Introduction aux bases de données relationnelles

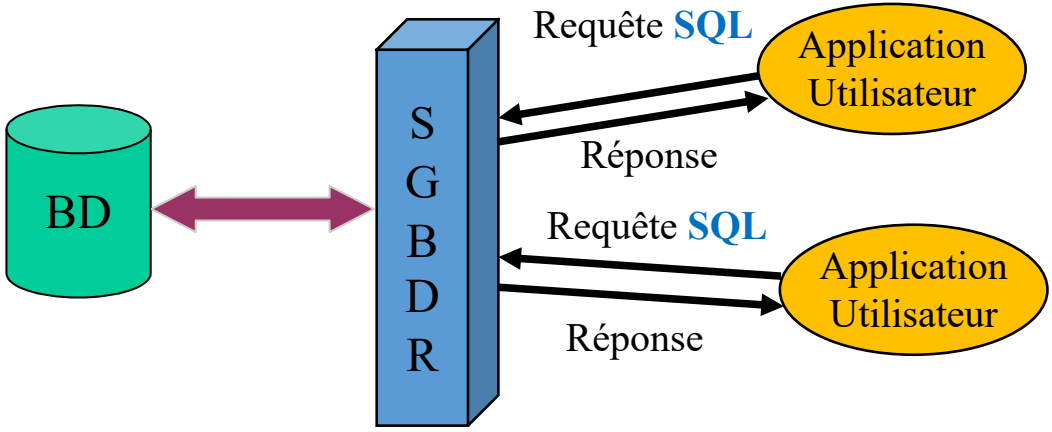
2

SQL - Premières requêtes SQL interrogatives

Jean-François Condotta

Quelques généralités sur le langage SQL

Le langage SQL



- Le langage SQL est un langage normalisé (Standards ANSI-ISO : SQL-86, SQL-89, SQL-92 (SQL2), SQL-99 (SQL3), SQL:2003 …) compris par l'ensemble des SGBDR.
- Le langage SQL est un langage informatique déclaratif (description du résultat).

Sous-langages de SQL

- Le **Langage de Définition de Données** (**LDD**) : Manipulation du schéma de la BD (création des tables, suppression des tables, création des contraintes d'intégrité …)
- Le **Langage de Contrôle de Données** (**LCD**) : Gestion de l'accès aux données (gestion de groupes d'utilisateurs/rôles, gestion des droits sur les objets de la BD …)
- Le **Langage de Contrôle de Transactions** (**LCT**) : Gestion des transactions (Démarrage/Validation/Annulation d'une transaction …)
- Le **Langage de Manipulation de Données** (**LMD**) : Manipulation du contenu des tables de la BD (Insertion/Suppression/Mise à jour de données, extraction de données : requêtes SQL interrogatives)

La base de données SPORTS

PERSONNE

id_pers	nom	prenom	age
0	'Dujardin'	'Marc'	20
1	'Devos'	'Evelyne'	43
2	'Panahi'	'Mahmoud'	38
3	'Buzek'	'Elsa'	50
4	'Amalric'	'Jeanne'	20
5	'Pheonix'	'Arthur'	32

SPORT

id_sport	nom_sport	categorie
0	'Marche'	'Athlétisme'
1	'Rugby'	'Collectif'
2	'Football'	'Collectif'
3	'Karaté'	'Art martial'
4	'Course à pied'	'Athlétisme'
5	'VTT'	'Cyclisme'
6	'Judo'	'Art martial'
7	'Volley-ball'	'Collectif'

PRATIQUE

id_pers	id_sport	nb_heures
0	2	2.00
1	4	1.50
1	5	3.50
2	2	4.00
3	0	**NULL**
4	3	2.50
4	6	2.00
5	0	**NULL**
5	4	**NULL**
5	7	3.00

Exemple de requête SQL interrogative

PERSONNE

id_pers	nom	prenom	age
0	Dujardin	Marc	20
1	Devos	Evelyne	43
2	Panahi	Mahmoud	38
3	Buzek	Elsa	50
4	Amalric	Jeanne	20
5	Pheonix	Arthur	32

- Quels sont l'identifiant et le nom de chaque personne ayant un âge strictement inférieur à 35 ans ? Le résultat sera trié suivant l'ordre alphabétique des noms.

