Chapitre 3 : Bases de données

Nicky Sonigo

Lycée Victor Hugo
PC*

2023/2024

Tables des matières

Système de gestion de bases de données (SGBD)

Un système de gestion de bases de données est un système permettant de traiter un grand nombre d'informations. La base de données peut alors être complétée et modifiée en évitant les incohérences. L'architecture trois tiers (utilisateur, serveur de traitement, serveur de bases de données) permet l'utilisation de la base de données par plusieurs utilisateurs simultanément depuis des postes informatiques différents.

Tables des matières

Tables des matières

Tables, entités, relations

Considérons l'exemple des données d'une bibliothèque, on souhaite en particulier conserver une trace de tous les emprunts de documents.
Les différentes informations concernant un objet concret ou abstrait sont enregistrées dans des **tables** ou **relations**.
Dans un premier temps, on considère les entités (objet concret ou abstrait à propos duquel on souhaite conserver des informations) suivantes :
`Document` : les livres, `Auteur` : les auteurs et `Emprunteur` : les clients de la bibliothèque.

Attributs, colonnes, domaine

Les informations attachées à chaque objet sont appelés les **attributs** ou **colonnes**.

Les attributs de la table `Document` sont : `titre`, `auteur`, `genre`, `parution`, `pages`.

Le **domaine d'un attribut** est l'ensemble des valeurs que peut prendre un attribut.

Attributs, colonnes, domaine

Les informations attachées à chaque objet sont appelés les **attributs** ou **colonnes**.
Les attributs de la table `Document` sont : `titre`, `auteur`, `genre`, `parution`, `pages`.
Le **domaine d'un attribut** est l'ensemble des valeurs que peut prendre un attribut.

Exemple

Pour la table `Document`, les attributs `titre`, `auteur` et `genre` sont des chaînes de caractères ; `parution` et `pages` sont des entiers.

Schéma relationnel d'une table

Le nom d'une table, ses attributs et les domaines des attributs sont donnés dans le **schéma relationnel** de la table sous la forme suivante :
Nom_table{attribut1(domaine1), attribut2(domaine2), ...}

Schéma relationnel d'une table

Le nom d'une table, ses attributs et les domaines des attributs sont donnés dans le **schéma relationnel** de la table sous la forme suivante :
Nom_table{attribut1(domaine1), attribut2(domaine2), ...}

Exemple

Le schéma de la table `Document` est :
Document{titre(string), auteur(string), genre(string), parution(entier), pages(entier)}

Enregistrements, lignes

Chaque description d'un objet dans une table est appelée **enregistrement** ou **ligne**.

Enregistrements, lignes

Chaque description d'un objet dans une table est appelée **enregistrement** ou **ligne**.

Exemple

Par exemple pour la table `Document` :

Document				
titre	auteur	genre	parution	pages
La cousine Bette	Honoré de Balzac	Roman	1846	240
Le feu	Henri Barbusse	Roman	1916	435
De la guerre	Carl von Clausewitz	Traité	1832	240
Mrs Dalloway	Virginia Woolf	Roman	1925	240
Cyrano de Bergerac	Edmond Rostand	Théâtre	1897	280
...	...	...	...	...

Remarque

On retrouve la notion de **relation** vue en mathématiques xRy, étendue ici à un nombre n d'éléments : $R(x_1, x_2, \cdots, x_n)$.
Une table T définit une relation R_T par :

$$R_T(x_1, \ldots, x_n) \Leftrightarrow (x_1, \ldots, x_n) \text{ est un enregistrement la table } T.$$

Tables des matières

Clé, clé primaire

Une **clé** d'une table est un ensemble minimal d'attributs (une partie du schéma) dont la valeur caractérise chaque enregistrement de la table. Parmi les différentes clés possibles (appelées clé candidates), on en choisit une appelée **clé primaire**. Les attributs de la clé primaire sont soulignés dans le schéma relationnel de la table.

Clé, clé primaire

Une **clé** d'une table est un ensemble minimal d'attributs (une partie du schéma) dont la valeur caractérise chaque enregistrement de la table. Parmi les différentes clés possibles (appelées clé candidates), on en choisit une appelée **clé primaire**. Les attributs de la clé primaire sont soulignés dans le schéma relationnel de la table.

