Introduction aux bases de données relationnelles.

I. Exemple .

Nous allons imaginer un opérateur de téléphonie qui veut créer une base de données pour gérer ses clients. Les clients peuvent obtenir plusieurs numéros de téléphone auprès de cet opérateur. Pour chacun de ces numéros, l'opérateur enregistrera les informations nécessaires pour établir une facture mensuelle détaillée.

Quelles sont ces données nécessaires ? Il faudra tout d'abord identifier de manière non équivoque le client : on enregistrera donc son nom, son prénom et son adresse. On stockera aussi le numéro de téléphone qu'il a utilisé (rappelons qu'il peut en avoir plusieurs ; on fera une facture par numéro). Il faudra aussi connaître, pour calculer les prix de la communication, le numéro appelé, la date et l'heure de l'appel (pour établir la facture détaillée) ainsi que la durée de l'appel. Enfin, comme le prix de l'appel peut varier selon l'heure et le numéro appelé, on mentionnera le tarif appliqué.

1. Première approche : un tableur

Une première idée pour implémenter une base de données consiste à utiliser un tableur. En effet, une base de données « plate » peut se voir comme un simple tableau : les colonnes représenteront des champs (nom, prénom, numéro de téléphone, etc.), et les lignes des enregistrements (ici des appels).
Par exemple, voici un extrait de cette base de données :

Nom	Prénom	Adresse	Numéro de tél.	No appelé	Date et heure de l'appel	Durée de l'appel (sec)	Tarif (CHF/min)
Müller	Didier	Petit-Chêne 3 2900 Porrentruy	0324660010	00333246634217	10.2.2010 14:32	455	0.70
Müller	Didier	Petit-Chêne 3 2900 Porrentruy	0324660010	0324711230	17.2.2010 18:37	62	0.20
Müller	Didier	Petit-Chêne 3 2900 Porrentruy	0324669987	0324711230	23.2.2010 9:21	145	0.20
Müller	Didier	Grand-Rue 49 2800 Delémont	0324221022	0214537124	12.2.2010 19:01	302	0.10
...	...	...	...	...	...	...	

L'avantage de cette méthode est sa simplicité : tout ce qui concerne un appel tient sur une ligne, et tous les appels sont répertoriés dans une table.

Cette approche a aussi des inconvénients :

2. Deuxième approche : base de données relationnelle.

Une autre solution est d'utiliser plusieurs tables reliées entre elles. On parle alors de base de données relationnelle. La liaison de différentes tables se fera en utilisant un logiciel de gestion de bases de données (SGBD).
Reprenons l'exemple de l'entreprise de téléphonie. On va utiliser trois tables au lieu d'une : la première contiendra les coordonnées des clients, la seconde les numéros de téléphone des clients, et la troisième la liste des appels.

Id_client	Nom	Prénom	Adresse
156	Müller	Didier	Grand-Rue 49 2800 Delémont
10234	Müller	Didier	Petit-Chêne 3 2900 Porrentruy
…	…	…	…

Table des clients

Numéro	Id_client
0324221022	156
0324660010	10234
0324669987	10234
…	…

Table des numéros

La table des numéros permet de faire la liaison entre un abonné et son ou ses numéros de téléphone. Elle est indispensable du fait qu'un abonné peut avoir plusieurs numéros de téléphone différents.

Id_appel	Numéro appelant	No appelé	Date et heure de l'appel	Durée de l'appel (sec)	Tarif (CHF / min)
123465	0324660010	00333246634217	10.2.2010 14:32	455	1.00
145674	0324660010	0324711230	17.2.2010 18:37	62	0.20
162346	0324669987	0324711230	23.2.2010 9:21	145	0.20
124369	0324221022	0214537124	12.2.2010 19:01	302	0.10
	...	...	...	...	...

Table des appels

Cette manière de faire est moins lisible au premier coup d'œil : il faut par exemple consulter deux tables pour connaître le propriétaire d'un numéro de téléphone. Mais cet inconvénient est vite balayé par les avantages :

II. SGBD.

Nous allons parler de systèmes informatiques qui nous aident à gérer des données. Nous avons d'un côté, un serveur de données quelque part sur la Toile, avec des disques magnétiques et leurs pistes qui gardent précieusement des séquences de bits. Supposons que le serveur soit celui d'IMDb qui gère une base de données sur le cinéma. Supposons que l'utilisateur, disons Alice, veuille savoir quels films ont été réalisés par Alfred Hitchcock. Pour ce faire, elle spécifie des mots-clés ou remplit les champs d'un formulaire proposé par IMDb. Sa question voyage depuis son navigateur jusqu'au serveur de données. Là, cette question est transformée en un programme qui s'exécute pour obtenir la réponse. Ce qui est important : **ce programme, Alice n'a pas envie de l'écrire ; elle n'a pas à l'écrire.**

Votre ordinateur personnel et votre téléphone stockent leurs données dans des systèmes de fichiers. Et parfois quand vous ne savez plus où vous avez mis quelque chose, vous faites des *recherches* dans ces système de fichiers. Un moteur de recherche de la Toile ne fait pas autre chose, seulement il le fait sur un système de fichiers à l'échelle de la planète.

