Une BDD, comment ça marche ?

Table des matières

1. La base de données

Les bases de données sont des outils absolument essentiels dans l'informatique, que ça soit dans le monde des logiciels, des applications mobiles ou des sites Web. Elles consistent en une organisation **structurée** d'**entités / table**, contenant des **champs**, permettant de stocker un nombre infini d'**occurrences / enregistrements**.

Imaginons une base de données d'un garage contenant des véhicules. Le garage aura besoin de garder une trace de ses véhicules en stockant :

- Leur marque
- Leur modèle
- Leur date d'entrée au garage

On peut donc modéliser cette base de données en proposant une **entité VOITURE**, qui contiendra les **champs (field)** :

- marque (string)
- modèle (string)
- date_entree (date)

On peut donc stocker toutes les informations dont on a besoin dans cette **table** en créant une nouvelle ligne dans la base, qu'on appelle une **occurrence**

marque	modele	date_entree
CHEVROLET	CRUZE	2025-07-25

Il manque cependant quelque chose ici, car pour l'instant ces données suffisent à identifier le véhicule, mais si jamais quelqu'un vient déposer une CHEVROLET CRUZE au même garage le même jour…

marque	modele	date_entree
CHEVROLET	CRUZE	2025-07-25
CHEVROLET	CRUZE	2025-07-25

On se retrouve avec deux lignes que plus rien ne différencie. Comment déterminer quelle voiture rendre à quel propriétaire ? Une problématique cruciale avec une réponse extrêmement simple : un **identifiant unique**.

La plupart des SGBD (Système de Gestion de Base de Données) sont configurés pour assigner cet identifiant unique automatiquement, on parle alors de **clé primaire** (PRIMARY_KEY). Sur MySQL par exemple, on peut définir qu'un champ est la clé primaire, et lui attribuer un paramètre appelé « AUTO_INCREMENT ». Ce paramètre nous permettra de laisser MySQL gérer la numérotation des identifiants, il s'assurera lui-même que chaque identifiant est unique. Si le dernier identifiant rajouté dans la **table** est 2, alors le prochain prendra automatiquement la valeur 3.

id (clé primaire)	marque	modele	date_entree
1	CHEVROLET	CRUZE	2025-07-25
2	CHEVROLET	CRUZE	2025-07-25

Maintenant, ça y est, les deux véhicules sont bien différentiables par leur ID. Comment les rattacher à leur propriétaire ?

Pour cela on va créer une autre entité **PROPRIETAIRE**, contenant les champs :

- id (clé primaire)
- nom (string)
- prénom (string)
- telephone (string)

id (clé primaire)	nom	prenom	telephone
1	LEGROS	Jacques	0612345678
2	LEPETIT	Joe	0698765432

Et maintenant qu'on a d'un côté les véhicules, et de l'autre les propriétaires, on va procéder à la **liaison** entre les deux. Pour cela on va rajouter une colonne dans la table **VEHICULE** et y stocker l'id du propriétaire à qui elle appartient.

id (clé primaire)	marque	modele	date_entree	id_proprietaire
1	CHEVROLET	CRUZE	2025-07-25	2
2	CHEVROLET	CRUZE	2025-07-25	1

On peut voir ici que la CHEVROLET avec l'id N°1 appartient au propriétaire N°2, et que la CHEVROLET N°2 appartient au propriétaire N°1.

Le champ id_proprietaire dans la table VEHICULE est une clé étrangère (FOREIGN KEY), référençant le champ id de la table PROPRIETAIRE

Une clé étrangère fait forcément référence à la **clé primaire** d'une autre table. On peut en mettre autant qu'on veut.

Par exemple on peut imaginer une table PRODUIT comme ceci :

- id (clé primaire)
- nom (string)
- prix (double)

id (clé primaire)	nom	prix
1	Kit nettoyage	19.90
2	Housse de siège	49.90

Dans ce cas, on aura une table FACTURE qui fera le lien entre les produits et les propriétaires, comme ceci :

- id (clé primaire)
- id_produit (int)

- id_proprietaire (int)
- quantite (int)

id (clé primaire)	id_produit	id_proprietaire	quantite
1	1	2	1
2	2	1	1

Cette structure de table nous permet de faire correspondre à chaque facture : Le produit acheté, par qui, et en quelle quantité.

2. Les formes normales

Première forme normale

Chaque champ de la BDD doit contenir une valeur atomique

On entend par « valeur atomique » l'idée que le contenu du champ ne peut pas être découpé en plusieurs valeurs. Un bon exemple de ça peut-être l'adresse :

On préférera avoir un champ pour chaque paramètre de l'adresse plutôt que toute l'adresse.

Exemple :

adresse : « 112 rue Dedieu, 69100 Villeurbanne »

Deviendra

adresse : 112 rue Dedieu

codePostal : 69100

ville : Villeurbanne

Intérêt ?

Cela nous garantit que nous pourrons effectuer des requêtes de **tri**, de **filtre** etc… plus précises sans avoir à **traiter** la donnée avant de l'utiliser. Par exemple si je veux extraire de ma base toutes les personnes habitant dans le Rhône, je peux regarder toutes celles dont le code postal commence par 69, alors que si la donnée était écrite dans une chaîne de caractère de type « adresse », je ne pourrais pas l'isoler.

Deuxième forme normale

Chaque champ dans la table doit dépendre de sa clé primaire

On entend par « dépendance » le fait qu'une valeur puisse être retrouvée à partir de sa clé primaire. Il faut que les informations contenues dans la table soient bien cohérentes entre elles et ne dépendent pas d'autre chose que de la clé primaire.