Exemple

Pour la table `Document` :
Document{titre(string), auteur(string), genre(string), parution(entier), pages(entier)}

Exemple

Pour la table `Emprunteur`, l'attribut `nom` seul ne suffit pas, (`nom`, `prénom`) non plus.

Exemple

Pour la table `Emprunteur`, l'attribut `nom` seul ne suffit pas, (`nom`, `prénom`) non plus.

S'il n'y a pas de clé naturelle, il possible d'ajouter un attribut (souvent appelé identifiant) qui sera la clé primaire, cet attribut prendra une valeur entière, certains systèmes de gestion de bases de données permettent d'incrémenter automatiquement l'attribut choisi comme clé.

Exemple

Pour la table `Emprunteur`, l'attribut `nom` seul ne suffit pas, (`nom`, `prénom`) non plus.

S'il n'y a pas de clé naturelle, il possible d'ajouter un attribut (souvent appelé identifiant) qui sera la clé primaire, cet attribut prendra une valeur entière, certains systèmes de gestion de bases de données permettent d'incrémenter automatiquement l'attribut choisi comme clé.

Pour la table `Emprunteur` :
Emprunteur{<u>id</u>(entier), nom(string), prénom(string), adresse(string), téléphone(string), mail(string)}

Exemple

Pour la table `Emprunteur`, l'attribut `nom` seul ne suffit pas, (`nom`, `prénom`) non plus.

S'il n'y a pas de clé naturelle, il possible d'ajouter un attribut (souvent appelé identifiant) qui sera la clé primaire, cet attribut prendra une valeur entière, certains systèmes de gestion de bases de données permettent d'incrémenter automatiquement l'attribut choisi comme clé.

Pour la table `Emprunteur` :
Emprunteur{id(entier), nom(string), prénom(string), adresse(string), téléphone(string), mail(string)}

De même : Auteur{nom_auteur(string), date(entier), biographie(string)}

Tables des matières

Association

Une **association** est un lien entre deux entités.

Association

Une **association** est un lien entre deux entités.

Exemple

Par exemple :

- ⋆ chaque livre est lié à un auteur, association : “écrit par”
- ⋆ un emprunteur est lié à des documents, association : “emprunté par”.

Types d'association

Les associations entre deux entités E_1 et E_2 peuvent être de type :

- $1-1$: chaque élément de E_1 est lié à au plus un élément de E_2 et réciproquement ;
- $1-*$: quitte à échanger E_1 et E_2, chaque élément de E_1 est lié à au plus un élément de E_2, mais un élément de E_2 peut être associé à plusieurs éléments de E_1 ;
- $*-*$: chaque élément de E_1 peut être lié à plusieurs de E_2 et réciproquement.

Types d'association

Les associations entre deux entités E_1 et E_2 peuvent être de type :

- $1-1$: chaque élément de E_1 est lié à au plus un élément de E_2 et réciproquement ;
- $1-*$: quitte à échanger E_1 et E_2, chaque élément de E_1 est lié à au plus un élément de E_2, mais un élément de E_2 peut être associé à plusieurs éléments de E_1 ;
- $*-*$: chaque élément de E_1 peut être lié à plusieurs de E_2 et réciproquement.

Dans le cas d'une association $1-*$ ou $1-1$, elle peut être représentée par un attribut dans l'une des tables.

Exemple

L'association "écrit par" est de type $1 - *$: chaque livre a un auteur, mais un auteur peut avoir écrit plusieurs livre. Cette association est représentée par l'attribut `auteur` de la table `Document` qui permet de faire.

Exemple

L'association "écrit par" est de type $1 - *$: chaque livre a un auteur, mais un auteur peut avoir écrit plusieurs livre. Cette association est représentée par l'attribut `auteur` de la table `Document` qui permet de faire.

Exemple

Pour la relation "emprunté par" est également de type $1 - *$, mais on veut avoir des informations complémentaires comme la date d'emprunt et la date de retours prévue. On crée alors une nouvelle table `Emprunt` pour représenter cette association, dont le schéma est :
`Emprunt` : { id_emprunteur(entier), titre(string), nom_auteur(string), date_emprunt(string), date_retour(string) }.

