

SI3: Exploitation des données

CHAP I - Introduction aux bases de données

Principaux concepts des SGBD et principes du modèle relationnel.





Sommaire

1 Généralités	3
1.1 Historique	3
1.2 Définition	4
2 Les modèles de base de données	5
2.1 Le modèle hiérarchique	5
2.2 Le modèle réseau	6
2.3 Le modèle relationnel	6
2.4 Le modèle objet	7
3 Les SGBDR	8
3.1 Composants	8
3.2 Fonctionnalités	9
3.3 Les différents types d'utilisateurs	9
4 Le modèle relationnel	10
4.1 Notions de table et de relation	10
4.2 Attributs et clés	10
4.2.1 Clé candidate et clé primaire	11
4.2.2 Clé étrangère	12
4.3 Notions d'intégrité	12
4.4 Représentations possibles	13
4.5 Application	14
5 Références	14

1 Généralités

1.1 Historique

Dès le début de l'informatique, on a voulu construire des systèmes pour effectuer des calculs (équations différentielles, calcul matriciel, ...).

L'approche classique de mise en place d'une application informatique dans une entreprise, consistait le plus souvent à l'écriture d'un certain nombre de programmes destinés à l'exploitation d'un ensemble de **fichiers** qu'il fallait aussi créer. Les principaux problèmes posés par cette démarche sont : la **redondance** des informations et la **dépendance** entre les données et les programmes qui les manipulent.

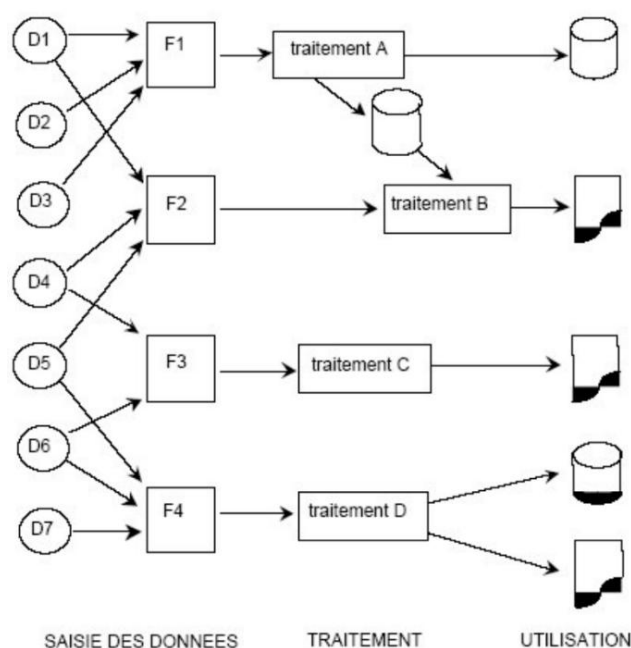


Fig.1 : Données organisées en fichiers (jusqu'aux années 60)

Aujourd'hui, la tendance actuelle est la gestion de grandes voire très grandes ¹ quantités d'informations. Cela revient à **stocker** des données et **manipuler** ces données. Notons que les données peuvent être de natures diverses, les opérations plus ou moins compliquées et le nombre d'utilisateurs plus ou moins important.

Exemples d'applications :

Applications de gestion (paye, stock, ...), applications transactionnelles (banque, réservation...), applications de documentation (bibliothèque, cartographie, ...), Ingénierie (PAO, CAO, ...).

¹ Les entrepôts de données (*data warehouse*) contiennent plusieurs téraoctets (1 To = 1024 Go) de données.

Aujourd'hui on parle du *Big data*. Les volumes des données sont tellement importants que les solutions classiques de gestion de bases de données sont désormais obsolètes.

