Sources : Pierre Crescenzo, Didier Müller, Noël Novelli, Dominique Laporte, wikipedia

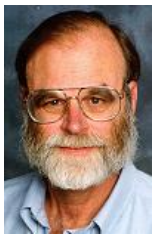
1. Introduction

Une base de données (BD) est un ensemble structuré d'informations. Dans le langage courant, le terme peut désigner toute source importante de données telle qu'une encyclopédie. Dans le domaine de l'informatique, c'est un ensemble d'informations regroupées sous la forme d'enregistrements stockés sur un système de fichiers structurés et organisées de manière à pouvoir être facilement manipulées.

L'organisation logique des données se fait selon un modèle de données et la structure physique des fichiers comporte des index destinés à accélérer les opérations de recherche et de tri. Le modèle de données relationnel est aujourd'hui le plus utilisé parce qu'il permet l'indépendance entre la structure physique et l'organisation logique des données.

Le logiciel qui manipule les bases de données est appelé système de gestion de base de données (SGBD). Il permet d'organiser, de contrôler, de consulter et de modifier la base de données. Les opérations sont parfois formulées dans un langage de requête tel que SQL, qui est le plus connu et employé pour les modèles relationnels.

1.1. Propriétés ACID



Les propriétés ACID sont un ensemble de propriétés qui garantissent qu'une transaction informatique est exécutée de façon fiable. Jim Gray a défini les propriétés qui garantissent des transactions fiables à la fin des années 1970 et a développé des technologies pour les mettre en œuvre automatiquement.

1. Atomicité : La propriété d'atomicité assure qu'une transaction se fait au complet ou pas du tout : si une partie d'une transaction ne peut être faite, il faut effacer toute trace de la transaction et remettre les données dans l'état où elles étaient avant la transaction. L'atomicité doit être respectée dans toutes situations, comme une panne d'électricité, une défaillance de l'ordinateur, ou une panne d'un disque magnétique.
2. Cohérence : La propriété de cohérence assure que chaque transaction amènera le système d'un état valide à un autre état valide. Tout changement à la base de données doit être valide selon toutes les règles définies, incluant mais non limitées aux contraintes d'intégrité, aux rollbacks[1] en cascade, aux déclencheurs de base de données, et à toutes combinaisons d'événements.
3. Isolation : Toute transaction doit s'exécuter comme si elle était la seule sur le système. Aucune dépendance possible entre les transactions. La propriété d'isolation assure que l'exécution simultanée de transactions produit le même état que celui qui serait obtenu par l'exécution en série des transactions. Chaque transaction doit s'exécuter en isolation totale : si T1 et T2 s'exécutent simultanément, alors chacune doit demeurer indépendante de l'autre.
4. Durabilité : La propriété de durabilité assure que lorsqu'une transaction a été confirmée, elle demeure enregistrée même à la suite d'une panne d'électricité, d'une panne de l'ordinateur ou d'un autre problème. Par exemple, dans une base de données relationnelle, lorsqu'un groupe d'énoncés SQL ont été exécutés, les résultats doivent être enregistrés de façon permanente, même dans le cas d'une panne immédiatement après l'exécution des énoncés.

1 méthode permettant d'annuler l'ensemble des requêtes que l'on vient de réaliser (le fait inverse du commit)

1.2. Un peu d'Histoire

Quelques dates phares dans le domaine des BD :

1961 Apparition du premier système que l'on nommera bien plus tard un SGBD : IDS (Integrated Data Storage) réalisé par la société General Electric. Il pose les bases du modèle réseau.

1965-1970 Les données sont traitées au moyen de systèmes de fichier. IBM développe le modèle hiérarchique avec IMS (Information Management System). IMS évolue vers IMS DB/DC (DataBase/DataCom) qui utilise le modèle réseau.

1970 À partir du début des années 1970, les universitaires s'intéressent aux BD et les font dès lors progresser plus sensiblement. Le modèle relationnel apparaît.

1972-1975 Premières conférences internationales sur le thème des BD. Début des travaux qui mèneront à la méthode Merise.