Clé étrangère

On appelle **clé étrangère** d'une table un ensemble d'attributs qui font partie de la clé primaire d'une autre table.

Clé étrangère

On appelle **clé étrangère** d'une table un ensemble d'attributs qui font partie de la clé primaire d'une autre table.

Exemple

Par exemple, dans la table `Document` l'attribut `auteur` est une clé étrangère : c'est la clé primaire de la table `Auteur`. Dans la table `Emprunt` : `id_emprunteur` est une clé étrangère (clé primaire de `Emprunteur`) et {`titre`, `nom_auteur`} est une autre clé étrangère (clé primaire de `Document`).

Tables des matières

Exemple

On est parti du principe qu'un document n'a qu'un seul auteur qui peut être caractérisé par son nom. Ce qui n'est pas très réaliste. Si l'on veut pouvoir indiquer plusieurs auteurs pour un même document, l'association "écrit par" est alors de type $* - *$.

Exemple

On est parti du principe qu'un document n'a qu'un seul auteur qui peut être caractérisé par son nom. Ce qui n'est pas très réaliste. Si l'on veut pouvoir indiquer plusieurs auteurs pour un même document, l'association "écrit par" est alors de type $* - *$.

Pour la représenter, on peut passer par une table `ecrit_par`. Avant cela on change de modèle pour la table `Document`{id_document(entier), titre(string), id_auteur(entier), genre(string), parution(entier), pages(entier)} et `Auteur`{id_auteur(entier), nom_auteur(string), date(string), biographie(string)}.

Exemple

On est parti du principe qu'un document n'a qu'un seul auteur qui peut être caractérisé par son nom. Ce qui n'est pas très réaliste. Si l'on veut pouvoir indiquer plusieurs auteurs pour un même document, l'association "écrit par" est alors de type $* - *$.

Pour la représenter, on peut passer par une table `ecrit_par`. Avant cela on change de modèle pour la table
`Document`{id_document(entier), titre(string), id_auteur(entier), genre(string), parution(entier), pages(entier)} et
`Auteur`{id_auteur(entier), nom_auteur(string), date(string), biographie(string)}.

On propose alors le schéma suivant pour `ecrit_par` :
`ecrit_par`{id_document(entier), id_auteur(entier)}.

Tables des matières

SGBD et langage SQL

Au lycée, on utilisera **MySQL**, avec lequel on travaille en mode "ligne de commande" et une interface graphique (**Workbench**).

Le langage utilisé est une implémentation du SQL (Structured Query Language).

Tables des matières

Utiliser une base de donnée

On commence toujours par renseigner la base de données sur laquelle on travaille :
pour travailler avec une base de données existante

```
USE <database_name>;
```

Utiliser une base de donnée

On commence toujours par renseigner la base de données sur laquelle on travaille :
pour travailler avec une base de données existante

```
USE <database_name>;
```

Exemple

Par exemple :

```
USE bibliotheque;
Database changed
```

Tables des matières

Schéma relationnel d'une table

```
DESCRIBE <table_name>;
```

Schéma relationnel d'une table

```
DESCRIBE <table_name>;
```

Exemple

Pour obtenir le schéma relationnel de la table Emprunteur :

```
DESCRIBE Emprunteur;
```

Tables des matières

Requête simple

Pour récupérer tous les attributs de tous les enregistrements d'une table

```
SELECT * FROM <table_name>;
```

Requête simple

Pour récupérer tous les attributs de tous les enregistrements d'une table

```
SELECT * FROM <table_name>;
```

Exemple

Par exemple :

```
SELECT * FROM Emprunteur
```

Requête simple

Pour récupérer tous les attributs de tous les enregistrements d'une table

```
SELECT * FROM <table_name>;
```

Exemple

Par exemple :

```
SELECT * FROM Emprunteur
```

id	nom	prénom	adresse	telephone	mail
1	Martin	Jacques	11 avenue de Wagram	123456789	...
2	Lagaffe	Gaston	Boulevard Pachéco	381000000	...
3	Martin	Jean-Pierre	Toulon	2147483647	...