1.2 Définition

Une **base de données**² est un ensemble d'informations stocké sur un système informatique. Cet ensemble est implanté physiquement (généralement sur disque dur) sous la forme d'un ou plusieurs fichiers.
Cette organisation est assurée par un logiciel spécialisé : Le **SGBD** (Système de Gestion de Base de Données)

Les fonctionnalités élémentaires³ d'un SGBD sont :

- **Structurer/organiser** les données
- **Stocker** les données
- **Mettre à jour** les données (ajout, modification et suppression d'informations)
- **Interroger** les données

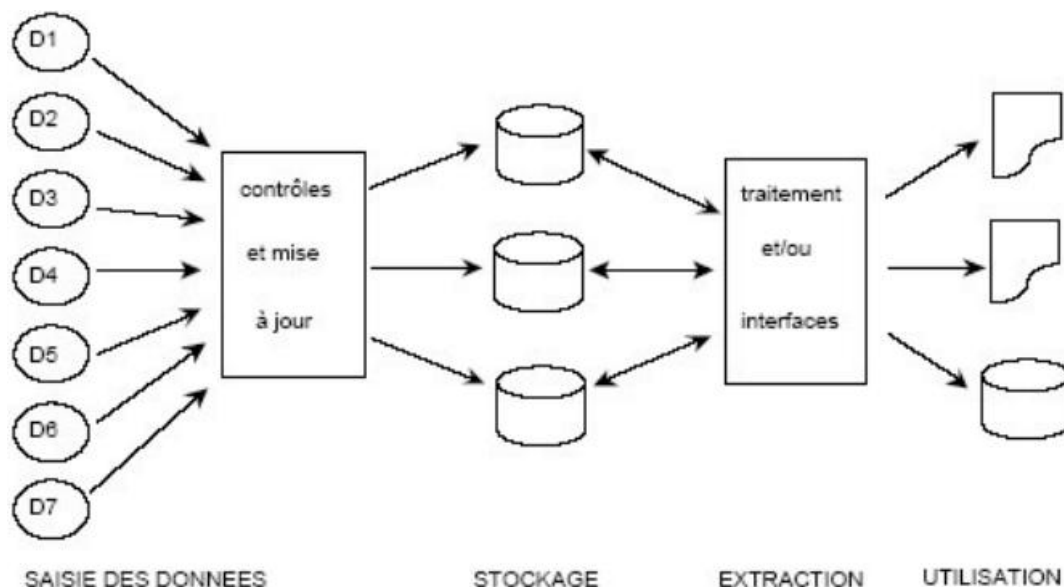


Fig. 2 : Principe de fonctionnement des SGBD

La construction d'une base de données passe tout d'abord par la réalisation d'un « plan » : un schéma conceptuel. Cette phase de conception est généralement guidée par une méthode (ex. MERISE) L'objectif est de modéliser⁴ le domaine étudié.

Exemple :

Dans une entreprise on souhaite réaliser une **base de données « fournisseurs »**. La modélisation consistera à recenser les informations nécessaires concernant les fournisseurs (**raison sociale, téléphone, e-mail...**) et les produits (**référence, désignation, prix...**) ainsi que les liens entre ces deux entités (Qui fournit quoi ?)

L'organisation sémantique des informations peut être réalisée suivant plusieurs modèles de données.

Les principaux modèles de base de données sont les suivants :

- Le modèle hiérarchique
- Le modèle réseau
- Le modèle relationnel
- Le modèle objet

Notons cependant que le **modèle relationnel** est aujourd'hui utilisé par la grande majorité des SGBD (environ $\frac{3}{4}$) et que l'on retrouve également d'autres modèles (ex. : navigationnel et déductif)

² Abréviations : « BD », « BdD »

³ 4 opérations de base sur les données : CRUD (*Create, Read, Update, Delete*)

⁴ La modélisation ne sera pas abordée dans ce module.



2 Les modèles de base de données

2.1 Le modèle hiérarchique

Conçu à la NASA pour la gestion des données du programme spatial Apollo (années 60), les données sont classées hiérarchiquement, selon une arborescence descendante. Ce modèle utilise des **pointeurs** entre les différents enregistrements : à chaque enregistrement correspond un enregistrement parent. Bien adapté à des données de type nomenclatures avec une relation 1 vers N mais inapproprié aux structures de données complexes.

Les fichiers XML⁵ constituent une réminiscence des bases de données hiérarchiques.

