# La base de données - SQL

Maintenant que nous avons vu comment peut transiter une requête d’un poste client vers le serveur et inversement, il est temps de se pencher sur la source d’information qui va être interrogée, c’est-à-dire -le plus souvent- ce que l‘on appelle une base de données.

Une base de données peut être définie comme un ensemble de données modélisant les objets d’une partie du monde réel et servant de support à une application informatique. Cet ensemble de données doit être interrogeable par le contenu et selon n’importe quel critère lié à sa structure.

Un SGBD (système de gestion de bases de données)[[1]](#footnote-1) peut être considéré comme un ensemble de logiciels systèmes permettant le stockage, la sauvegarde, l’interrogation, la recherche, la mise en forme et la sécurisation des données. Dans le cas d’un procédure d’interfaçage Web/DB on parlera généralement de SGBD relationnels. Voyons cela de plus près …

Hébergement partagé/serveurs dédiés

**Systèmes de gestion**  
  
- Oracle

- DB2  
- Sybase  
- PostgreSQL  
- MS SQLServer  
- MySQL

- Firebird

- FileMaker

- Interbase  
- SQLite  
- MS Access  
...



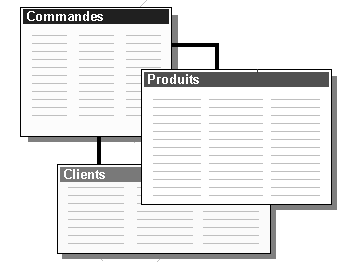
## Bases de données relationnelles

Le modèle relationnel a été proposé par E.F. Codd en 1970. Il est considéré par beaucoup comme le plus simple et le plus pratique des modèles de représentation de données. Il repose sur une vision tabulaire des informations et sur la théorie mathématique des ensembles.

Parmi les objectifs à l'origine de ce modèle citons-en deux de premier ordre :

- permettre un haut degré d'indépendance entre les applications (programmes, interfaces) et la représentation interne des données (fichiers, chemins d'accès);  
  
- établir une base solide pour traiter les problèmes de cohérence et de redondance des données.

Dans ce modèle, les données sont structurées sous forme de tables composées de lignes et de colonnes. Il est possible de mettre en relation différentes tables et d’effectuer des opérations (algèbre) relationnelles sur les tables.

Dans le cadre du développement d’une interface Web/DB, c’est ce modèle qui est pour l’instant le plus souvent rencontré.

### Table, ligne, attribut, clé primaire, clé étrangère

Une table est un ensemble de lignes de données distinctes. Une table est constituée d'une clé primaire et de plusieurs attributs (ou colonnes) qui dépendent de cette clé.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| CLIENTS | | | | |
| **NCLI** | NOM | ADRESSE | LOCALITE | COMPTE |
| Une ligne regroupe des informations concernant un objet, un individu, un événement, un concept du monde réel. Une colonne correspond à un attribut qui caractérise cet objet, individu, événement… | | | | |

La clé primaire d'une table est un attribut ou un groupe d'attributs de la table qui détermine tous les autres de façon unique. Une table ne devrait posséder qu’une seule clé primaire. Par contre, une table peut présenter plusieurs clés candidates qui pourraient jouer ce rôle.

Clé primaire unique :

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| CLIENTS | | | | |
| **NCLI** | NOM | ADRESSE | LOCALITE | COMPTE |

Clé primaire composée :

|  |  |  |
| --- | --- | --- |
| CATALOGUES | | |
| **ANNEE** | **SAISON** | EDITEUR |

Le domaine d'un attribut est l'ensemble des valeurs que peut prendre cet attribut. Le domaine est constitué d'un type, d'une longueur et de contraintes qui réduisent l'ensemble des valeurs permises.

