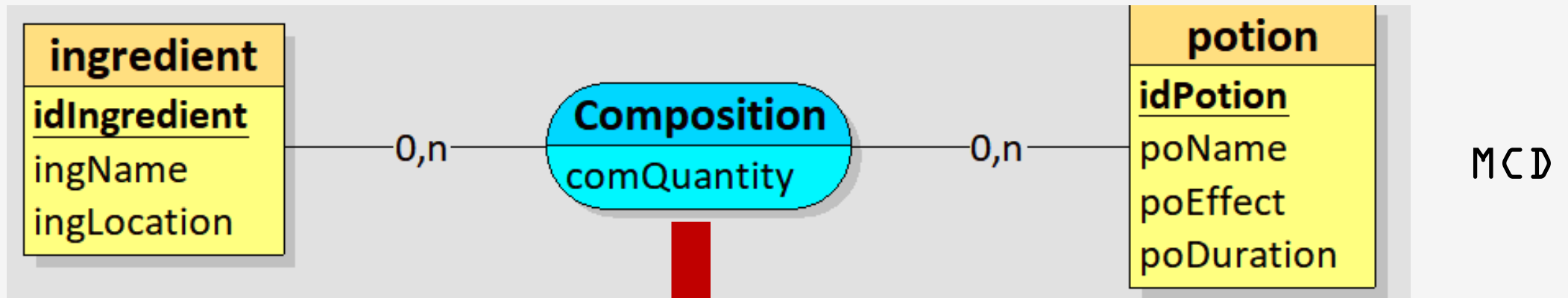
Les règles de transformation : Conversion d'une association-type

Chaque patte a une cardinalité maximale égale à n



Les règles de transformation : Conversion d'une association-type

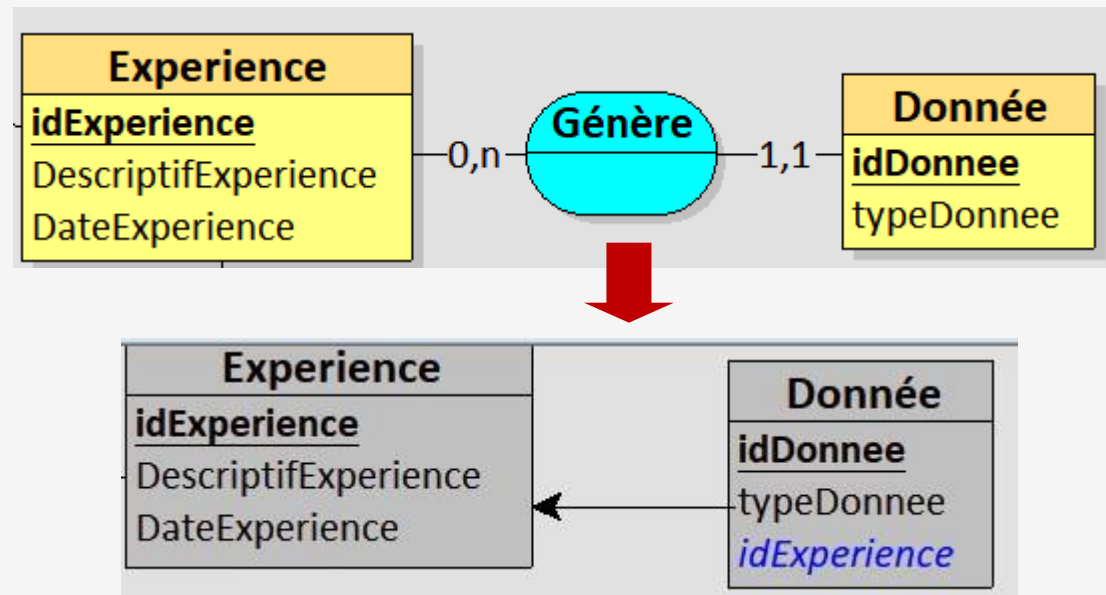
Chaque patte a une cardinalité maximale égale à n



Les règles de transformation : Conversion d'une association-type

Une patte a une cardinalité maximale égale à 1

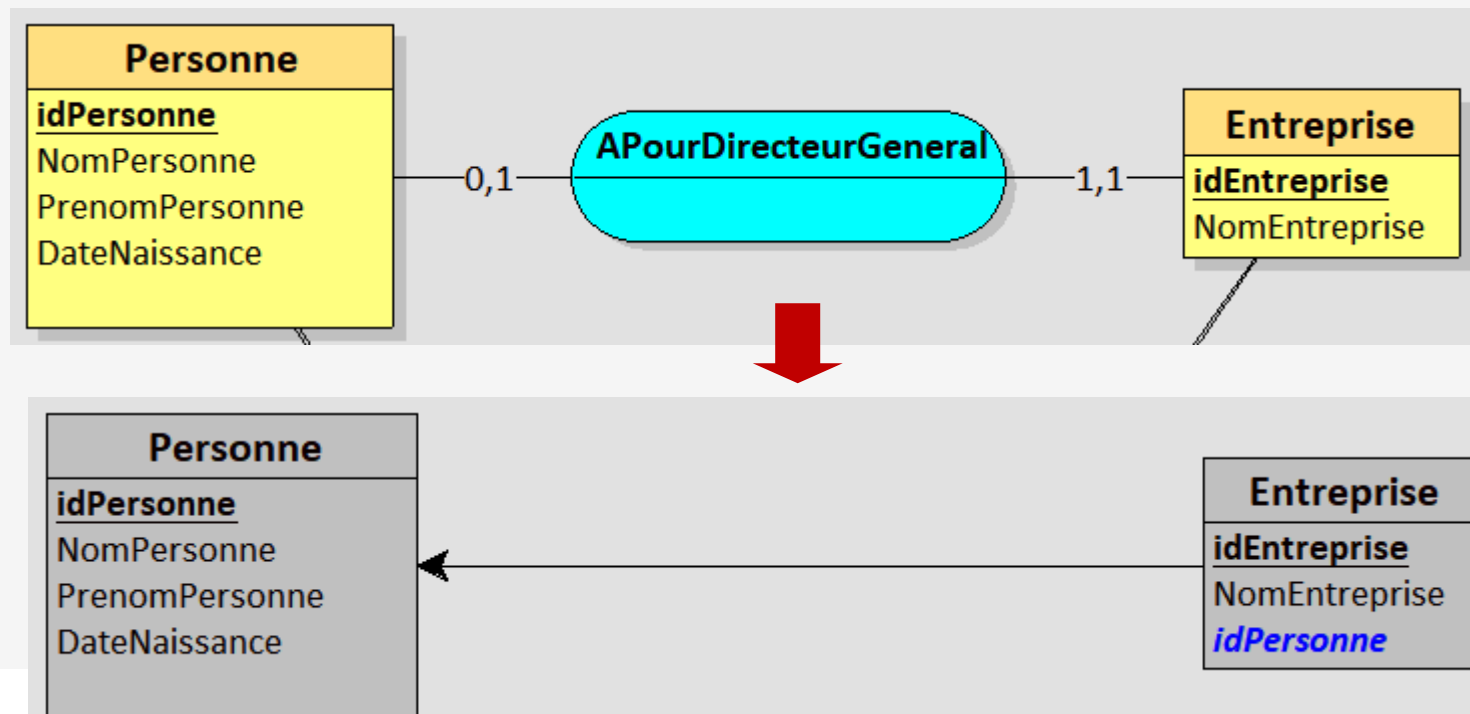
- Ne devient pas une relation
- On ajoute à la table correspondant à l'entité-type dont la patte vers l'association-type a la cardinalité maximale égale à 1, un attribut (clé étrangère) qui est l'identifiant de l'autre entité