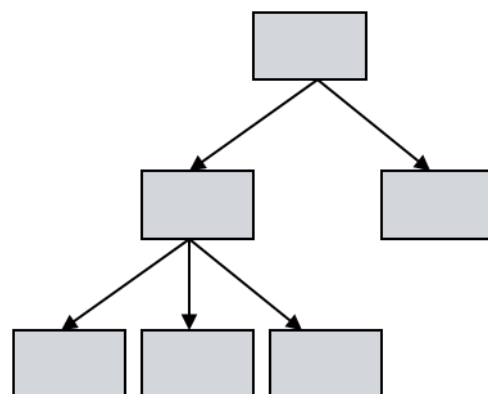


Fig. 3 : Modèle hiérarchique

```

<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE CD SYSTEM "cd.dtd">
<CD>
  <Titre>Stan Getz Finest Hour</Titre>
  <Interprete>STAN GETZ</Interprete>
  <Genre>Jazz/blues</Genre>
  <Piste>
    <Titre>It Never Entered My Mind</Titre>
    <Compositeur>Hart</Compositeur>
    <Compositeur>Rodgers</Compositeur>
    <Duree>3:51</Duree>
  </Piste>
  <Piste>
    <Titre>S-h-i-n-e</Titre>
    <Compositeur>Brown</Compositeur>
    <Duree>8 :56</Duree>
  </Piste>
</CD>
  
```

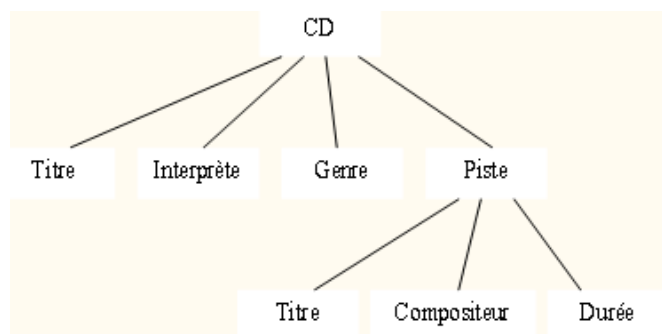
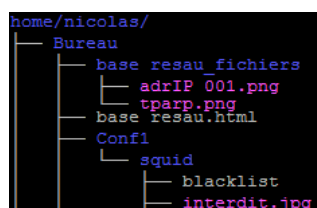
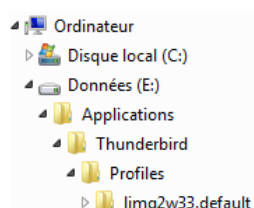


Fig. 4 : Exemple d'arborescence et de contenu d'un fichier XML

Exemples :

- La **classification du vivant** relève du modèle hiérarchique de base de données : Le chat appartient à la famille des félidés, du sous-ordre félifomes de l'ordre des carnivores, de la classe des mammifères, du sous-embranchement des vertébrés du règne animal.
- Vu de l'utilisateur, un **système de fichiers** s'apparente à une base de données hiérarchique (arborescence des dossiers)



⁵ XML : Extensible Markup Language (langage de balisage extensible). Ce format de données est une référence pour l'échange de données informatisées.

2.2 Le modèle réseau

Ce modèle constitue une extension du modèle hiérarchique, il utilise des pointeurs vers des enregistrements selon une structure arborescente.

Il est cependant possible d'établir des liens sans restriction entre les différents éléments.

Imaginé par Charles Bachman, sa spécification a été publiée en 1969 par le consortium Codasyl, à l'origine du langage Cobol.

Plus que le modèle hiérarchique, le modèle réseau implique une connaissance de la structure de la base de données pour permettre l'accès aux données : les logiciels sont dépendants de la structure de la base.

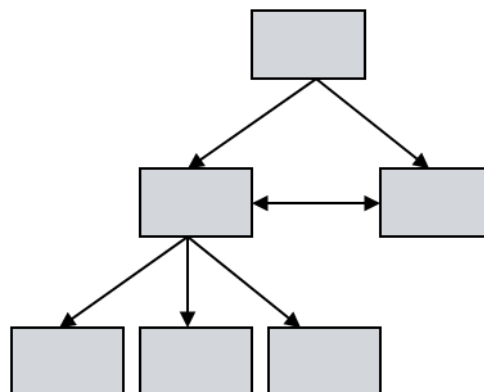


Fig. 5 : Modèle réseau

Exemple :

- Des données **généalogiques** peuvent être organisées selon le modèle réseau. Un enregistrement de type parent dispose d'un pointeur vers chacun des membres de sa descendance. Chaque membre de la descendance dispose d'un pointeur vers son aîné et d'un autre vers son cadet.