Exemple:

EDITEUR char 255 not null

NCLI serial not null

DATE date

Une **clé étrangère** dans une table est une clé primaire ou candidate dans une autre table. Elle est à la base des liaisons entre tables.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| CLIENTS | | | | |
| **NCLI** | NOM | ADRESSE | LOCALITE | COMPTE |

|  |  |  |
| --- | --- | --- |
| COMMANDES | | |
| **NCOM** | NCLI | DATE |

|  |  |  |  |
| --- | --- | --- | --- |
| PRODUITS | | | |
| **NPRO** | LIBELLE | PRIX | QSTOCK |

|  |  |  |
| --- | --- | --- |
| LIGNESCOM | | |
| **NCOM** | **NPRO** | QCOM |

Légende :

* Table CLIENTS : chaque ligne décrit un client; les colonnes décrivent successivement le numéro du client (NCLI), son nom (NOM), son adresse (ADRESSE), sa localité (LOCALITE) et l’état de son compte. (NCLI) est la clé primaire.
* Table PRODUITS : chaque ligne décrit un produit; les colonnes décrivent successivement le numéro du produit (NPRO), son libellé (LIBELLE), son prix unitaire (PRIX) et la quantité restant en stock (QSTOCK). (NPRO) est la clé primaire.
* Table COMMANDES : chaque ligne décrit une commande passée par un client; les colonnes décrivent successivement le numéro de la commande (NCOM), le numéro du client qui a passé la commande (NCLI) et la date de la commande (DATE). (NCOM) est la clé primaire. (NCLI) est une clé étrangère vers la table CLIENTS.
* Table LIGNESCOM : chaque ligne décrit une ligne de commande ; les colonnes décrivent successivement le numéro de la commande à laquelle la ligne appartient (NCOM), le numéro du produit commandé (NPRO) et la quantité commandée (QCOM). (NCOM) et (NPRO) composent la clé primaire et sont également deux clés étrangères respectivement vers la table COMMANDES et PRODUITS.

Exemple de ce que pourrait contenir à un moment donné la structure de base de données présentée ci-dessus :

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| CLIENTS | | | | |
| **NCLI** | NOM | ADRESSE | LOCALITE | COMPTE |
| 4623 | RAFINE | 30, r. F. Celine | Paris | 0 |
| 4624 | MEULA | 156, r. des Roses | Liège | 1000 |
| 4788 | MARIN | 423, av. Albert | Luxembourg | 23000 |

|  |  |  |
| --- | --- | --- |
| COMMANDES | | |
| **NCOM** | NCLI | DATE |
| 24 | 4624 | 20/09/1999 |
| 67 | 4788 | 12/10/1999 |
| 82 | 4623 | 20/10/1999 |

|  |  |  |  |
| --- | --- | --- | --- |
| PRODUITS | | | |
| **NPRO** | LIBELLE | PRIX | QSTOCK |
| 1 | pelle | 650 | 78 |
| 2 | pioche | 600 | 12 |

|  |  |  |
| --- | --- | --- |
| LIGNESCOM | | |
| **NCOM** | **NPRO** | QCOM |
| 24 | 1 | 3 |
| 24 | 2 | 2 |
| 67 | 2 | 1 |
| 82 | 1 | 14 |

### Règles d’intégrité

Les bases de données relationnelles sont régies par ce qu’on appelle les règles d’intégrité. Parmi celles-ci, on distingue les contraintes de domaine qui restreignent l'ensemble des valeurs que peut prendre un attribut dans une table, les contraintes d'intégrité d'entité qui précisent qu'une table doit toujours avoir une clé primaire et les contraintes d'intégrité référentielles qui précisent les conditions dans lesquelles peuvent être ajoutés ou supprimés des enregistrements lorsqu'il existe des associations entre tables par l'intermédiaire de clés étrangères.

## Langage SQL

SQL (*Structured Query Language)* est un langage normalisé qui constitue le standard d’accès aux bases de données relationnelles. Les autres interfaces par menus, fenêtres, grilles, etc., ou de programmation type langage de 3e et 4e génération, sont le plus souvent offerts au-dessus du langage SQL. Celui-ci constitue donc le point d’entrée obligatoire des SGBD relationnels.