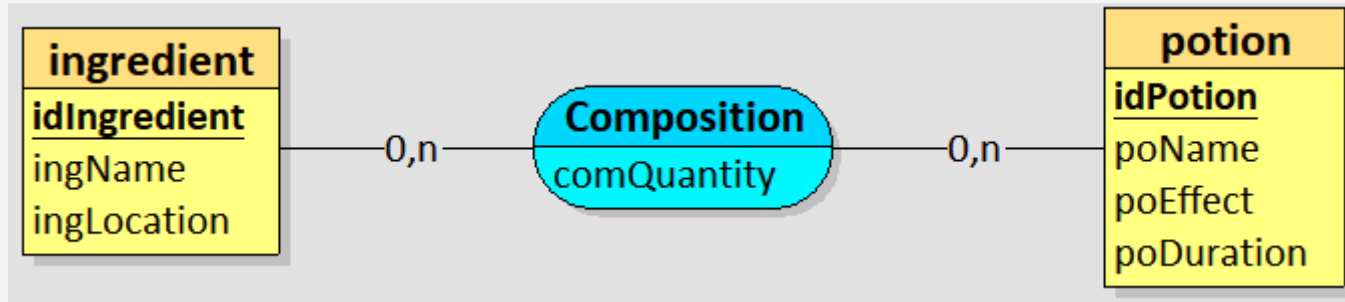
Les règles de transformation : Conversion d'une association-type

Les pattes ont une **cardinalité maximale égale à 1**

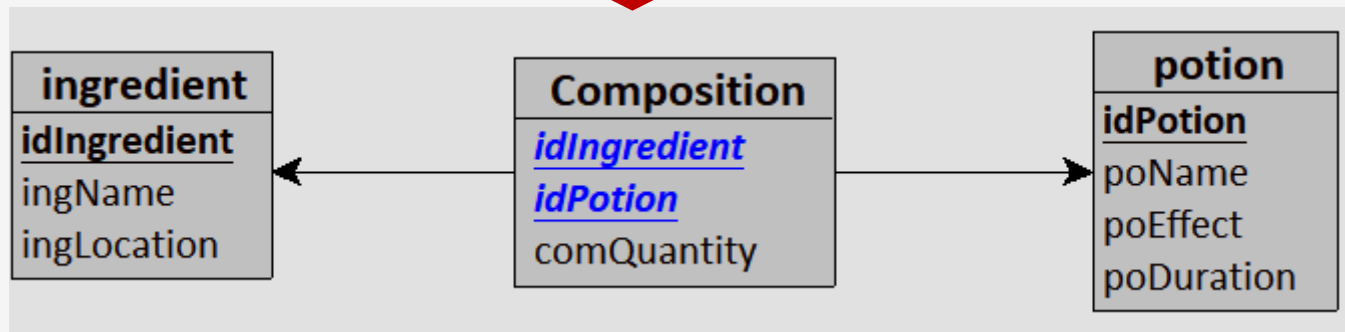
- Ne devient pas une relation
- On ajoute à la table correspondant à l'entité-type dont la patte vers l'association a la cardinalité minimale égale à 1, un attribut (clé étrangère) qui est l'identifiant de l'autre entité



Exemple des gauloiseries



MCD



MLD

Exemple des gauloiseries

Script de création de la base de données

The screenshot displays a database modeling software interface with three main panels:

- ER Diagram (MCD):** Shows three entities: **ingredient** (rectangle) with attributes idIngredient, ingName, and ingLocation; **potion** (rectangle) with attributes idPotion, poName, poEffect, and poDuration; and **Composition** (oval) with attribute comQuantity. There are two-to-many (0,n) relationships between ingredient and Composition, and between Composition and potion.
- SQL Script (SQL):**

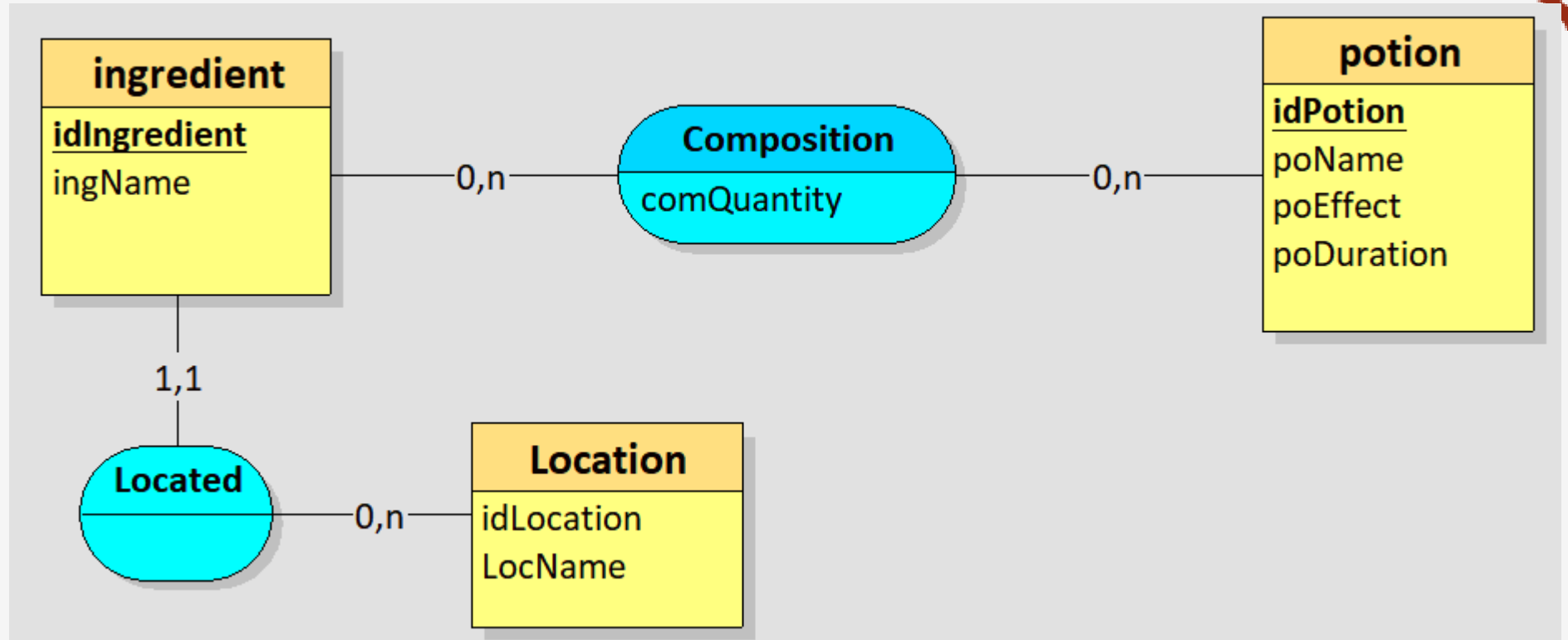
```
CREATE TABLE ingredient(  
  idIngredient COUNTER,  
  ingName VARCHAR(50),  
  ingLocation VARCHAR(50),  
  PRIMARY KEY(idIngredient)  
);  
  
CREATE TABLE potion(  
  idPotion COUNTER,  
  poName VARCHAR(50),  
  poEffect VARCHAR(50),  
  poDuration INT,  
  PRIMARY KEY(idPotion)  
);  
  
CREATE TABLE Composition(  
  idIngredient INT,  
  idPotion INT,  
  comQuantity INT,  
  PRIMARY KEY(idIngredient, idPotion),  
  FOREIGN KEY(idIngredient) REFERENCES ingredient(idIngredient),  
  FOREIGN KEY(idPotion) REFERENCES potion(idPotion)  
);
```
- MLD (Metadata Language):**

```
ingredient = (idIngredient COUNTER, ingName VARCHAR(50), ingLocation VARCHAR(50));  
potion = (idPotion COUNTER, poName VARCHAR(50), poEffect VARCHAR(50), poDuration INT);  
Composition = (#idIngredient, #idPotion, comQuantity INT);
```

Exemple



MCD



MLD



Partie 4 : SQL



La requête SELECT

SQL

SQL : Structured Query Language.

Langage conçu dans les années 1970 par IBM

Langage relationnel standard géré par presque tous les produits du marché

1^o version de SQL intégrée dans un SGBD en 1981
(ORACLE)

Historique de SQL

SQL1 : 1986 l'ANSI fait de SQL une norme ... SQL

2020+

Nous travaillerons avec la version de SQL installée
sur MySQL

Les sous-langages de SQL

DML : Data Manipulation Language

Permet d'extraire et de mettre à jour les données dans la base.

DDL : Data Definition Language

Permet de créer, modifier ou supprimer des tables, de définir les liens entre ces tables, leurs clés primaires.