2.3 Le modèle relationnel

Ce modèle est fondé sur la théorie mathématique des relations.

Le schéma conceptuel peut être vu comme un ensemble de **tables (ou relations)** à n colonnes, n désignant le degré de la relation.

Avec le modèle relationnel, une table sert à représenter aussi bien une **classe d'objets**⁶ qu'une **association entre des classes d'objets**. Chaque élément d'une table est appelé un n-uplet.

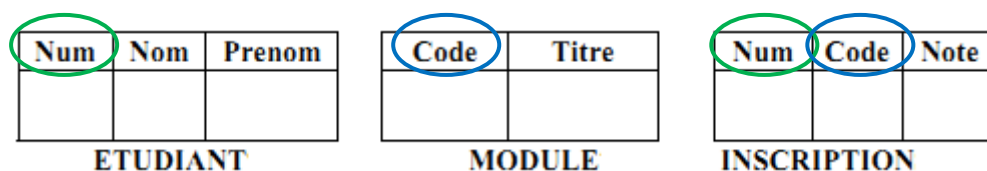


Fig. 6 : Modèle relationnel

Dans l'exemple de la Fig. 6, la table INSCRIPTION décrit l'association entre la classe d'objets ETUDIANT et la classe MODULE.

Elle permet de modéliser le fait qu'un étudiant peut s'inscrire à aucun, un, ou plusieurs modules.

Ce modèle est le plus fréquent, il fera l'objet d'un paragraphe un peu plus loin.

Les logiciels qui s'appuient sur ce modèle sont les SGBDR (R pour relationnel)

Le langage dédié aux opérations sur les données est le SQL (Structured Query Language)

⁶ Le terme objet représente ici un élément, un acteur du système d'information (ex. : une facture, un produit, un client...)

2.4 Le modèle objet

La notion de bases de données objet ou relationnel-objet est plus récente.

Les données sont représentées sous forme d'objets. Comme en programmation orientée objet⁷, les objets contiennent les **données qui les décrivent** ainsi que la **logique** qui permet de les utiliser ou de les modifier.

Chaque enregistrement de la base de données constitue une instance de la classe d'objets correspondante. Ces instances sont classées de manière hiérarchique dans la base de données.

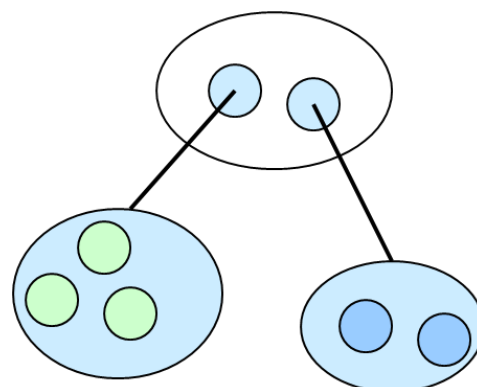


Fig. 7 : Modèle objet

Les **SGBDO** (Systèmes de Gestion de Bases de Données orientés Objet) sont recommandés pour les applications nécessitant des performances élevées dans la manipulation de données complexes. Certains SGBDO fonctionnent efficacement avec les langages orientés objet tels que C++, C#, Java ou encore Python et Visual Basic .Net.

Il existe un langage spécifique pour l'interrogation des bases de données orientées objet : OQL (Object Query Language), dérivé de SQL.

Notes :

- Les SGBDO sont globalement cantonnés à des niches telles que l'ingénierie, les études spatiales, ou encore la recherche fondamentale en physique et en biologie moléculaire.
- Le serveur d'applications Web Zope utilise une base de données objet, de même que les annuaires (LDAP⁸ ou X.500⁹, par exemple).

⁷ La POO (Programmation Orienté Objet) sera abordée dans le module SI4.