1976 Définition de la méthode Merise et publication du modèle Entité-Association.

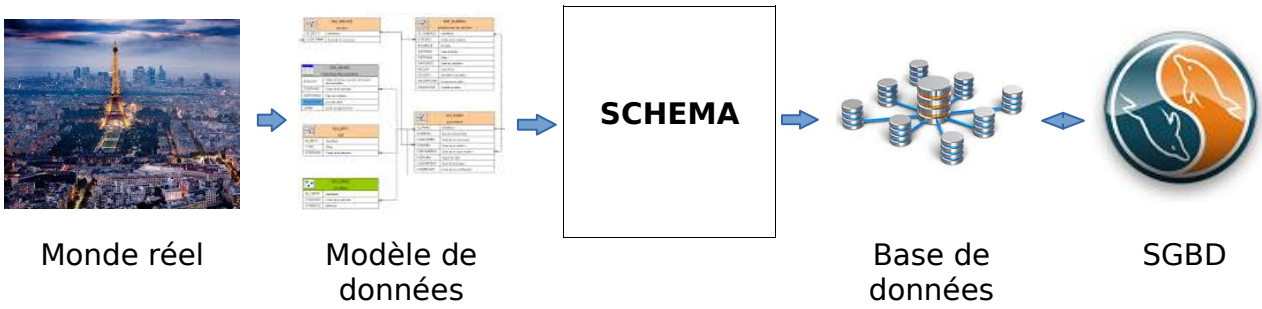
1975-1980 Les premiers SGBDR sont diffusés, par exemple : SYSTEM-R d'IBM et INGRES de l'Université de Californie. Ils restent cependant, à ce stade, assez expérimentaux.

1980 Les SGBDR prennent leur essor et conquièrent peu à peu le marché des BD, remplaçant les SGBD hiérarchiques et réseaux. Les outils logiciels deviennent de plus en plus performants et ergonomiques.

1990 Dans les années 1990, les SGBDOO émergent peu à peu. Plus puissants que leurs équivalents relationnels, ils sont aussi plus complexes et ne se sont pas encore imposés à l'heure actuelle. Ils restent l'évolution future la plus probable en matière de BD.

2. Objectif et démarche

Les bases de données sont basées sur une approche de « structuration » du monde réel qui donne une représentation abstraite (le schéma) résultant de l'application d'un modèle de données (data model).



Replongeons-nous avant les années 1980, lorsque la téléphonie mobile n'existait pas. 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.

2.1. 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)
Camé	Léon	Petit-Chêne 3 2900 Porrentruy	0324660010	00333246634217	10.2.2010 14:32	455	0.70
Camé	Léon	Petit-Chêne 3 2900 Porrentruy	0324660010	0324711230	17.2.2010 18:37	62	0.20
Camé	Léon	Petit-Chêne 3 2900 Porrentruy	0324669987	0324711230	23.2.2010 9:21	145	0.20
Camé	Léon	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 :

1. Les informations sont redondantes : l'adresse du client apparaît plusieurs fois, encombrant inutilement la mémoire. Il aurait été plus judicieux d'avoir une autre table où l'on aurait enregistré les coordonnées des clients une fois pour toutes.
2. Si un utilisateur change d'adresse, il faut reporter ce changement sur chaque ligne où il apparaît, sous peine d'avoir une table incohérente.
3. Pour identifier un client, on a besoin de trois champs. Il aurait été plus simple d'associer à chaque client un numéro de client unique (ce que toutes les entreprises font).
4. Un tableur n'est tout simplement pas fait pour gérer une grande base de données.

2.2. 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	Camé	Léon	Grand-Rue 49 2800 Delémont
10234	Camé	Léon	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.

Si cela n'avait pas été le cas, on aurait simplement pu ajouter un champ « numéro » à la table des clients. Ce champ aurait alors pu être utilisé comme identifiant et remplacer le champ « Id_Client ».