Dans ce chapitre, nous parlerons de systèmes qui gèrent aussi des données mais qui sont bien plus sophistiqués que les systèmes de fichiers : les systèmes de gestion de bases de données (**SGBD**).
Ils permettent à des individus ou des programmes d'exprimer des requêtes pour interroger des bases de données ou pour les modifier. Nous nous focaliserons ici sur les plus répandus d'entre ces systèmes, les

systèmes relationnels, parmi lesquels nous trouvons des logiciels commerciaux très répandus comme celui d'Oracle, ***Microsoft Access*** édité par *Microsoft* et des logiciels libres très utilisés comme MySQL.

3. Principe et architecture.

Au fil des ans, trois grands principes ont émergé qui ont façonné le domaine de la gestion de données :

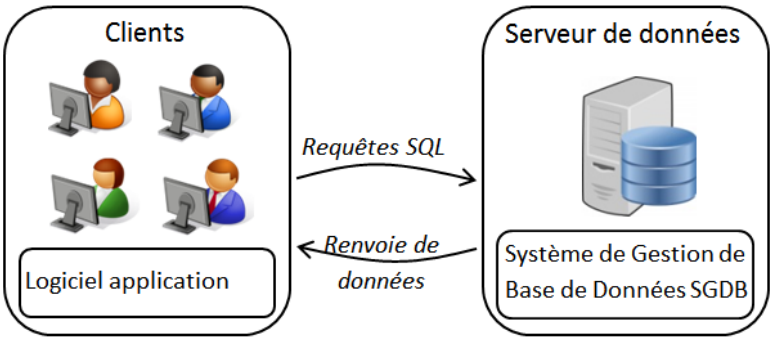
- **Abstraction :**

- **Universalité :**

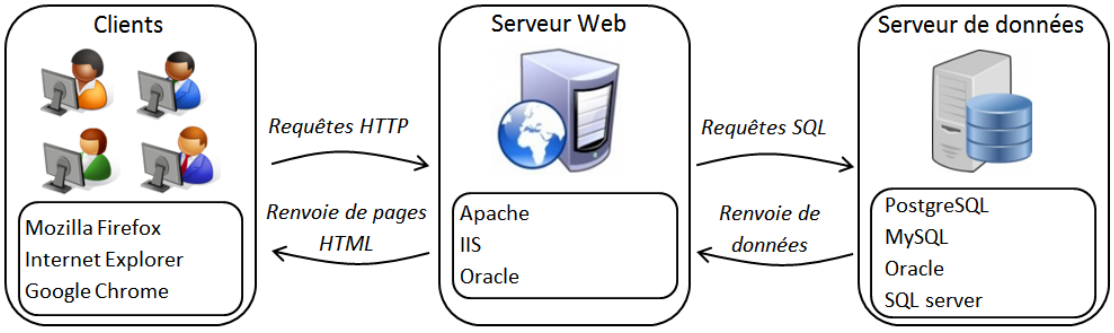
- **Indépendance :**

Architectures les plus répandues de SGBD :

- **client/serveur :**

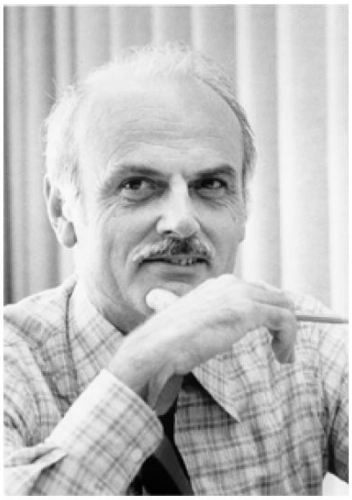


- **Trois Tiers :**



4. Algèbre relationnelle.

Un système de gestion de bases de données doit aussi proposer un langage, pour exprimer des requêtes, facilement utilisable par des êtres humains. Ces exigences forment le point de départ du modèle relationnel proposé par Ted Codd (1923-2003) .



Son article *« A Relational Model of Data for Large Shared Data Banks », CACM 13, No. 6, June 1970* qui fonde le modèle relationnel.

Dans le modèle relationnel, les données sont organisées en tableaux à deux dimensions que nous appellerons des **relations**.

Comme illustration, nous utiliserons une base de données consistant en une relation ***Film*** et une relation ***Séance*** :

Film			Séance		
Titre	**Réalisateur**	**Acteur**	**Titre**	**Salle**	**Heure**
Casablanca	*M. Curtiz*	*Humphrey Bogart*	*Casablanca*	*Lucernaire*	*19 :00*
Casablanca	*M. Curtiz*	*Peter Lore*	*Casablanca*	*Studio*	*20 :00*
Les 400 coups	*F. Truffaut*	*J.-P. Leaud*	*Star Wars*	*Sel*	*20 :30*
Star Wars	*G. Lucas*	*Harrison Ford*	*Star Wars*	*Sel*	*22 :15*

Une base de données relationnelles

Une ligne de ces relations est appelée un ***n-uplet***. Par exemple**,** <*Star Wars, Sel, 22 :15* > est un n-uplet d'**arité** 3, c'est-à dire un triplet, dans la relation Séance.