⁸ LDAP : Lightweight Directory Access Protocol. Protocole permettant l'interrogation et la modification des services d'annuaire.

⁹ X.500 : Ensemble des normes informatiques sur les services d'annuaire.



3 Les SGBDR



Figure 8 : Logos des principaux SGBDR du marché

Les SGBD relationnels sont à l'heure actuelle les plus diffusés sur le marché¹⁰.

Ils permettent d'organiser les données sous formes de tables. La description de la base de données est faite grâce à un schéma conceptuel ou relationnel permettant de décrire toutes les tables (relations) implantées sur disque.

Un SGBDR sert à effectuer des opérations ordinaires telles que consulter, modifier, construire, organiser, transformer, copier, sauvegarder ou restaurer des bases de données.

3.1 Composants

Un SGBD est un ensemble de logiciels parmi lesquels on trouve un moteur de base de données, un interprète du langage SQL, une interface de programmation, et diverses interfaces utilisateur.

- Le **moteur** de base de données.
C'est le composant central du SGBD qui effectue la majorité des traitements de manipulation du contenu des bases de données.
- Interprète **SQL**.
SQL est un langage informatique qui sert à exprimer des requêtes d'opérations sur les bases de données. L'interprète SQL décode les requêtes et les transforme en un plan d'exécution détaillé qui est alors transmis au moteur de base de données.
- **Interface de programmation**.
C'est une bibliothèque logicielle qui permet à un programme tiers de communiquer avec le SGBD, de demander des opérations et de récupérer des données provenant des bases de données. Le détail des demandes est souvent formulé en langage SQL.

ODBC¹¹ est un logiciel médiateur (*middleware*) qui permet à des logiciels, par l'intermédiaire d'une interface de programmation unique de communiquer avec différents SGBD ayant chacun une interface de programmation différente. C'est un logiciel souvent utilisé avec les SGBD.

- **Interface utilisateur**.
C'est l'interface graphique (homme-machine) qui permet de mettre en œuvre toutes les fonctionnalités proposées par le SGBD.
On retrouve parfois une interface dédiée à l'interrogation des données appelée QBE (Query By Example) : Le principe est que l'utilisateur présente un exemple du résultat de recherche attendu (sous forme d'une matrice), puis le soumet au SGBD.

¹⁰ En 2009, IBM DB2, Oracle Database, MySQL, PostgreSQL et Microsoft SQL Server sont les principaux systèmes de gestion de base de données sur le marché (Wikipédia)

¹¹ ODBC : Open Database Connectivity



3.2 Fonctionnalités

Les fonctionnalités d'un SGBDR sont :

- **Administration** des données.
Le SGBD doit proposer de outils pour créer des bases de données, les déplacer, les copier, les répliquer¹², effectuer des sauvegardes et des restaurations de données.
- **Manipulation** des données par des langages non procéduraux (cf. fonctionnalités élémentaires : CRUD).
Des utilisateurs non informaticiens doivent pouvoir manipuler simplement les données, c'est-à-dire les interroger et les mettre à jour sans préciser d'algorithmes d'accès.
Les SGBD professionnels sont néanmoins plutôt utilisés par des informaticiens !
- Efficacité des **accès** aux données.
Nécessité de garantir un bon débit et un bon temps de réponse entre les utilisateurs.
- **Cohérence** des données.
Certaines informations doivent être vérifiées (Ex. : Le prix d'un produit doit être un entier positif) Le SGBD doit veiller à ce que les applications respectent cette règle lors des modifications de données. Une telle règle est appelée contrainte d'intégrité.
- **Partage** des données.
Diverses applications doivent pouvoir partager les données de la base dans le temps et simultanément, comme si elles étaient seules à les utiliser. Le SGBD doit gérer les accès concurrents.
- **Sécurité** des données.
Les données doivent être protégées contre les accès non autorisés ou mal intentionnés. Leur sécurité doit aussi être assurée en cas de panne d'un programme ou du système, voire de la machine.

3.3 Les différents types d'utilisateurs

On distingue plusieurs rôles que peuvent jouer un individu ou un groupe d'individus pour concevoir, créer, mettre en œuvre et exploiter une base de données.