Tables des matières

Requête simple

Pour récupérer certains attributs de tous les enregistrements d'une table

```
SELECT <column_name,...> FROM <table_name>;
```

Requête simple

Pour récupérer certains attributs de tous les enregistrements d'une table

```
SELECT <column_name,...> FROM <table_name>;
```

Exemple

Par exemple :

```
SELECT nom, prénom FROM Emprunteur;
```

Requête simple

Pour récupérer certains attributs de tous les enregistrements d'une table

```
SELECT <column_name,...> FROM <table_name>;
```

Exemple

Par exemple :

```
SELECT nom, prénom FROM Emprunteur;
```

```
+-----------+-------------+
| nom       | prénom      |
+-----------+-------------+
| Martin    | Jacques     |
| Lagaffe   | Gaston      |
| Martin    | Jean-Pierre |
+-----------+-------------+
```

Tables des matières

Sélection

Pour récupérer tous les attributs de certains enregistrements d'une table

```
SELECT * FROM <table_name> WHERE <condition>;
```

Sélection

Pour récupérer tous les attributs de certains enregistrements d'une table

```
SELECT * FROM <table_name> WHERE <condition>;
```

Exemple

Par exemple :

```
SELECT * FROM Emprunteur WHERE nom = 'Martin';
```

```
+----+--------+---------+-----------------------+-----------+-----------------+
| id | nom    | prénom  | adresse               | telephone | mail            |
+----+--------+---------+-----------------------+-----------+-----------------+
|  1 | Martin | Jacques | 11 avenue de Wagram | 123456789 | jmartin@free.fr |
+----+--------+---------+-----------------------+-----------+-----------------+
```

Conditions de sélection

Pour créer les conditions de sélection, on peut utiliser les opérateurs : `+`, `-`, `*`, `/` sur les entiers et les flottants, `=`, `<>`, `<`, `<=`, `>`, `>=` sur les entiers, flottants et chaînes de caractères (ordre lexicographique) et `AND`, `OR`, `NOT` sur le booléens.

Conditions de sélection

Pour créer les conditions de sélection, on peut utiliser les opérateurs : `+`, `-`, `*`, `/` sur les entiers et les flottants, `=`, `<>`, `<`, `<=`, `>`, `>=` sur les entiers, flottants et chaînes de caractères (ordre lexicographique) et `AND`, `OR`, `NOT` sur le booléens.

Projections et sélections peuvent être combinées pour récupérer certains attributs de certains enregistrements d'une table

```
SELECT <column_name...> FROM <table_name> WHERE <condition>;
```

Exemple

Par exemple :

```
SELECT nom, adresse, telephone
FROM Emprunteur
WHERE nom = 'Martin';
```

Exemple

Par exemple :

```
SELECT nom, adresse, telephone
FROM Emprunteur
WHERE nom = 'Martin';
```

```
+--------+-----------------------+-----------+
| nom    | adresse               | telephone |
+--------+-----------------------+-----------+
| Martin | 11 avenue de Wagram   | 123456789 |
+--------+-----------------------+-----------+
```

Tables des matières

Renommage

Pour renommer un attribut :

```
SELECT <column_name> AS <nouveau_nom> FROM <table_name>;
```

Renommage

Pour renommer un attribut :

```
SELECT <column_name> AS <nouveau_nom> FROM <table_name>;
```

Exemple

Par exemple :

```
SELECT nom AS NOM, prénom AS Prénom FROM Emprunteur;
```

Renommage

Pour renommer un attribut :

```
SELECT <column_name> AS <nouveau_nom> FROM <table_name>;
```

Exemple

Par exemple :

```
SELECT nom AS NOM, prénom AS Prénom FROM Emprunteur;
```

```
+-----------+-------------+
| NOM       | Prénom      |
+-----------+-------------+
| Martin    | Jacques     |
| Lagaffe   | Gaston      |
| Martin    | Jean-Pierre |
+-----------+-------------+
```