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 :

1. Les informations ne sont plus redondantes : par exemple, l'adresse d'un client apparaît une seule fois et est donc facilement modifiable.
2. Un client est identifié par un numéro, ce qui évite des confusions.
3. Il est très facile de tirer la liste des clients, puisqu'ils sont tous rassemblés dans une table.

3. Les bases de données relationnelles

C'est l'article d'Edgar Frank Codd (1923-2003) « A Relational Model of Data for Large Shared Data Banks », CACM 13, No. 6, June 1970 qui fonde le modèle relationnel, mais une première description de ce modèle avait déjà été publiée l'année précédente dans un rapport technique d'IBM : « Derivability, Redundancy, and Consistency of Relations Stored in Large Data Banks », IBM Research Report RJ599.

3.1. Tables

Dans les bases de données relationnelles, une table est un ensemble de données organisées sous forme d'un tableau où les colonnes correspondent à des champs et les lignes à des enregistrements, également appelés entrées.

La notion de table est apparue dans les années 1970 chez IBM avec l'algèbre relationnelle qui est une théorie mathématique en relation avec la théorie des ensembles. Cette théorie a pour but de clarifier et de faciliter l'utilisation d'une base de données.

3.2. Conception d'une table

Lors de la conception d'une base de données relationnelle, il est important de clairement définir toutes les tables qui la composeront et les différentes relations qui les lient, de manière à pouvoir dresser le schéma conceptuel qui permettra de décrire le fonctionnement de la base de données avant de la mettre « physiquement » en place.

On distinguera les tables courantes, qui contiendront divers champs contenant des informations, et les tables de liaison, qui feront les liens entre deux tables courantes.

3.3. Contenu d'une table

Par nature, chaque colonne d'une table, également nommé « champ », doit contenir des données d'un même type et ce champ doit être nommé. Chaque table est l'implémentation physique d'une relation entre les différents champs. Chaque correspondance est définie par une ligne de la table. Il y a certaines règles à respecter, notamment le fait qu'il faille mettre un identifiant pour chaque enregistrement dans la table. Il y a deux possibilités :

- Mettre un identifiant qui s'auto-incrémente au fur et à mesure des données entrées (par exemple « Id_appel » dans la « table des appels »).
- Choisir un champ pouvant servir d'identifiant unique (par exemple « Numéro » dans la « table des numéros »).

3.4. Travail sur une table

Il y a deux niveaux de travail sur une table :

- un niveau de définition des données d'une table, qui permet de définir, lier, et contraindre les données via un langage de définition de données.
- un niveau de manipulation des données d'une table, qui permet d'ajouter, supprimer, rechercher des données via un langage de manipulation de données

Actuellement, le langage standardisé pour travailler sur les tables est le SQL. Il est utilisé avec quelques variantes sur la plupart des systèmes de gestion de bases de données.

4. Modèle entité-association

Le modèle entité-association (aussi appelé « modèle entité-relation ») est un type de schéma conceptuel très utilisé pour les bases de données, notamment les bases de données relationnelles. Il a été inventé par Peter Pin-Shan Chen en 1975 et est destiné à clarifier l'organisation des données dans les bases de données relationnelles en notifiant :

4.1. Les entités

- Ce sont des objets concrets que l'on peut identifier (client, livre, individu, voiture, etc.).
- On peut représenter un ensemble d'entités de la réalité par une entité type (un client pour l'ensemble des clients).
- Ces entités sont caractérisées par leurs attributs (pour un client : nom, prénom, adresse, ...).

Parmi ces attributs, on définit un identifiant qui va permettre de caractériser de façon unique un enregistrement dans l'entité (un numéro de client par exemple).

4.2. Les relations entre les entités

- Elles représentent les liens existant entre une ou plusieurs entités.
- Elles sont caractérisées par un nom, une propriété d'association et éventuellement des attributs.