DCL : Data Control Language

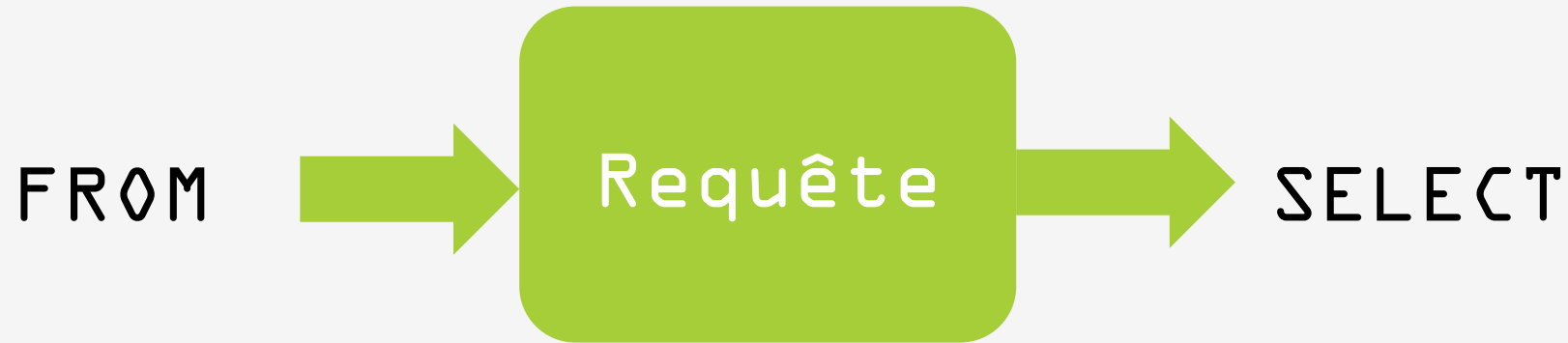
Permet de gérer des protections d'accès aux tables en environnement multi-utilisateurs.

L'obtention des données

Syntaxe de l'ordre **SELECT**

```
SELECT Liste-Attributs  
FROM Liste-tables  
WHERE critère de sélection  
GROUP BY liste-attributs-  
groupe  
HAVING critère-groupe  
ORDER BY Liste-attributs
```

Syntaxe de l'ordre SELECT



2 clauses
obligatoires :

- SELECT : sorties
- FROM : entrées

Nom des potions



Les informations qui nous intéressent

sdamy potion	
🔑	idPotion : int(11)
📄	poName : varchar(50)
📄	poEffect : varchar(50)
#	poDuration : int(11)

sdamy composition	
🔑	idIngredient : int(11)
🔑	idPotion : int(11)
#	comQuantity : int(11)

sdamy ingredient	
🔑	idIngredient : int(11)
📄	ingName : varchar(50)
📄	ingLocation : varchar(30)

Nom des potions



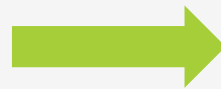
Les informations qui nous intéressent

v	⚙	sdamy potion
🔑		idPotion : int(11)
📄		poName : varchar(50)
📄		poEffect : varchar(50)
#		poDuration : int(11)

v	⚙	sdamy composition
🔑		idIngredient : int(11)
🔑		idPotion : int(11)
#		comQuantity : int(11)

v	⚙	sdamy ingredient
🔑		idIngredient : int(11)
📄		ingName : varchar(50)
📄		ingLocation : varchar(30)

potion



Requête



poName

Nom des potions



La requête

```
SELECT poName
FROM potion
```



idPotion	poName	poEffect	poDuration
1	Magic potion	Makes you strong	30
2	Happiness Potion	Makes people happy	120
3	Invisibility Potion	Makes you invisible	10
4	Anger potion	Makes you angry	5



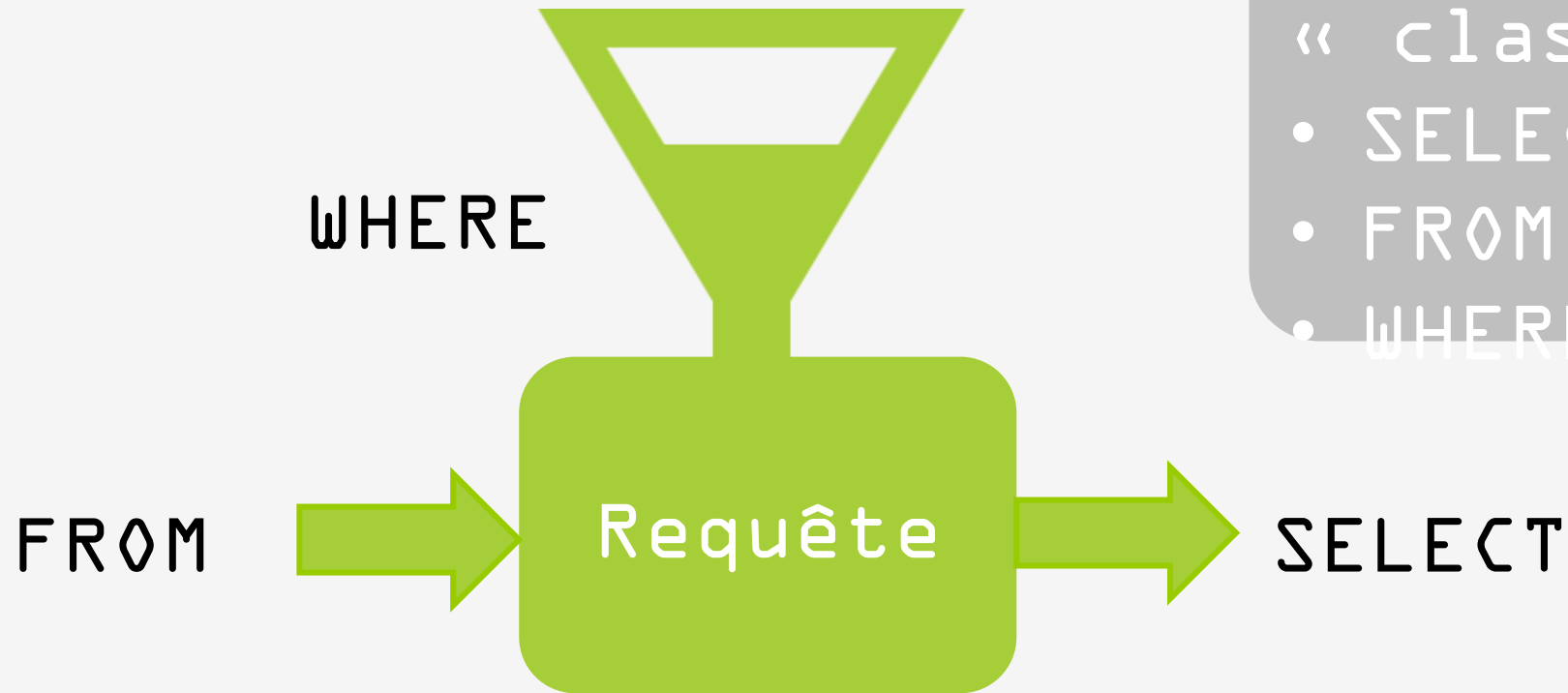
poName
Magic potion
Happiness Potion
Invisibility Potion
Anger potion

Syntaxe de l'ordre SELECT

Requête

« classique » :






- SELECT : sorties
- FROM : entrées
- WHERE : sélection











Nom des potions dont la durée > 20
minutes



Les informations qui nous intéressent

v  sdamy potion	
	idPotion : int(11)
	poName : varchar(50)
	poEffect : varchar(50)
	poDuration : int(11)