Dans le cadre des interfaces Web/DB, le dialogue entre le langage de programmation sur le serveur (PHP, Servlet Java, ASP …) et la base de données va s’établir autour du langage SQL. **En fait, le rôle des scripts sur le serveur consiste principalement à faire passer des « ordres » SQL à la base de données et à en récupérer le résultat**.

Sur le serveur, le dialogue avec la base de données se décompose en deux étapes:

1. le script (PHP, Java …) formule et envoie la requête SQL
2. le SGBD répond à la requête SQL du script

Voyons à présent comment ces instructions SQL peuvent être formulées.

Ce chapitre décrit quatre des principales opérations qu’offre le SQL[[2]](#footnote-2) :

* La recherche (SELECT) qui permet de retrouver des lignes ou parties de lignes;
* L’insertion (INSERT) qui permet d’ajouter des lignes dans une table;
* La suppression (DELETE) qui permet de supprimer des lignes dans une table;
* La modification (UPDATE) qui permet de mettre à jour des lignes dans une table.

### SELECT – Consultation de données

Dans une base de données relationnelles les données sont stockées dans des tables :

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Clients | | | | |
| **ncli** | Nom | Adresse | Localite | Compte |
| 4623 | RAFINE | 30, r. F. Celine | Paris | 0 |
| 4624 | MEULA | 156, r. des Roses | Liège | 1000 |
| 4788 | MARIN | 423, av. Albert | Luxembourg | 23000 |

La clause SELECT du langage SQL est utilisée pour consulter une base de données.

Syntaxe :

**SELECT** nom\_colonne(s)

**FROM** nom\_de\_la\_table

La clause SELECT est suivie des attributs qui doivent être récupérés.

La clause FROM est suivie des tables qui contiennent les attributs concernés.

Exemple :

**SELECT** Nom, Adresse, Localite   
**FROM** Clients

Cette requête a pour résultat :

|  |  |  |
| --- | --- | --- |
| RAFINE | 30, r. F. Celine | Paris |
| MEULA | 156, r. des Roses | Liège |
| MARIN | 423, av. Albert | Luxembourg |

La consultation d’une base de données peut se faire sur base de critères de sélection précis grâce à la clause WHERE.

Syntaxe :

**SELECT** nom\_colonne(s)

**FROM** nom\_de\_la\_table

**WHERE** nom\_colonne **opérateur** valeur

La clause WHERE est suivie des conditions qui guident la sélection.

Dans le langage SQL, il existe six opérateurs relationnels:

|  |  |
| --- | --- |
| = | égalité |
| != | différence |
| < | plus petit que |
| > | plus grand que |
| <= | plus petit ou égal |
| => | plus grand ou égal |

Exemple:

A partir de la table Client, on désire obtenir le numéro de client de ceux qui ont un compte supérieur à 500:

**SELECT** ncli   
**FROM** Clients   
**WHERE** compte **>** 500

Cette requête a pour résultat :

|  |
| --- |
| 4624 |
| 4788 |

Plusieurs conditions peuvent être combinées grâce aux opérateurs boléens ‘AND’ et ‘OR’.

**SELECT** \* **FROM** EMPLOYE **WHERE** POSITION **=** 'Manager'

**SELECT** \* **FROM** EMPLOYE

**WHERE**

NOM='Clemens'

**AND** **(**PRENOM='Louis' **OR** PRENOM='Robert'**)**

**SELECT** NOM, PRENOM, DATE   
**FROM** EMPLOYE   
**WHERE** **(**POSITION = 'Manager' **AND** SALAIRE > 60000**)**

**OR** BENEFICE > 12000

### INSERT – Insertion de données

La clause INSERT du langage SQL est utilisée pour insérer des informations dans une base de données.

Syntaxe :

**INSERT INTO** nom\_de\_la\_table

**VALUES** (valeur1, valeur2, valeur3,...)

ou

**INSERT INTO** nom\_de\_la\_table (colonne1, colonne2, colonne3...)

**VALUES** (valeur1, valeur2, valeur3...)

La clause INSERT est suivie du nom de la table concernée et, entre parenthèses, des colonnes qui vont recevoir les données.