4.3. Degré de relation et cardinalités

- Le degré de la relation (ou dimension de la relation) est le nombre d'entités qui sont impliquées dans cette relation.
- La cardinalité (d'une entité par rapport à une relation) exprime le nombre de participations possibles d'une entité à une relation. Comme c'est un nombre variable, on note la cardinalité minimum (0 ou 1) et maximum (1 ou n) pour chaque entité.

4.4. Exemple

Dans notre exemple du début, un abonné peut avoir de 1 à n numéros de téléphone, mais le numéro de téléphone n'appartient qu'à un client ou aucun client (numéro pas encore attribué ou plus attribué). Il ne peut pas avoir 0 numéro, car dans ce cas il ne serait plus client de l'opérateur.

1. Il y a 3 entités : les clients, les numéros de téléphone et les appels.

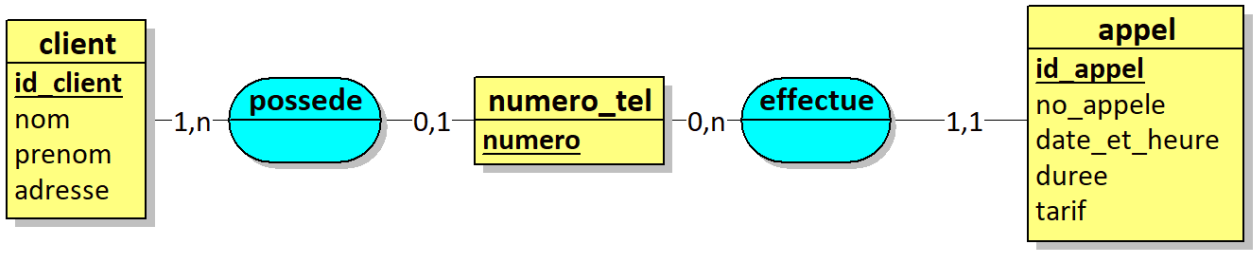
2. Les attributs :

- Pour les clients : id, nom, prénom, adresse.
- Pour les numéros de téléphone : le numéro.

- Pour les appels : le numéro appelé, la date, la durée et le tarif.

3. Les relations et les cardinalités :

- Un client possède 1 ou plusieurs numéros.
- Un numéro appartient à (est possédé par) 0 ou 1 client.
- Un numéro effectue 0 ou plusieurs appels.
- Un appel est fait depuis (est effectué par) 1 et un seul numéro.



On peut classer les relations en 3 types selon leur cardinalité maximale :

- les relations un à un (par exemple 1:1 – 0:1)
- les relations un à plusieurs (c'est le cas des deux relations ci-dessus)
- les relations plusieurs à plusieurs (par exemple 0:n – 1:n)

Exercice 1

On veut créer une petite base de données permettant de garder le contact avec nos copains de classe. On supposera qu'ils sont tous domiciliés en France, qu'ils n'ont qu'un numéro de téléphone, mais éventuellement plusieurs adresses. On veut stocker les renseignements suivants : nom, prénom, date de naissance, numéro de téléphone, numéro et adresse de la rue, code postal, ville.

Réalisez un schéma entité-association en suivant la démarche décrite ci-dessus.

Exercice 2

On veut créer une base de données permettant de gérer les clients étrangers d'une entreprise et les pays de ces clients. La table des clients sera simplifiée et comportera leur nom, leur prénom, le pays de résidence et le solde de leur compte, dans la monnaie du pays.

On veut aussi que les clients aient la possibilité de faire des réclamations et que la base de données gère toutes ces réclamations pour chacun des clients.

Réalisez un schéma entité-association.

Exercice 3

Une école veut informatiser ses listes de classe. Une classe est formée d'élèves (toujours les mêmes et un élève n'appartient qu'à une seule classe). Il y a un professeur responsable de classe par classe. Un professeur peut enseigner plusieurs disciplines.

Réalisez un schéma entité-association.