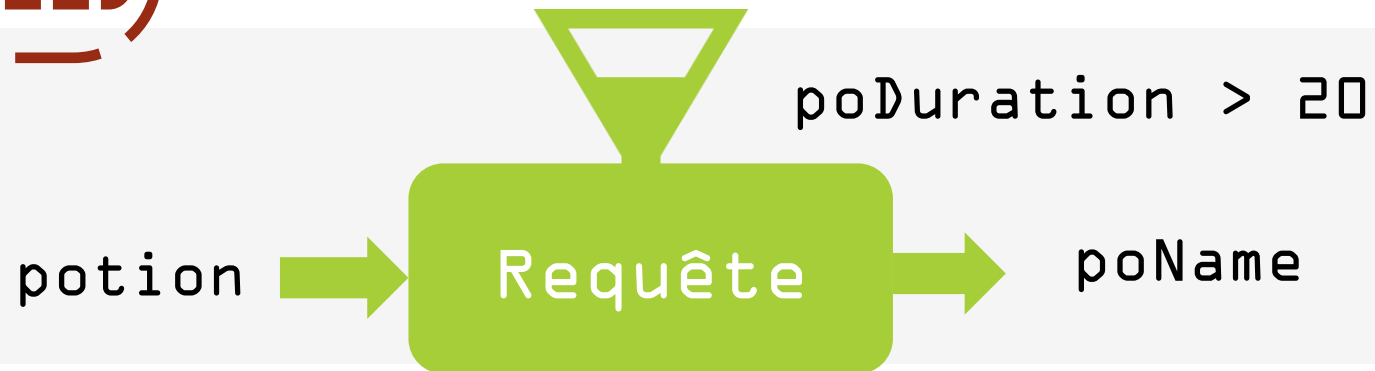
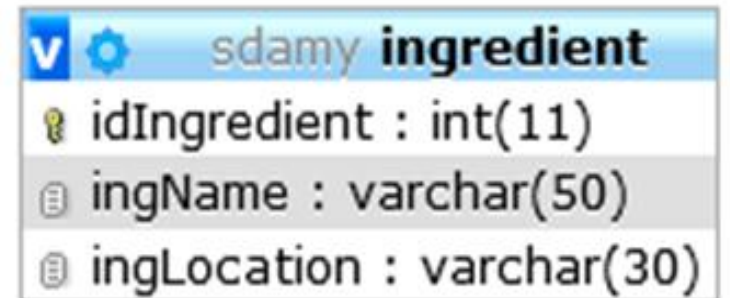
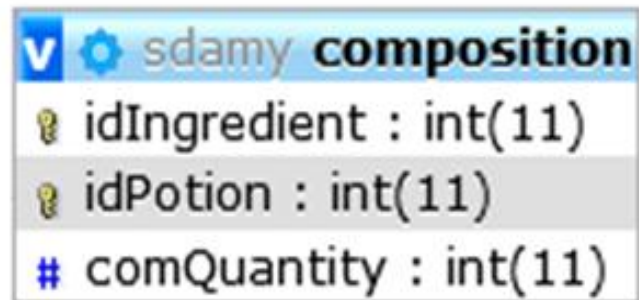
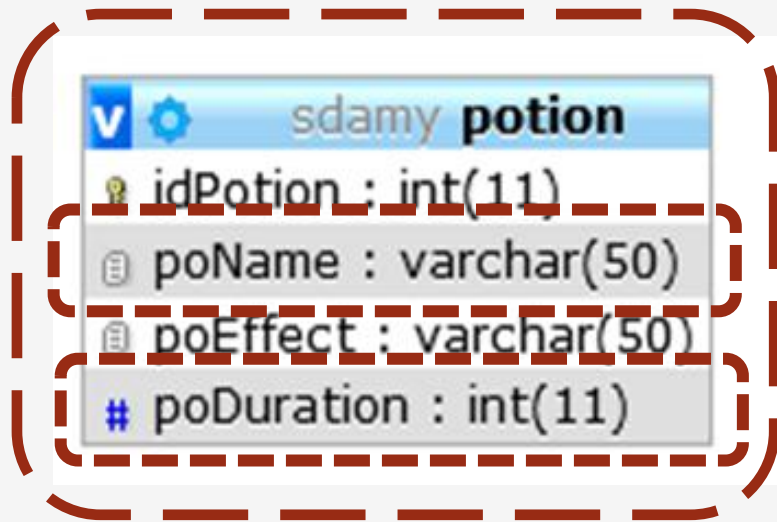
v  sdamy composition	
	idIngredient : int(11)
	idPotion : int(11)
	comQuantity : int(11)

v  sdamy ingredient	
	idIngredient : int(11)
	ingName : varchar(50)
	ingLocation : varchar(30)

Nom des potions dont la durée > 20 minutes



Les informations qui nous intéressent



Nom des potions dont la durée > 20
minutes



La requête

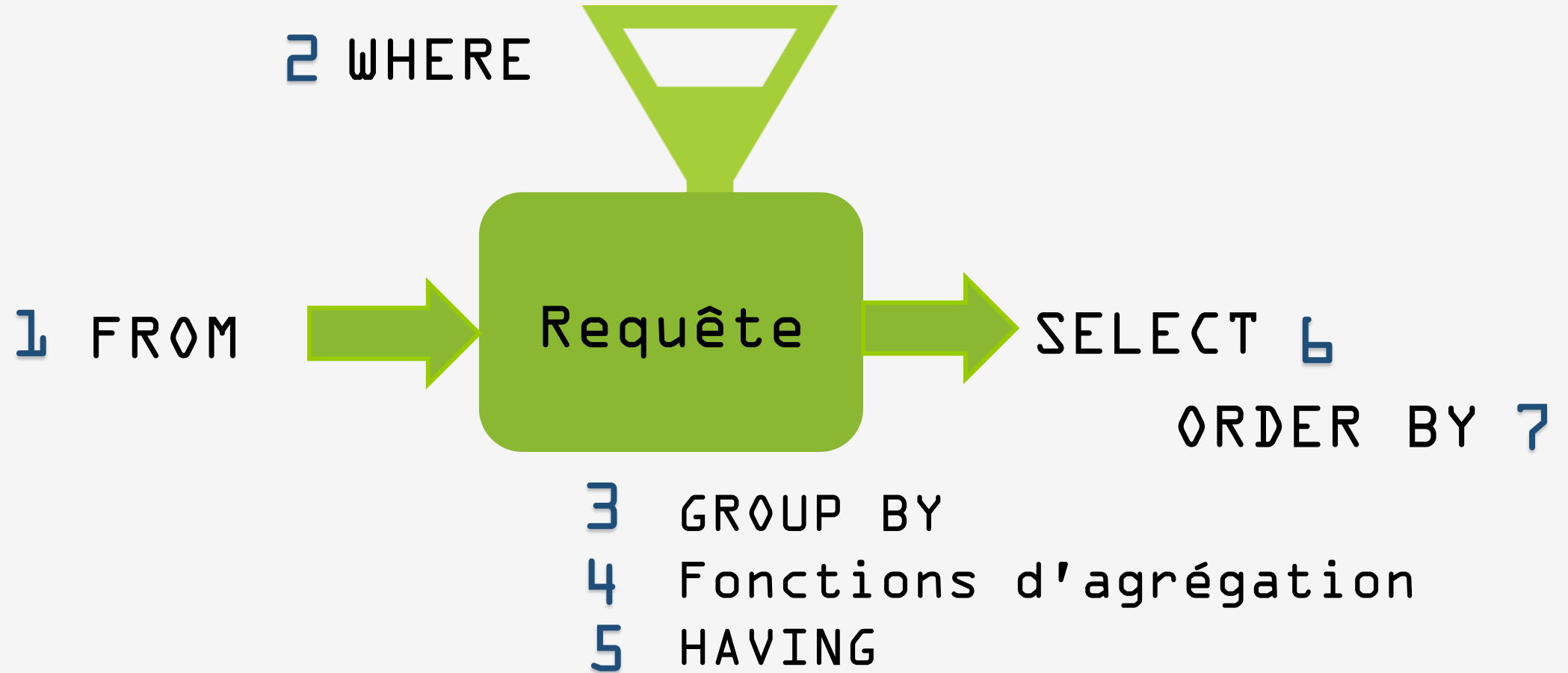
```
SELECT poName  
FROM potion  
WHERE poDuration > 20
```

idPotion	poName	poEffect	poDuration
1	Magic potion	Makes you strong	30
2	Happiness Potion	Makes people happy	120
3	Invisibility Potion	Makes you invisible	10
4	Anger potion	Makes you angry	5



poName
Magic potion
Happiness Potion

Evaluation d'une requête SELECT



La clause SELECT

Que retourne la requête

- Attributs
- Expressions

Syntaxe :

SQL est un langage informatique

Les expressions peuvent être des noms d'attributs : *idPotion*, *poName*,
... ou le résultat de fonctions : *sum*, *average*, ...