Exemple

Pour un produit, il va avoir un poids, un prix, et… une marque ? Mais les données de la marque comme son année de création, son capital, son statut juridique ne doivent pas dépendre du produit. On va donc stocker la marque dans la table produit avec une clé étrangère qui référencera la table marque

Table produit

id	nom	prix	id_marque
1	Barbie	10.90	1

Table marque

id	raison_sociale	statut_juridique	capital
1	Mattel	SARL	900 000€

Intérêt ?

Cela nous garantit la **cohérence** des données en plus d'améliorer les performances de la base

Troisième forme normale

Les champs dans la table ne doivent dépendre d'aucun champ qui ne sert pas à l'identification

On entend par « champ qui sert à l'identification » le ou LES champs qui forment la clé primaire.

Exemple

Dans une table **commande** on va avoir :

id_commande,

id_user,

id_produit,

categorieProduit,

quantite

Ici la catégorie du produit dépend du produit, mais pas de l'identifiant, elle n'a donc pas sa place ici

Intérêt ?

Cette forme normale élimine les anomalies potentielles en assurant une cohérence des données. Elle ajoute aussi de la précision

3. Les types de données

En base, il est obligatoire de **typer** les champs de notre **table**. On trouve parmi les types les plus fréquents :

INT

DOUBLE(taille, nbDecimales)

VARCHAR(nombre de caractères maximale)

DATE

BOOL/BOOLEAN/TINYINT

- INT

Permet de gérer les nombres entiers. Il sera dans presque 100% des cas le type à donner à notre champ **clé primaire**.

- DOUBLE

Permet de gérer les nombres décimaux. On lui précise le nombre de chiffres maximal à afficher en premier, puis en deuxième, le nombre de décimales à garder. Très utilisé pour les prix.

- VARCHAR

Permet de gérer les chaînes de caractères courtes. On lui précise la taille maximale, souvent à 255 caractères, et il s'adapte selon la longueur réelle de la chaîne contenue afin de ne pas gâcher d'espace de stockage. Pour les textes plus longs, on préfèrera le type **TEXT**

- DATE

Permet de gérer les dates au format YYYY-MM-DD. Pratique car avec ce format on peut les trier par ordre alphabétique et ça va correspondre à l'ordre chronologique également. Mais il faudra les convertir avant de les afficher au format DD/MM/YYYY

- DATETIME

Permet de gérer les dates au format YYYY-MM-DD H:m:s, donc en enregistrant les heures, les minutes et les secondes.

- BOOL/BOOLEAN/TINYINT

Permet de gérer les valeurs booléennes, à savoir **vrai** ou **faux**. Toujours n'admettre que deux valeurs possibles, 0 ou 1, et ne pas oublier de préciser une valeur par défaut. Un booléen non défini est un problème.

Exercices :

1 Créer la table « produit » permettant de stocker les informations suivantes pour un produit :

Il aura un prix, un nom, un poids, et un stock.

PRODUIT

id (int)	nom (varchar)	prix (double)	poids (double)	stock (int)
1	Kit nettoyage	10.90	2.25	100
2	Brosse pare-brise	5.90	1.00	78

2 Créer la table « marque » permettant de stocker les informations suivantes : nom, logo (mettre un chemin d'image, on va considérer que c'est un champ texte), année de création, statut juridique, capital

MARQUE

id (int)	nom (varchar)	logo (varchar)	annee_creation (date)	statut_juridique (varchar)	capital (int)
1	MOTUL	logo.jpg	2017-03-12	SAS	100000
2	ELF	logo.png	2004-12-25	SARL	120000

3 Mettre à jour la table produit pour que le produit soit lié à sa marque

PRODUIT

id (int)	nom (varchar)	prix (double)	poids (double)	stock (int)	id_marque (int)
1	Kit nettoyage	10.90	2.25	100	2
2	Brosse pare-brise	5.90	1.00	78	1

4 Créer la table catégorie de produit contenant un libellé, puis associer la table catégorie de produit avec la table produit

CATEGORIEPRODUIT

id (int)	nom (varchar)
1	Kit nettoyage
2	Brosse pare-brise

PRODUIT

id (int)	nom (varchar)	prix (double)	poids (double)	stock (int)	id_categorieProduit	id_marque (int)
1	Kit nettoyage	10.90	2.25	100	1	2
2	Brosse pare-brise	5.90	1.00	78	1	1

5 Créer la table utilisateur avec un nom, un prénom, une adresse

UTILISATEUR

id (int)	nom (varchar)	prenom (varchar)	adresse (varchar)	code_postal (varchar)	ville (varchar)
1	SAVOCA	Nicolas	6 Rue Omer Louis	69003	Lyon
2	LEGROS	Michel	2 rue du Picon	69049	Saint-Bel

6 Créer une table commande avec une date de commande, et la lier à un utilisateur

COMMANDE

id (int)	date_commande (datetime)	id_utilisateur (int)
1	2025-06-30 22:15:24	2
2	2009-11-17 11:30:54	1

7 Enfin, créer la table de liaison ligne de commande, qui, pour chaque produit commandé, fera le lien entre le produit et la commande, une quantité, et un prix

LIGNE_COMMANDE

id (int)	**id_commande (int)**	**id_produit (int)**	quantite (int)
1	1	1	2
2	1	2	5
3	2	1	7
4	2	2	1

Voici un exemple de MCD (Modèle conceptuel de données) pour représenter ces quelques exercices. Pas d'inquiétude, un chapitre entier sera consacré à la **modélisation**, mais familiarisez-vous avec ce type de notation. On peut y voir que la table de liaison ligne_commande est représentée avec une association.