5. Traduction du schéma conceptuel en un modèle relationnel

Les règles principales de transformation d'un schéma conceptuel entité-association en un schéma relationnel sont :

Règle I

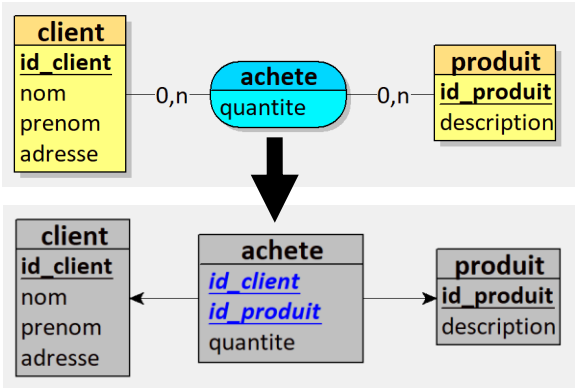
Toute entité est traduite en une table relationnelle dont les caractéristiques sont les suivantes :

- le nom de la table est le nom de l'entité
- la clé de la table est l'identifiant de l'entité
- les autres attributs de la table forment les autres colonnes de la table

Règle II

Toute relation binaire plusieurs à plusieurs est traduite en une table relationnelle dont les caractéristiques sont les suivantes :

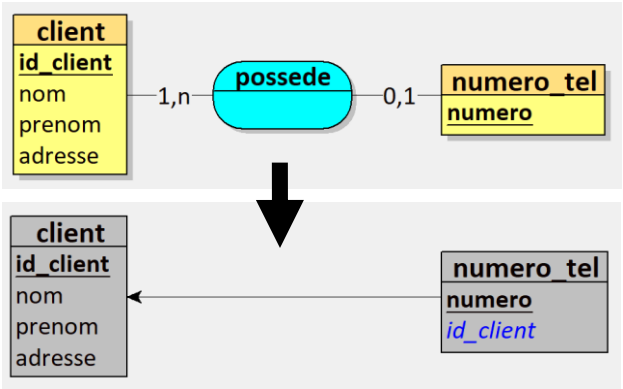
- le nom de la table est le nom de la relation
- la clé de la table est formée par « l'addition » des identifiants des entités participant à la relation
- les attributs spécifiques de la relation forment les autres colonnes de la table



Règle III

Toute relation binaire un à plusieurs est traduite :

1. soit par un report de clé (voir schéma ci-contre) : l'identifiant de l'entité participant à la relation côté N est ajoutée comme colonne supplémentaire à la table représentant l'autre entité. Cette colonne est parfois appelée clé étrangère. Le cas échéant, les attributs spécifiques à la relation sont eux aussi ajoutés à la même table ;

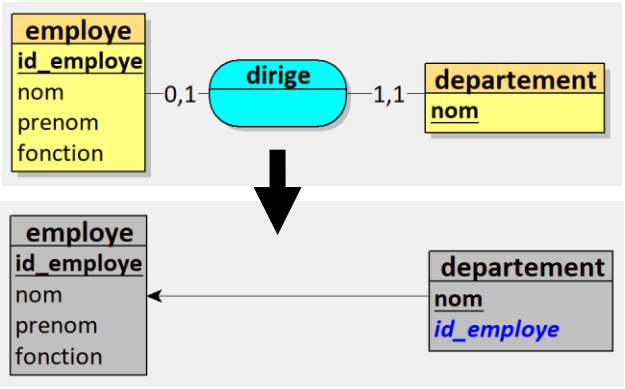


2. soit par une table spécifique dont les caractéristiques sont les suivantes :
   - le nom de la table est le nom de la relation
   - la clé de la table est l'identifiant de l'entité participant à la relation côté 1
   - les attributs spécifiques de la relation forment les autres colonnes de la table

Règle IV

Toute relation binaire un à un est traduite, au choix, par l'une des trois solutions suivantes :

- choix 1 : fusion des tables des entités qu'elle relie
- choix 2 : report de clé d'une table dans l'autre (voir ci-dessous)
- choix 3 : création d'une table spécifique reliant les clés des deux entités



Exercice 1 : Transformez le schéma entité-association de l'exercice 1 du §4.4 en un modèle relationnel.