- Le **développeur** d'applications (ou analyste programmeur)
Après modélisation du système d'information étudié, c'est lui qui propose le modèle relationnel de la future base de données.
Il est chargé ensuite d'élaborer les programmes pour exploiter la base de données.
- **L'administrateur** de la base de données.
C'est lui qui (à partir du modèle relationnel) est chargé de l'aspect plus technique de la création de la base. Il assure les fonctionnalités d'administration et de sécurité des données.
- **L'utilisateur.**
Il s'agit de caractériser ici la personne qui se sert simplement de la base de données et qu'on appelle couramment l'utilisateur final (*End User* en anglais).
Ces personnes ne sont pas des informaticiens, elles utilisent les ressources logicielles mises à disposition par le développeur et l'administrateur.
L'utilisateur « averti » est capable d'interroger la base en utilisant le langage SQL.

¹² La réplication de données correspond à une duplication des bases sur plusieurs serveurs pour faire face à une forte charge ou assurer une continuité de service en cas de panne d'une machine. Le SGBD est chargé d'assurer la synchronisation des différentes bases.



4 Le modèle relationnel

4.1 Notions de table et de relation

Une base de données est constituée d'un ensemble de tables reliées entre elles.

Une **table** est un tableau dans lequel les lignes sont appelées les **enregistrements** (ou tuples) et l'intitulé des colonnes les **champs**.

A l'intersection d'une ligne et d'une colonne figure une **valeur**.

Exemples de table :

	Dossard	Nom	Prenom	DateNais	Enseigne
+	1	Sanne	Hubert	10/10/1960	Le Sympa
+	2	Mandin	Thierry	12/05/1963	Bar des Sports
+	3	Blanchet	Patrick	08/03/1958	Bar des Sports
+	4	Hurand	Philippe	04/02/1965	Bar des Sports
+	5	Daviaud	Michel	03/04/1950	Chez Rolland
+	6	Arnaud	Christian	25/01/1956	Le Terminus

Fig. 9 : Une table MS-Access

refproduit	designproduit	commentaireproduit	prix TTC produit	codegamme
CH001	Lit 160 x 200 cm	En palissandre des Indes	917	2
CH002	Commode L110 H76 P55 cm	Montée à la mode asiatique en palissandre des Inde...	509	2
CH003	Armoire L100 H170 P85 cm	Montée à la mode asiatique en palissandre des Inde...	700	2
CH004	Méridienne en teck	Assise et dossier en rotin tressé (livrée sans les	139	3
CH005	Lit 140x 190 cm	Lit en teck livré avec le sommier	229	3

Figure 10 : Une table MySQL

Au niveau relationnel, ces tables correspondent aux **relations** ci-dessous :

GARCON(Dossard, Nom, Prenom, DateNais, Enseigne)

Produit(refproduit, designproduit, commentaireproduit, prix TTC produit, codegamme)

Ce mode de représentation est appelé « en intention », il représente la structure de la base de données (son schéma, son plan...)

Au niveau relationnel, on parlera d'**occurrences** pour désigner les enregistrements et d'**attributs** pour parler des champs.

4.2 Attributs et clés

Un attribut est une information, une donnée élémentaire, une rubrique désignant le plus petit élément d'information manipulable. Il est caractérisé par un **nom** et un **type**.

Exemples :

NomClient : attribut de type alphabétique (ex. de valeur : "DUPOND", "PAYET", ...)

QteCmdee : attribut de type entier (ex. de valeurs : 5, 10, 2, ...)



Soit la relation Lecteur(NomLecteur, PrenomLecteur) et le contenu de la table correspondante :


NomLecteur	PrenomLecteur
ALAMARD	Frédéric
BERTRAND	Carole
SPAGUETTI	Soisic
DUPUIS	Laurence
BERTRAND	Pierre
ELAS	Ludivine
FOUX	Eric

Fig. 11 : Table lecteur

Que pensez-vous de l'exemple ci-dessus ?

La présence d'un **homonyme** (le lecteur BERTRAND) ne permettra pas d'identifier de manière certaine un lecteur.