Syntaxe

```
SELECT [DISTINCT] expr1 [AS nom1], expr2 [AS  
nom2], ...
```

```
SELECT [DISTINCT] *
```

La clause SELECT

Mot réservé **DISTINCT** : élimination des doublons.

La clause SELECT : DISTINCT



idIngredient	ingName	ingLocation
1	water	anywhere
2	mistletoe	tree
3	Apple	Apple tree
4	Beer	hostel
5	Honey	forest
6	Lemon	Lemon tree
7	Raspberry	Garden
8	green salad	Garden

```
SELECT  
ingLocation  
FROM ingredient
```

```
ingLocation  
anywhere  
tree  
Apple tree  
hostel  
forest  
Lemon tree  
Garden  
Garden
```

```
SELECT DISTINCT  
ingLocation  
FROM ingredient
```

```
ingLocation  
anywhere  
tree  
Apple tree  
hostel  
forest  
Lemon tree  
Garden
```

La clause FROM

Permet :

- d'indiquer quelles tables ou requêtes sont utilisées par la requête,
- de définir une table qui est le produit cartésien des tables décrites,
- de décrire les jointures de tables.

Syntaxe

```
FROM table1 [AS nom1], table2 [AS nom2],  
...
```

La clause WHERE

Permet :

de spécifier les t-uplets à sélectionner dans une table ou dans le produit cartésien de plusieurs tables,

de définir le critère de sélection : un prédicat qui est évalué pour chaque t-uplet.

Syntaxe

```
WHERE critère de sélection (expr.  
booléenne)
```

Critères de sélection

Op. comparaison : <, >, >=, <=, =, <>

Opérateurs logiques : AND, OR, NOT

Prédicats servant à définir des ensembles : IN, BETWEEN
... AND ..., LIKE

Comparaison avec la valeur NULL : IS NULL, IS NOT NULL

Utilisation de caractères génériques :

- "_" (remplace 1 car),
- "%" (remplace une chaîne de car.)



Identifiant des potions qui ont dans leur composition l'ingrédient d'id 2

Les informations qui nous intéressent

sdamy potion	
🔑	idPotion : int(11)
📄	poName : varchar(50)
📄	poEffect : varchar(50)
#	poDuration : int(11)

sdamy composition	
🔑	idIngredient : int(11)
🔑	idPotion : int(11)
#	comQuantity : int(11)

sdamy ingredient	
🔑	idIngredient : int(11)
📄	ingName : varchar(50)
📄	ingLocation : varchar(30)



Identifiant des potions qui ont dans leur composition l'ingrédient d'id 2

Les informations qui nous intéressent → En entrée

sdamy potion	
🔑	idPotion : int(11)
📄	poName : varchar(50)
📄	poEffect : varchar(50)
#	poDuration : int(11)

sdamy composition	
🔑	idIngredient : int(11)
🔑	idPotion : int(11)
#	comQuantity : int(11)

sdamy ingredient	
🔑	idIngredient : int(11)
📄	ingName : varchar(50)
📄	ingLocation : varchar(30)

idIngredient = 2



Identifiant des potions qui ont dans leur composition l'ingrédient d'id 2

Les informations qui nous intéressent → En résultat

sdamy potion	
🔑	idPotion : int(11)
📄	poName : varchar(50)
📄	poEffect : varchar(50)
#	poDuration : int(11)

sdamy composition	
🔑	idIngredient : int(11)
🔑	idPotion : int(11)
#	comQuantity : int(11)

sdamy ingredient	
🔑	idIngredient : int(11)
📄	ingName : varchar(50)
📄	ingLocation : varchar(30)

idPotion



Identifiant des potions qui ont dans leur composition l'ingrédient d'id 2

Les informations qui nous intéressent → Table composition

sdamy potion	
idPotion	: int(11)
poName	: varchar(50)
poEffect	: varchar(50)
poDuration	: int(11)

sdamy composition	
idIngredient	: int(11)
idPotion	: int(11)
comQuantity	: int(11)

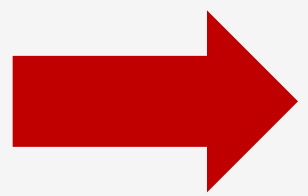
sdamy ingredient	
idIngredient	: int(11)
ingName	: varchar(50)
ingLocation	: varchar(30)



Identifiant des potions qui ont dans leur composition l'ingrédient d'id 2

Les informations qui nous intéressent → Table composition

idIngredient	idPotion	comQuantity
1	1	2
1	2	2
2	1	1
4	4	10
5	1	1



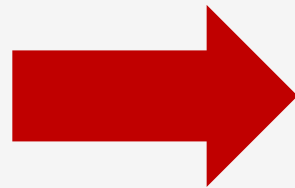
idPotion
1



Identifiant des potions qui ont dans leur composition l'ingrédient d'id 2

idIngredient	idPotion	comQuantity
1	1	2
1	2	2
2	1	1
4	4	10
5	1	1

```
SELECT idPotion
FROM composition
WHERE idIngredient =
2
```



idPotion
1

Identifiant des potions qui ont dans leur composition l'ingrédient d'id 2



Le résultat

```
SELECT idPotion  
FROM composition  
WHERE idIngredient = 2
```

La table
utilisée pour
la requête

Condition de sélection

Identifiant des potions qui ont dans leur composition l'ingrédient d'id 2



idIngredient	idPotion	comQuantity
1	1	2
1	2	2
2	1	1
4	4	10
5	1	1



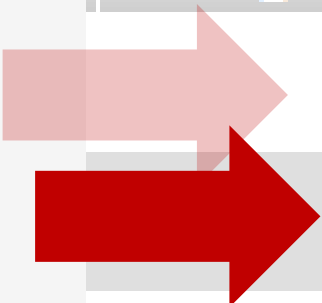
La clause WHERE teste la condition ligne par ligne

idIngredient <> 2
on ne garde pas la ligne

Identifiant des potions qui ont dans leur composition l'ingrédient d'id 2



idIngredient	idPotion	comQuantity
1	1	2
1	2	2
2	1	1
4	4	10
5	1	1



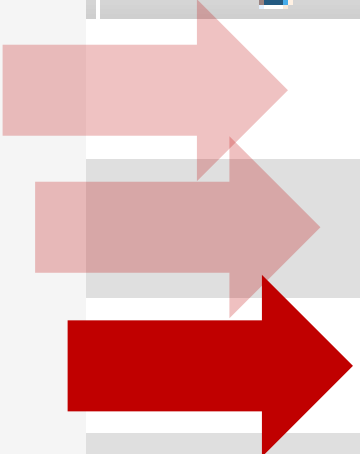
La clause WHERE teste la condition ligne par ligne

idIngredient <> 2
on ne garde pas la ligne

Identifiant des potions qui ont dans leur composition l'ingrédient d'id 2



idIngredient	idPotion	comQuantity
1	1	2
1	2	2
2	1	1
4	4	10
5	1	1



La clause WHERE teste la condition ligne par ligne

idIngredient = 2
on garde la ligne

Identifiant des potions qui ont dans leur composition l'ingrédient d'id 2



idIngredient	idPotion	comQuantity
1	1	2
1	2	2
2	1	1
4	4	10
5	1	1



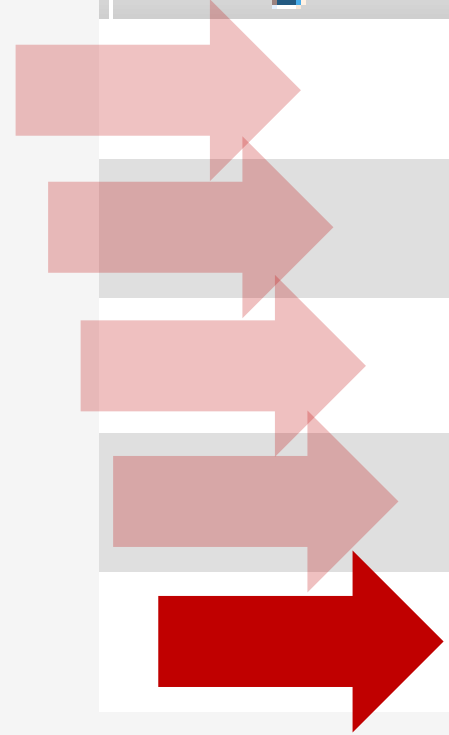
La clause WHERE teste la condition ligne par ligne