Exercice 2 : Transformez le schéma entité-association de l'exercice 2 du §4.4 en un modèle relationnel.

Exercice 3 : Transformez le schéma entité-association de l'exercice 3 du §4.4 en un modèle relationnel.

7. La création d'une base de données

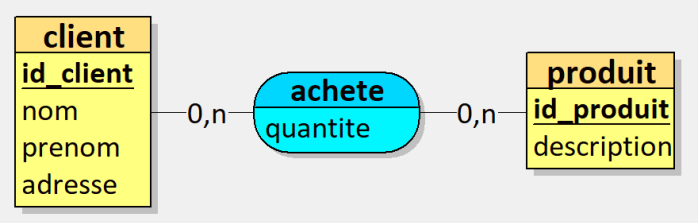
La construction d'une base de données commence par une phase d'analyse. Cela se fait au moyen de méthodes d'analyse comme Merise[2] ou UML[3]. Cette analyse conceptuelle permet de se représenter le fonctionnement du système d'informations pour lequel on cherche à construire une base de données.

L'ANSI SPARC[4] propose différentes approches de schéma de données qui ont été reprises par ces méthodes :

- « schéma conceptuel »,
- « schéma logique »
- « schéma physique »

7.1. Modèle Conceptuel des Données (MCD)

Le Modèle Conceptuel des Données (ou Modèle entité-association), permet de représenter la structure du système d'information, du point de vue des données, et définit également les dépendances ou relations entre ces différentes données.



Les concepts de base du modèle conceptuel de données sont : l'entité, l'association, la propriété (ou attribut) et les cardinalités.

7.2. Modèle logique des données (MLD)

Le modèle logique des données consiste à décrire la structure de données utilisée sans faire référence à un langage de programmation. Il s'agit donc de préciser le type de données utilisées lors des traitements.

Ainsi, le modèle logique est indépendant du type de base de données utilisé.

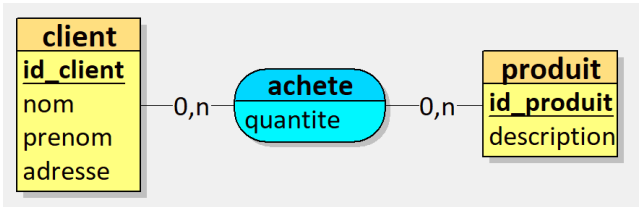
MLD

client = (**id_client** *VARCHAR(50)*, nom *VARCHAR(50)*, prenom *VARCHAR(50)*, adresse *VARCHAR(50)*);
produit = (**id_produit** *INT*, description *VARCHAR(50)*);
achete = (***#id_client, #id_produit***, quantite *INT*);

NB : les clés sont soulignées et les clés étrangères précédées par #.

7.3. Modèle Physique des Données (MPD)

Le Modèle Physique des Données peut être composé soit de tableaux décrivant le schéma, soit du code SQL de création du schéma : le code SQL est une déduction directe du MPD en fonction du SGBD choisi et de sa version.



SQL

```
CREATE TABLE client(
  id_client VARCHAR(50),
  nom VARCHAR(50) NOT NULL,
  prenom VARCHAR(50) NOT NULL,
  adresse VARCHAR(50),
  PRIMARY KEY(id_client)
);

CREATE TABLE produit(
  id_produit INT,
  description VARCHAR(50) NOT NULL,
  PRIMARY KEY(id_produit)
);

CREATE TABLE achete(
  id_client VARCHAR(50),
  id_produit INT,
  quantite INT NOT NULL,
  PRIMARY KEY(id_client, id_produit),
  FOREIGN KEY(id_client) REFERENCES client(id_client),
  FOREIGN KEY(id_produit) REFERENCES produit(id_produit)
);
```

client_achete_produit.loo | client_achete_produit.loo

2 http://fr.wikipedia.org/wiki/Merise_%28informatique%29
3 http://fr.wikipedia.org/wiki/UML_%28informatique%29
4 http://fr.wikipedia.org/wiki/Architecture_Ansi/Sparc