Tables des matières

Mot-clé DISTINCT : permet de supprimer les doublons

```
SELECT nom FROM Emprunteur;
```

```
+----------+
| nom      |
+----------+
| Martin   |
| Lagaffe  |
| Martin   |
| Dupont   |
| Dupond   |
+----------+
```

```
SELECT DISTINCT nom FROM Emprunteur;
```

```
+----------+
| nom      |
+----------+
| Martin   |
| Lagaffe  |
| Dupont   |
| Dupond   |
+----------+
```

Mot-clé LIMIT : permet de limiter le nombre d'enregistrements

```
SELECT nom, prénom FROM Emprunteur LIMIT 5;
```

```
+-----------+-------------+
| nom       | prénom      |
+-----------+-------------+
| Martin    | Jacques     |
| Lagaffe   | Gaston      |
| Martin    | Jean-Pierre |
| Dupont    | Marcel      |
| Dupond    | Louis       |
+-----------+-------------+
```

```
SELECT nom, prénom FROM Emprunteur LIMIT 3;
```

```
+-----------+-------------+
| nom       | prénom      |
+-----------+-------------+
| Martin    | Jacques     |
| Lagaffe   | Gaston      |
| Martin    | Jean-Pierre |
+-----------+-------------+
```

Mot-clé OFFSET : associé à LIMIT permet de spécifier le rang de la première ligne à prendre en compte, en comptant à partir de 0

```
SELECT nom, prénom FROM Emprunteur LIMIT 1 OFFSET 2;
```

```
+-----------+-------------+
| nom       | prénom      |
+-----------+-------------+
| Martin    | Jean-Pierre |
+-----------+-------------+
```

```
SELECT nom, prénom FROM Emprunteur LIMIT 3 OFFSET 1;
```

```
+-----------+-------------+
| nom       | prénom      |
+-----------+-------------+
| Lagaffe   | Gaston      |
| Martin    | Jean-Pierre |
| Dupont    | Marcel      |
+-----------+-------------+
```

Mot-clé ORDER BY : permet de trier les résultats en fonction des colonnes

```
SELECT nom, prénom FROM Emprunteur ORDER BY nom (ASC);
```

```
+-----------+-------------+
| nom       | prénom      |
+-----------+-------------+
| Dupond    | Louis       |
| Dupont    | Marcel      |
| Lagaffe   | Gaston      |
| Martin    | Jacques     |
| Martin    | Jean-Pierre |
+-----------+-------------+
```

Mot-clé ORDER BY : permet de trier les résultats en fonction des colonnes

```
SELECT nom, prénom FROM Emprunteur ORDER BY nom DESC;
```

nom	prénom
Martin	Jacques
Martin	Jean-Pierre
Lagaffe	Gaston
Dupont	Marcel
Dupond	Louis

Mot-clé ORDER BY : permet de trier les résultats en fonction des colonnes

```
SELECT nom, prénom FROM Emprunteur ORDER BY prénom;
```

nom	prénom
Lagaffe	Gaston
Martin	Jacques
Martin	Jean-Pierre
Dupond	Louis
Dupont	Marcel

Mot-clé ORDER BY : permet de trier les résultats en fonction des colonnes

```
SELECT nom, prénom FROM Emprunteur ORDER BY nom ASC, prénom
    DESC;
```

nom	prénom
Dupond	Louis
Dupont	Marcel
Lagaffe	Gaston
Martin	Jean-Pierre
Martin	Jacques

En résumé :

- ⋆ *SQL* est un **langage déclaratif** :
  - On demande avec une requête ce que l'on attend en sortie.
  - On ne dit pas comment le SGBD doit s'y prendre techniquement pour satisfaire la requête.
  - Le SGBD se débrouille : il optimise les requêtes.
- ⋆ *Python* est, au contraire, un **langage impératif** :
  - On décrit comment les choses doivent être traitées.

Tables des matières

Tables des matières

Exemple

Tables des matières

Exemple

Tables des matières

Exemple

Exemple

Tables des matières

Exemple

Tables des matières

Exemple

Tables des matières

Exemple

Tables des matières

Exemple

Tables des matières

Exemple

Tables des matières

Exemple