`idIngredient <> 2`
on ne garde pas la ligne

Identifiant des potions qui ont dans leur composition l'ingrédient d'id 2



idIngredient	idPotion	comQuantity
1	1	2
1	2	2
2	1	1
4	4	10
5	1	1



La clause WHERE teste la condition ligne par ligne

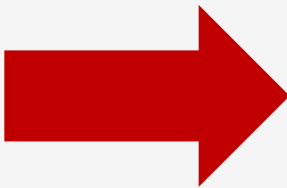
idIngredient <> 2
on ne garde pas la ligne

Identifiant des potions qui ont dans leur composition l'ingrédient d'id 2



idIngredient	idPotion	comQuantity
1	1	2
1	2	2
2	1	1
4	4	10
5	1	1

```
SELECT idPotion
FROM composition
WHERE idIngredient =
2
```



idPotion
1

Clause WHERE : exemples

```
SELECT *  
FROM composition  
WHERE idPotion = 1
```

idIngredient	idPotion	comQuantity
1	1	2
2	1	1
5	1	1

```
SELECT *  
FROM composition  
WHERE idPotion = 1  
AND comQuantity = 1
```

idIngredient	idPotion	comQuantity
2	1	1
5	1	1

Clause WHERE : examples

```
SELECT *  
FROM ingredient  
WHERE ingLocation =  
'Garden'
```

idIngredient	ingName	ingLocation
1	water	anywhere
2	mistletoe	tree
3	Apple	Apple tree
4	Beer	hostel
5	Honey	forest
6	Lemon	Lemon tree
7	Raspberry	Garden
8	green salad	Garden
9	Strawberry	NULL

idIngredient	ingName	ingLocation
7	Raspberry	Garden
8	green salad	Garden

Clause WHERE : examples

```
SELECT *  
FROM ingredient  
WHERE ingLocation =  
'Garden'  
OR ingLocation = 'tree'
```

```
SELECT *  
FROM ingredient  
WHERE ingLocation IN  
( 'Garden', 'tree' )
```

idIngredient	ingName	ingLocation
1	water	anywhere
2	mistletoe	tree
3	Apple	Apple tree
4	Beer	hostel
5	Honey	forest
6	Lemon	Lemon tree
7	Raspberry	Garden
8	green salad	Garden
9	Strawberry	NULL

idIngredient	ingName	ingLocation
2	mistletoe	tree
7	Raspberry	Garden
8	green salad	Garden

Clause WHERE : examples

```
SELECT idIngredient,  
       ingLocation  
FROM ingredient  
WHERE ingLocation Like 'a%'
```

idIngredient	ingName	ingLocation
1	water	anywhere
2	mistletoe	tree
3	Apple	Apple tree
4	Beer	hostel
5	Honey	forest
6	Lemon	Lemon tree
7	Raspberry	Garden
8	green salad	Garden
9	Strawberry	NULL

idIngredient	ingLocation
1	anywhere
3	Apple tree

Clause WHERE : examples

```
SELECT idIngredient, ingName
FROM ingredient
WHERE ingLocation IS NULL
```

idIngredient	ingName
9	Strawberry

idIngredient	ingName	ingLocation
1	water	anywhere
2	mistletoe	tree
3	Apple	Apple tree
4	Beer	hostel
5	Honey	forest
6	Lemon	Lemon tree
7	Raspberry	Garden
8	green salad	Garden
9	Strawberry	<i>NULL</i>

Noms des potions qui utilisent
l'ingrédient d'id 1



Pouvez-vous me donner le nom de ces potions ?

Noms des potions qui utilisent l'ingrédient d'id 1



Où sont les informations qui nous intéressent ?

sdamy potion	sdamy composition	sdamy ingredient
idPotion : int(11)	idIngredient : int(11)	idIngredient : int(11)
poName : varchar(50)	idPotion : int(11)	ingName : varchar(50)
poEffect : varchar(50)	comQuantity : int(11)	ingLocation : varchar(30)
poDuration : int(11)		

Noms des potions qui utilisent l'ingrédient d'id 1



idPotion	poName	poEffect	poDuration
1	Magic potion	Makes you strong	30
2	Happiness Potion	Makes people happy	120
3	Invisibility Potion	Makes you invisible	10
4	Anger potion	Makes you angry	5

idIngredient	idPotion	comQuantity
1	1	2
2	1	1
4	4	10
5	1	1

Noms des potions qui utilisent l'ingrédient d'id 1



idPotion	poName	poEffect	poDuration
1	Magic potion	Makes you strong	30
2	Happiness Potion	Makes people happy	120
3	Invisibility Potion	Makes you invisible	10
4	Anger potion	Makes you angry	5

idIngredient	idPotion	comQuantity
1	1	2
2	1	1
4	4	10
5	1	1

Noms des potions qui utilisent l'ingrédient d'id 1



idPotion	poName	poEffect	poDuration
1	Magic potion	Makes you strong	30
2	Happiness Potion	Makes people happy	120
3	Invisibility Potion	Makes you invisible	10
4	Anger potion	Makes you angry	5

idIngredient	idPotion	comQuantity
1	1	2
2	1	1
4	4	10
5	1	1

poName
Magic potion



Comment obtenez-vous ce résultat ?

Noms des potions qui utilisent
l'ingrédient d'id 1



Comment avez-vous fait ?

Vous avez « jointuré » les deux tables

Requêtes avec plusieurs tables

Modèle relationnel : plusieurs tables

Extraire des données d'une BD

→ extraire des données dans différentes tables

→ il faut recomposer l'information

→ Opérateur de jointure

La jointure en SQL

Un modèle relationnel est constitué de plusieurs tables et pour extraire des données d'une BD il est très souvent nécessaire d'extraire des données provenant de différentes tables.

La jointure permet de travailler avec plusieurs tables

Une jointure met en relation deux tables en utilisant une clause de jointure

La jointure en SQL

Les jointures indiquent comment on lie les t-uplets d'une table avec les t-uplets d'une autre table.

Une condition de jointure définit la manière dont deux tables sont liées dans une requête :

- en spécifiant la colonne de chaque table à utiliser pour la jointure.
- en spécifiant un opérateur de comparaison des valeurs des colonnes.

La jointure en SQL

Différentes versions de SQL : différentes expressions de cet opérateur

- Dans la clause WHERE
- L'ordre JOIN dans la clause FROM

Jointure dans la clause WHERE

Permet de réaliser une jointure interne, mais pas externe

Syntaxe

```
SELECT ...  
  FROM R, S  
  WHERE condition de  
jointure
```

Noms des potions qui utilisent
l'ingrédient d'id 1



```
SELECT poName
FROM composition, potion
WHERE idIngredient = 1
AND composition.idPotion =
potion.idPotion
```

Condition de jointure

poName
Magic potion

Noms des acteurs jouant dans le film de numéro 1



Jointure

`composition.idPotion=potion.idPot`



idIngredient	idPotion	comQuantity	idPotion	poName	poEffect	poDuration
1	1	2	1	Magic potion	Makes you strong	30
2	1	1	1	Magic potion	Makes you strong	30
4	4	10	1	Magic potion	Makes you strong	30
5	1	1	1	Magic potion	Makes you strong	30
1	1	2	2	Happiness Potion	Makes people happy	120
2	1	1	2	Happiness Potion	Makes people happy	120
4	4	10	2	Happiness Potion	Makes people happy	120
5	1	1	2	Happiness Potion	Makes people happy	120
1	1	2	3	Invisibility Potion	Makes you invisible	10
2	1	1	3	Invisibility Potion	Makes you invisible	10
4	4	10	3	Invisibility Potion	Makes you invisible	10
5	1	1	3	Invisibility Potion	Makes you invisible	10
1	1	2	4	Anger potion	Makes you angry	5
2	1	1	4	Anger potion	Makes you angry	5
4	4	10	4	Anger potion	Makes you angry	5
5	1	1	4	Anger potion	Makes you angry	5