Les colonnes ont des noms, appelés **attributs**, comme *Titre*. Un n-uplet est noté de la manière suivante : < "*Casablanca*", "*M. Curtiz*", "H*umphreyBogart*">

Les données sont interrogées en utilisant comme langage le calcul relationnel. Le calcul relationnel s'appuie sur des noms qui représentent des relations comme Film ou Séance, des entrées de ces relations comme *Star Wars*, des variables comme *t, h,* et des symboles logiques, ^ (et), ∧_ (ou), ¬ (non), ⇒ (implique), ∃(existe), ∀(pour tout).

À partir de ces ingrédients, des formules logiques peuvent être construites telles que :

$$qHB = \exists\, t, r;\ (\text{Film}(t, r, \text{“}Humphrey\ Bogart\text{”}) \wedge \text{Séance}(t, s, h))$$

qui se lit : il existe un titre *t* et un réalisateur *r* tels que le n-uplet < *t, r, "Humphrey Bogart"* > se trouve dans la relation Film, et le n-uplet <*t, s, h* > dans Séance. Observez que *s* et *h* ne sont pas quantifiées dans la formule précédente ; nous dirons que ces deux variables sont **libres**. La formule *qHB* peut être vue comme une **requête** du calcul relationnel. Elle se lit alors : donnez-moi les salles *s* et les horaires *h*, s'il existe un réalisateur *r* et un titre *t* tels que...

Où et à quelle heure puis-je voir un film avec Humphrey Bogart ? Pour faciliter la lisibilité de ces formules logiques, on peut exprimer une requête en définissant précisément la forme du n-uplet en lui affectant les variables libres que l'on cherche à obtenir (ce qui était implicite dans l'expression précédente) :

$$\{res(s,h) / \exists t,r;(FILM(t,r,"Humphrey\ Bogart") \wedge SEANCE(t,s,h)\}$$

L'intérêt de ce procédé est qu'on peut ensuite réutiliser les n-uplets résultats exactement comme s'il s'agissait d'une relation. C'est ce qu'on appelle le mécanisme des vues. Une **vue** est une relation, qui au lieu d'être stockée dans la base de données, est définie intentionnellement. Un utilisateur de la base de données peut l'utiliser comme n'importe quelle autre relation.

III Vocabulaire.

FILM

Titre	Directeur	Acteur
Mais qui a tué Harry ?	*Hitchcock*	*Gwenn*
Mais qui a tué Harry ?	*Hitchcock*	*Forsythe*
Mais qui a tué Harry ?	*Hitchcock*	*MacLaine*
Mais qui a tué Harry ?	*Hitchcock*	*Hitchcock*
	. . .	
Cris et chuchotements	*Bergman*	*Andersson*
Cris et chuchotements	*Bergman*	*Sylwan*
Cris et chuchotements	*Bergman*	*Thulin*
Cris et chuchotements	*Bergman*	*Ullman*

COORDONNÉES

Salle	Adresse	Téléphone
Gaumont Opéra	*31 bd. des Italiens*	*47 42 60 33*
Saint André des Arts	*30 rue Saint André des Arts*	*43 26 48 18*
Le Champo	*51 rue des Ecoles*	*43 54 51 60*
	. . .	
Georges V	*144 av. des Champs-Elysées*	*45 62 41 46*
Les 7 Parnassiens	*98 bd. du Montparnasse*	*43 20 32 20*

SÉANCE

Salle	Titre	Horaire
Gaumont Opéra	*Cris et chuchotements*	*20 :30*
Saint-André des Arts	*Mais qui a tué Harry ?*	*20 :15*
Georges V	*Cris et chuchotements*	*22 :15*
	. . .	
Les 7 Parnassiens	*Cris et chuchotements*	*20 :45*

La base de données **CINEMA**

On dispose d'un ensemble **att** d'attributs. On associe à un attribut *a* un ensemble de *constantes* noté **dom**(*a*). On note **dom** l'union des ensembles de toutes les constantes de tous les attributs.

Exemple :

Un **schéma relationnel** *R* consiste en un ensemble fini de *n* attributs $U = \{u1, \ldots, un\}$. On le note aussi $R[U] = \{u1, \ldots, un\}$. . On définit une fonction **sort**, qui associe à chaque schéma relationnel son ensemble d'attributs, c'est-à-dire **sort**(*R*) = *U*. L'**arité** d'un schéma relationnel est son nombre d'attributs.

Exemple :

Un ***schéma de base de données*** **R** est un ensemble fini de *p* schémas relationnels que l'on écrit **R** = {*R*1[*U*1], . . . ,*Rp*[*Up*]} pour indiquer les schémas relationnels qui la composent ainsi que leurs sorts.

Exemple :

Un ***n-uplet*** est une fonction sur un ensemble fini *U* d'attributs. Plus précisément, un n-uplet *u* sur *U* est une fonction de *U* dans **dom**.
Une ***relation***, appelée également *instance sur un schéma relationnel R*[*U*] (ou sur un ensemble *U* d'attributs), est un ensemble fini *I* de n-uplets sur *U*.

Exemple :