La clause VALUES est suivie des valeurs à insérer dans les colonnes précisées plus haut via la clause INSERT.

Remarque : si le nom des colonnes n’est pas spécifié, les données sont insérées en fonction de l’ordre des colonnes établi lors de leur création.

Exemple:

**INSERT INTO** Clients (ncli, Nom, Adresse, Localite, Compte)

**VALUES** (4789, 'Carton', '45, av Etienne', 'Charleroi', 20000)

Cette requête a pour résultat l’insertion d’une nouvelle ligne dans la table Clients:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Clients | | | | |
| **ncli** | Nom | Adresse | Localite | Compte |
| 4623 | RAFINE | 30, r. F. Celine | Paris | 0 |
| 4624 | MEULA | 156, r. des Roses | Liège | 1000 |
| 4788 | MARIN | 423, av. Albert | Luxembourg | 23000 |
| 4789 | Carton | 45, av Etienne | Charleroi | 20000 |

### DELETE – Suppression de données

La clause DELETE du langage SQL est utilisée pour supprimer des informations dans une base de données.

Syntaxe :

**DELETE FROM** nom\_de\_la\_table

**WHERE** colonne=valeur

La clause DELETE FROM est suivie du nom de la table concernée .

La clause WHERE est suivie des conditions qui guident la suppression.

Exemple:

**DELETE FROM** Clients

**WHERE** Nom = 'MEULA'

Cette requête a pour résultat la suppression d’une ligne dans la table Clients:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Client | | | | |
| **ncli** | Nom | Adresse | Localite | Compte |
| 4623 | RAFINE | 30, r. F. Celine | Paris | 0 |
| 4788 | MARIN | 423, av. Albert | Luxembourg | 23000 |
| 4789 | Carton | 45, av Etienne | Charleroi | 20000 |

Utilisée, comme dans l’exemple ci-dessous, sans clause WHERE la commande DELETE aura pour effet de vider la table précisée en argument.

DELETE FROM Clients

Cette commande efface le contenu de la table Clients.

La structure de la table est conservée, mais les données sont perdues.

### UPDATE – Modification de données

La clause UPDATE du langage SQL est utilisée pour modifier des informations dans une base de données.

Syntaxe :

**UPDATE** nom\_de\_la\_table

**SET** colonne1=valeur1, colonne2=valeur2,...

**WHERE** colonne=valeur

La clause UPDATE est suivie du nom de la table concernée.

La clause SET est suivie des colonnes à mettre à jour et leur assigne une valeur.

La clause WHERE est suivie des conditions qui guident la modification.

Exemple:

**UPDATE** Clients

**SET** Compte = 50000

**WHERE** Nom = 'Carton';

Cette requête a pour résultat la mise à jour d’un des champs d’une ligne dans la tableClient :

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Clients | | | | |
| **ncli** | Nom | Adresse | Localite | Compte |
| 4623 | RAFINE | 30, r. F. Celine | Paris | 0 |
| 4788 | MARIN | 423, av. Albert | Luxembourg | 23000 |
| 4789 | Carton | 45, av Etienne | Charleroi | 50000 |

**SQL, prérequis indispensable !**

Dans le cadre des interfaces Web/DB, le rôle des scripts sur le serveur consiste principalement à faire passer des « ordres » SQL à la base de données et à en récupérer le résultat. La connaisance de ce langage constitue donc un prérequis indispensable.

1. SGBD courants: Oracle, MS SQLServer, MySQL, Postgresql, Sybase, DB2, MS Access, FileMaker, Firebird, SQLite. [↑](#footnote-ref-1)
2. Ce chapitre n’a pas pour ambition de passer en revue toutes les aspects du langage SQL – il en donne juste l’aperçu nécessaire pour la bonne compréhension du cours. [↑](#footnote-ref-2)