Produit cartésien

idIngredient	idPotion	comQuantity	idPotion	poName	poEffect	poDuration
1	1	2	1	Magic potion	Makes you strong	30
2	1	1	1	Magic potion	Makes you strong	30
5	1	1	1	Magic potion	Makes you strong	30
4	4	10	4	Anger potion	Makes you angry	5

Sélection

`idIngredient=1`

idIngredient	idPotion	comQuantity	idPotion	poName	poEffect	poDuration
1	1	2	1	Magic potion	Makes you strong	30

Projection

`SELECT poName`

poName

Magic potion

Jointure avec l'opérateur JOIN

Permet de réaliser une jointure interne, ou externe

Syntaxe

```
SELECT ...  
  FROM R Op jointure S  
  ON condition de  
  jointure
```

Jointure interne : INNER JOIN ou JOIN

Jointure externe : OUTER JOIN

- LEFT OUTER JOIN
- RIGHT OUTER JOIN

Noms des potions qui utilisent
l'ingrédient d'id 1



```
SELECT poName
FROM composition JOIN potion ON
      composition.idPotion =
potion.idPotion
WHERE idIngredient = 1
```

Condition de jointure

Noms des acteurs jouant dans le film de numéro 1



Jointure

`composition.idPotion=potion.idPot`



idPotion	poName	poEffect	poDuration
1	Magic potion	Makes you strong	30
2	Happiness Potion	Makes people happy	120
3	Invisibility Potion	Makes you invisible	10
4	Anger potion	Makes you angry	5

idIngredient	idPotion	comQuantity
1	1	2
2	1	1
4	4	10
5	1	1

idIngredient	idPotion	comQuantity	idPotion	poName	poEffect	poDuration
1	1	2	1	Magic potion	Makes you strong	30
2	1	1	1	Magic potion	Makes you strong	30
5	1	1	1	Magic potion	Makes you strong	30
4	4	10	4	Anger potion	Makes you angry	5

Sélection

`idIngredient=1`

idIngredient	idPotion	comQuantity	idPotion	poName	poEffect	poDuration
1	1	2	1	Magic potion	Makes you strong	30

Projection

`SELECT poName`

poName

Magic potion

Jointure externe

La jointure externe permet de récupérer toutes les données des tables, même si certaines données n'ont pas de correspondance dans l'autre table.

- Jointure interne : JOIN
- Jointure externe :
 - LEFT JOIN
 - RIGHT JOIN

Nom des ingrédients et id des potions qui les utilisent



Utilisation de 2 tables → Jointure

sdamy potion	
🔑	idPotion : int(11)
📄	poName : varchar(50)
📄	poEffect : varchar(50)
#	poDuration : int(11)

sdamy composition	
🔑	idIngredient : int(11)
🔑	idPotion : int(11)
#	comQuantity : int(11)

sdamy ingredient	
🔑	idIngredient : int(11)
📄	ingName : varchar(50)
📄	ingLocation : varchar(30)

Nom des ingrédients et id des potions qui les utilisent



Utilisation de 2 tables → Jointure

sdamy potion	
🔑	idPotion : int(11)
📄	poName : varchar(50)
📄	poEffect : varchar(50)
#	poDuration : int(11)

sdamy composition	
🔑	idIngredient : int(11)
🔑	idPotion : int(11)
#	comQuantity : int(11)

sdamy ingredient	
🔑	idIngredient : int(11)
📄	ingName : varchar(50)
📄	ingLocation : varchar(30)

`composition.idIngredient = ingredient.idIngredient`

Nom des ingrédients et id des potions qui les utilisent



composition

idIngredient	idPotion	comQuantity
1	1	2
2	1	1
4	4	10
5	1	1

ingredient

idIngredient	ingName	ingLocation
1	water	anywhere
2	mistletoe	tree
3	Apple	Apple tree
4	Beer	hostel
5	Honey	forest
6	Lemon	Lemon tree
7	Raspberry	Garden
8	green salad	Garden

Jointure interne :

`composition.idIngredient = ingredient.idIngredient`

Nom des ingrédients et id des potions qui les utilisent



composition

idIngredient	idPotion	comQuantity
1	1	2
2	1	1
4	4	10
5	1	1

ingredient

idIngredient	ingName	ingLocation
1	water	anywhere
2	mistletoe	tree
3	Apple	Apple tree
4	Beer	hostel
5	Honey	forest
6	Lemon	Lemon tree
7	Raspberry	Garden
8	green salad	Garden
9	Strawberry	NULL

```
SELECT ingName, idPotion
FROM composition JOIN ingredient
ON composition.idIngredient=ingredient.idIngredient
```



ingName	idPotion
water	1
mistletoe	1
Honey	1
Beer	4

Nom des ingrédients et id des potions qui les utilisent



composition

idIngredient	idPotion	comQuantity
1	1	2
2	1	1
4	4	10
5	1	1

ingredient

idIngredient	ingName	ingLocation
1	water	anywhere
2	mistletoe	tree
3	Apple	Apple tree
4	Beer	hostel
5	Honey	forest
6	Lemon	Lemon tree
7	Raspberry	Garden
8	green salad	Garden
9	Strawberry	NULL

Ces ingrédients n'apparaissent pas dans le résultat

```
SELECT ingName, idPotion
FROM composition JOIN ingredient
ON composition.idIngredient=ingredient.idIngredient
```



ingName	idPotion
water	1
mistletoe	1
Honey	1
Beer	4

Nom des ingrédients et id des potions qui les utilisent



composition

idIngredient	idPotion	comQuantity
1	1	2
2	1	1
4	4	10
5	1	1

ingredient

idIngredient	ingName	ingLocation
1	water	anywhere
2	mistletoe	tree
3	Apple	Apple tree
4	Beer	hostel
5	Honey	forest
6	Lemon	Lemon tree
7	Raspberry	Garden
8	green salad	Garden
9	Strawberry	NULL

```
SELECT ingName, idPotion
FROM composition RIGHT JOIN ingredient
ON composition.idIngredient=ingredient.idIngredient
```

Nom des ingrédients et id des potions qui les utilisent



composition

idIngredient	idPotion	comQuantity
1	1	2
2	1	1
4	4	10
5	1	1

ingredient

idIngredient	ingName	ingLocation
1	water	anywhere
2	mistletoe	tree
3	Apple	Apple tree
4	Beer	hostel
5	Honey	forest
6	Lemon	Lemon tree
7	Raspberry	Garden
8	green salad	Garden
9	Strawberry	NULL

```
SELECT ingName, idPotion
FROM composition RIGHT JOIN ingredient
ON composition.idIngredient=ingredient.idIngredient
```

Nom des ingrédients et id des potions qui les utilisent



composition

idIngredient	idPotion	comQuantity
1	1	2
2	1	1
4	4	10
5	1	1

ingredient

idIngredient	ingName	ingLocation
1	water	anywhere
2	mistletoe	tree
3	Apple	Apple tree
4	Beer	hostel
5	Honey	forest
6	Lemon	Lemon tree
7	Raspberry	Garden
8	green salad	Garden

ingName	idPotion
water	1
mistletoe	1
Apple	NULL
Beer	4
Honey	1
Lemon	NULL
Raspberry	NULL
green salad	NULL
Strawberry	NULL

```
SELECT ingName, idPotion
FROM composition RIGHT JOIN ingredient
ON composition.idIngredient=ingredient.idIngredient
```

Jointure externe

```
          LEFT          RIGHT  
FROM table1 ??? JOIN table2
```

LEFT : jointure externe par rapport à table1

RIGHT : jointure externe par rapport à table2

Lieux de récolte des ingrédients de la potion 1



A vous de faire

sdamy potion	
🔑	idPotion : int(11)
📄	poName : varchar(50)
📄	poEffect : varchar(50)
#	poDuration : int(11)

sdamy composition	
🔑	idIngredient : int(11)
🔑	idPotion : int(11)
#	comQuantity : int(11)

sdamy ingredient	
🔑	idIngredient : int(11)
📄	ingName : varchar(50)
📄	ingLocation : varchar(30)

Les fonctions de groupe

Fonctions d'agrégation :
permettent d'obtenir des
informations sur un
ensemble de t-uplets
dans une table