Il faut définir une clé pour cette relation.

 La clé est un attribut qui permet de distinguer chaque occurrence d'une relation par rapport à toutes les autres. Toutes les valeurs de cet attribut doivent être **uniques**.
Une relation doit posséder au moins un attribut et si c'est le cas, ce doit être la clé.

4.2.1 Clé candidate et clé primaire

Les clés candidates sont des attributs susceptibles de pouvoir jouer le rôle de clé.

Dans le cas de la « **Fig. 11** : Table lecteur », aucun des champs ne peut jouer le rôle de clé. On rajoute alors un champ supplémentaire qui permettra de distinguer chaque lecteur :

Lecteur(NumLecteur, NomLecteur, PrenomLecteur)

Le champ NumLecteur devient alors la **clé primaire** de la relation. Par convention, ce doit être le premier attribut de la relation et il doit être souligné.

Dans la plupart des SGBDR, un type numérique spécial est dédié à ce type clé. Appelé NuméroAuto, AutoIncrement... il est géré automatiquement.


Nom du champ	Type de données
 NumLecteur	NuméroAuto
NomLecteur	Texte
PrenomLecteur	Texte

Figure 12 : Caractéristiques de la table Lecteur

Dans l'exemple ci-dessous, précisez pour chaque champ s'il peut ou non être clé candidate. Justifiez.

ELEVE					
CodeElève	NomElève	PrénomElève	DateNaissance	CodeINSEE	CodeDivision
18	WILDA	Rachid	14-07-1989	1 89 07 75 021 342	7
12	MEZZ	Janine	20-02-1988	2 88 02 77 103 045	4
47	CHICKEN	Aude	12-04-1988	2 88 04 75 021 342	5
115	GROCK	Alain	20-11-1988	1 88 11 77 103 045	2
14	SEM	Alain	13-05-1989	1 89 05 93 018 112	
39	SÉTALA	Maud	23-07-1990	2 90 07 95 145 112	8

Figure 13 : Table Eleve



4.2.2 Clé étrangère

Ce type d'attribut permet de matérialiser les liens entre les différentes tables.

Une clé étrangère correspond à la clé primaire d'une autre table.

Dans l'exemple de la « Figure 10 : Une table MySQL » le dernier champ appelé CodeGamme correspond en fait à une valeur existante dans une autre table : Gamme.

Table « Gamme »

codegamme	libellegamme
1	Rotin
2	Asie
3	Teck

Table « Produit »

refproduit	designproduit	commentaireproduit	prix TTC produit	codegamme
CH001	Lit 160 x 200 cm	En palissandre des Indes	917	2
CH002	Commode L110 H76 P55 cm	Montée à la mode asiatique en palissandre des Inde...	509	2
CH003	Armoire L100 H170 P85 cm	Montée à la mode asiatique en palissandre des Inde...	700	2
CH004	Méridienne en teck	Assise et dossier en rotin tressé (livrée sans les	139	3
CH005	Lit 140x 190 cm	Lit en teck livré avec le sommier	229	3

Fig. 14 : Relations entre tables

Le champ **CodeGamme** assure la **liaison** entre les deux tables. Ainsi, dans la table Produit, on note que le produit désigné par « Lit 140x 190 cm » fait partie de la gamme n°3.

Dans la table Gamme on trouvera que la gamme n°3 correspond aux produits fabriqués en teck.

Le schéma relationnel de cette base peut s'écrire comme ci-dessous :

Gamme(codegamme, libellegamme)

Produit(refproduit, designproduit, commentaireproduit, prix TTC produit, **#codegamme**)

Par convention, une **clé étrangère** est précédée d'un # et est placée en dernier dans la liste des attributs d'une relation.

4.3 Notions d'intégrité

Dans le paragraphe « 3.2 Fonctionnalités » nous avons vu que l'un des principaux objectifs d'un SGBDR était d'assurer la cohérence des données appelée également **intégrité des données**. Cette cohérence est en partie assurée par la mise en place de **contraintes d'intégrité**.

a) Intégrité de « niveau table »

Dans la plupart des cas, chaque table dispose d'une **clé primaire**.