On travaille sur les
colonnes et non sur les
lignes

Syntaxe

nom-de-la-
fonction(paramètre)

	Nom
Valeur moyenne	AVG
Somme	SUM
Plus petite valeur	MIN
Plus grande valeur	MAX
Variance	VARIANCE
Ecart type	STDDEV
de	COUNT

Durée la plus longue des potions



```
SELECT MAX(poDuration)  
FROM potion
```

max(poDuration)
120

idPotion	poName	poEffect	poDuration
1	Magic potion	Makes you strong	30
2	Happiness Potion	Makes people happy	120
3	Invisibility Potion	Makes you invisible	10
4	Anger potion	Makes you angry	5

```
SELECT MAX(poDuration) AS  
MaxDuration  
FROM potion
```

MaxDuration
120

Dernier nom de potion par ordre
alphabétique



```
SELECT MAX(poName)
FROM potion
```

max(poName)
Magic potion

Premier et dernier nom de potion par ordre alphabétique

```
SELECT MIN(poName), MAX(poName)
FROM potion
```

MIN(poName)	MAX(poName)
Anger potion	Magic potion

Nombre de potions



```
SELECT COUNT(*) AS Nbre_Potions  
FROM potion
```

idPotion	poName	poEffect	poDuration
1	Magic potion	Makes you strong	30
2	Happiness Potion	Makes people happy	120
3	Invisibility Potion	Makes you invisible	10
4	Anger potion	Makes you angry	5



Nbre_Potions
4

Nombre d'ingrédients récoltés dans le jardin (garden)



```
SELECT COUNT(idIngredient)  
FROM ingredient  
WHERE ingLocation = 'garden'
```

idIngredient	ingName	ingLocation
1	water	anywhere
2	mistletoe	tree
3	Apple	Apple tree
4	Beer	hostel
5	Honey	forest
6	Lemon	Lemon tree
7	Raspberry	Garden
8	green salad	Garden

COUNT(idIngredient)

2

Particularités de la fonction COUNT

- `COUNT(*)` : compte le nombre de t-uplets obtenus dans la requête (lignes)
- `COUNT(attribut)` : compte le nombre de valeurs de l'attribut différentes de NULL
- `COUNT(DISTINCT attribut)` : compte le nombre de valeurs différentes de l'attribut

`COUNT(DISTINCT attribut)` n'existe pas dans tous les SGBD



Utilisations de la fonction COUNT

```
SELECT count(*)  
FROM ingredient
```

```
count(*)  
9
```

```
SELECT count(ingLocation) AS NB_Location  
FROM ingredient
```

```
NB_Location  
8
```

```
SELECT count(Distinct ingLocation) AS  
NB_Location  
FROM ingredient
```

```
NB_Location  
7
```

idIngredient	ingName	ingLocation
1	water	anywhere
2	mistletoe	tree
3	Apple	Apple tree
4	Beer	hostel
5	Honey	forest
6	Lemon	Lemon tree
7	Raspberry	Garden
8	green salad	Garden
9	Strawberry	NULL

Pour chaque potion : son nombre d'ingrédients



sdamy potion	
🔑	idPotion : int(11)
📄	poName : varchar(50)
📄	poEffect : varchar(50)
#	poDuration : int(11)

sdamy composition	
🔑	idIngredient : int(11)
🔑	idPotion : int(11)
#	comQuantity : int(11)

sdamy ingredient	
🔑	idIngredient : int(11)
📄	ingName : varchar(50)
📄	ingLocation : varchar(30)

Où sont les informations qui nous intéressent ?

Pour chaque potion : son nombre d'ingrédients



sdamy potion	sdamy composition	sdamy ingredient
idPotion : int(11)	idIngredient : int(11)	idIngredient : int(11)
poName : varchar(50)	idPotion : int(11)	ingName : varchar(50)
poEffect : varchar(50)	comQuantity : int(11)	ingLocation : varchar(30)
poDuration : int(11)		

Où sont les informations qui nous intéressent ?

Pour chaque potion : son nombre
d'ingrédients

idIngredient	idPotion	comQuantity
1	1	2
2	1	1
4	4	10
5	1	1

idPotion	nbre_Ingredients
1	3
4	1

La clause GROUP BY

Permet de subdiviser la table en groupes

Une seule ligne représente l'ensemble des t-uplets de la table regroupés, tous les t-uplets regroupés ont la même valeur pour les expressions du regroupement

Souvent utilisée avec une fonction de groupe

Syntaxe

```
GROUP BY expr1, expr2, ...
```

Pour chaque potion : son nombre
d'ingrédients



Comment regrouper ?

Par potion : idPotion

Choisir pour le regroupement un attribut
significatif (ici la clé de potion)

Compter combien on a d'ingrédients dans chaque groupe

Pour chaque potion : son nombre d'ingrédients



idIngredient	idPotion	comQuantity
1	1	2
2	1	1
4	4	10
5	1	1

idIngredient	idPotion	comQuantity
1, 2, 5	1	2, 1, 1
4	4	1

Regroupement potion :
idPotion

Pour chaque potion : son nombre
d'ingrédients



```
SELECT idPotion, count(*)  
FROM composition  
GROUP BY idPotion
```

idPotion	count(*)
1	3
4	1

Les restrictions de la clause GROUP BY

Les seules opérations réalisables sur une "*table regroupée*" sont :

- opérations qui prennent en compte un ensemble de lignes (fonctions de groupe)
- opérations de projection portant sur des attributs impliqués dans le regroupement

Les restrictions de la clause GROUP BY



Regroupement

idIngredient	idPotion	comQuantity
1	1	2
2	1	1
4	4	10
5	1	1

Fonctions de groupe

La clause HAVING

- Permet de sélectionner des groupes définis par la clause GROUP BY
- Même syntaxe que le prédicat de la clause WHERE
- Ne porte que sur les caractéristiques du groupe

Syntaxe

HAVING critère de sélection de groupes

Potions dont le nombre d'ingrédients est supérieur à 2



```
SELECT idPotion, count(*)  
FROM composition  
GROUP BY idPotion
```

idPotion	count(*)
1	3
4	1

Repartons de la requête vue précédemment qui compte les ingrédients par potion

Potions dont le nombre d'ingrédients est supérieur à 2



```
SELECT idPotion
FROM composition
GROUP BY idPotion
HAVING count(*) > 2
```

idIngredient	idPotion	comQuantity
1	1	2
2	1	1
4	4	10
5	1	1

idPotion	count(*)
1	3
4	1

idPotion
1

Les restrictions de la clause HAVING

Liées aux restrictions de la clause GROUP BY

Les seules expressions réalisables dans une clause HAVING sont :

- des expressions qui utilisent des fonctions de groupe sur les attributs qui ne font pas partie du regroupement,
- des expressions utilisant sous une forme quelconque les attributs du regroupement

Les restrictions de la clause HAVING



Regroupement

idIngredient	idPotion	comQuantity
1	1	2
2	1	1
4	4	10
5	1	1

Fonctions de groupe

```
GROUP BY idPotion  
HAVING  
    count(distinct idIngredient) > 3
```



```
GROUP BY idPotion  
HAVING  
    count(comQuantity) > 2
```



```
GROUP BY idPotion  
HAVING comQuantity =  
2
```



La clause ORDER BY

Permet de préciser dans quel ordre les t-uplets sélectionnés seront donnés.

Le tri peut s'effectuer selon plusieurs critères, en allant du premier critère au dernier.

Syntaxe

```
ORDER BY expr1 [DESC, ASC] [, expr2 [DESC,  
ASC], ...]
```

Potion par ordre croissant de potion



```
SELECT *  
FROM potion  
ORDER BY poName
```

idPotion	poName ▲ 1	poEffect	poDuration
4	Anger potion	Makes you angry	5
2	Happiness Potion	Makes people happy	120
3	Invisibility Potion	Makes you invisible	10
1	Magic potion	Makes you strong	30

Par défaut : ordre
croissant

Les compositions par quantité et id d'ingrédient



```
SELECT *  
FROM composition  
ORDER BY comQuantity, idIngredient
```

idIngredient	idPotion	comQuantity
2	1	1
5	1	1
1	1	2
4	4	10

Et maintenant à vous de
jouer !!!