Les **contraintes de domaines** sont liées aux colonnes des tables (attribut non nul, entier positif, valeurs comprises dans un intervalle...)

b) Intégrité référentielle

Il s'agit ici pour le SGBDR de vérifier la **cohérence clé étrangère/clé primaire**.

La définition d'une telle contrainte forcera le SGBDR à faire les contrôles suivants :

- Dans un champ clé étrangère il est impossible de renseigner une valeur qui n'existe pas dans la clé primaire (Le code gamme n° 4 ne peut être affecté à un produit si la gamme n'existe pas)
- Impossible de supprimer un enregistrement de la table contenant la clé primaire s'il existe des enregistrements liés (Dans la table gamme, l'enregistrement correspondant au code gamme n°2 ne pourra être supprimé car il existe, dans la table produit, des enregistrements liés.
- Impossible de modifier une valeur de clé primaire dans la table primaire si cet enregistrement a des enregistrements liés.



4.4 Représentations possibles

La représentation du modèle relationnel est indépendante de la base de données utilisée et peut se présenter sous plusieurs formes.

Tout schéma relationnel doit cependant préciser :

- Les tables avec leurs noms
- Les champs des tables
- Les clés primaires
- Les clés étrangères
- Les liaisons clés primaires/clés étrangères

Pour illustrer les différents modes de représentation nous partirons d'un exemple classique, des commandes réalisées par des clients :

- Un client est identifié par un numéro et on note son nom, son adresse, etc.
- Pour une commande, identifiée par un numéro, on précisera sa date ainsi que le client qui l'a passée.

a) Représentation en intention :

Client(NumClient, NomClient, AdresseClient)
Commande(NumCmde, DateCmde, #NumClient)

b) Représentation détaillée :

Client(NumClient, NomClient, AdresseClient)
NumClient : Clé primaire

Commande(NumCmde, DateCmde, NumClient)
NumCmde : Clé primaire
NumClient : Clé étrangère en référence à NumClient de la table Client

L'avantage de cette écriture est qu'elle mettra en évidence les relations entre des champs qui ne portent pas forcément le même nom.

Ainsi nous aurions pu avoir :

Commande(NumCmde, DateCmde, NumClientCmde)
NumCmde : Clé primaire
NumClientCmde : Clé étrangère en référence à NumClient de la table Client

c) Représentation graphique :

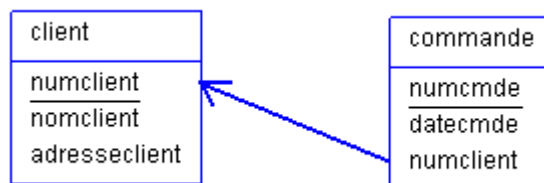


Fig. 15 : Représentation graphique d'un schéma relationnel

Remarque : La plupart des SGBDR permettent d'afficher sous forme graphique le modèle relationnel de leurs bases de données. Cette représentation permet de mieux appréhender le système d'information.



4.5 Application

L'exemple de la Fig. 6 (repris ci-dessous) modélise le fait qu'un étudiant peut s'inscrire à 0, 1 ou plusieurs modules.

Num	Nom	Prenom

ETUDIANT

Code	Titre

MODULE

Num	Code	Note

INSCRIPTION

- a) Donner des exemples de tuples pour chaque table.
- b) Donner les différentes représentations possibles de ce modèle relationnel.

5 Références

Frédéric DI GALLO, « Cours de base de données », cycle probatoire CNAM BORDEAUX 1999-2000

Dr. Brahim BELATTAR, « Bases de données à l'usage de l'étudiant », LISA - Dpt d'informatique - Faculté des sciences de l'Ingénieur - Univ. de Batna, 2002

Philippe LACURIE, « Les bases de données », ALSI BTS IG 1^{ère} année, 2005

Alain VAN SANTE et Michel AUGUSTE, « Variations pédagogiques autour de l'enseignement du modèle relationnel et des SGBDR en 1^{ère} STG », Réseau CERTA 2005

Pierre VETTER, « Introduction aux bases de données », B1 SUPINFO, 2011

Wikipédia, « Base de données et Système de gestion de base de données », 2011