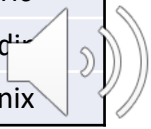
Requête SQL interrogative

```
SELECT id_pers,nom
FROM PERSONNE
WHERE age<35
ORDER BY nom ASC
```

Évaluation de la requête par le SGBDR

Résultat

id_pers	nom
4	Amalric
0	Dujardin
5	Pheonix



Requêtes SQL interrogatives simples

(SELECT … FROM … WHERE … ORDER BY …)

Signification des différentes clauses

SELECT <une liste d'attributs> ➡ { Les attributs pour lesquels nous souhaitons les valeurs.

FROM <un contenu> ➡ { Le contenu à partir duquel sont extraites les données souhaitées. Une table, une liste de tables, …

WHERE <une condition sur les tuples> ➡ { Une condition permettant de spécifier les tuples du contenu à garder.

ORDER BY <une liste d'attributs avec ASC/DESC> ➡ { Une liste d'attributs dont les valeurs sont utilisées pour trier le résultat.

- Les clauses WHERE et ORDER BY sont optionnelles.
- L'ordre des clauses est toujours SELECT/FROM/WHERE/ORDER BY.

La clause SELECT : ordre des attributs

PERSONNE

id_pers	nom	prenom	age
0	Dujardin	Marc	20
1	Devos	Evelyne	43
2	Panahi	Mahmoud	38
3	Buzek	Elsa	50
4	Amalric	Jeanne	20
5	Pheonix	Arthur	32

- Quels sont l'identifiant et le nom de chaque personne ?

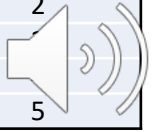
```
SELECT id_pers,nom
FROM PERSONNE
```

id_pers	nom
0	Dujardin
1	Devos
2	Panahi
3	Buzek
4	Amalric
5	Pheonix

- Quels sont le nom et l'identifiant de chaque personne ?

```
SELECT nom,id_pers
FROM PERSONNE
```

nom	id_pers
Dujardin	0
Devos	1
Panahi	2
Buzek	
Amalric	
Pheonix	5



Désignation des attributs

PERSONNE

id_pers	nom	prenom	age
0	Dujardin	Marc	20
1	Devos	Evelyne	43
2	Panahi	Mahmoud	38
3	Buzek	Elsa	50
4	Amalric	Jeanne	20
5	Pheonix	Arthur	32

- Quels sont l'identifiant et le nom de chaque personne ?

Désignation courte des attributs

```
SELECT id_pers,nom
FROM PERSONNE
```

Désignation longue des attributs

```
SELECT PERSONNE.id_pers,PERSONNE.nom
FROM PERSONNE
```

id_pers	nom
0	Dujardin
1	Devos
2	Panahi
3	Buzek
4	Amalric
5	Pheonix

- La **désignation longue** sera **parfois nécessaire** lorsque plusieurs attributs de même nom se trouve dans différentes tables utilisées dans la clause FROM !

Désignation de l'ensemble des attributs d'une table

PERSONNE

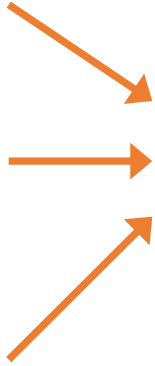
id_pers	nom	prenom	age
0	Dujardin	Marc	20
1	Devos	Evelyne	43
2	Panahi	Mahmoud	38
3	Buzek	Elsa	50
4	Amalric	Jeanne	20
5	Pheonix	Arthur	32

- Quels sont l'identifiant, le nom, le prénom et l'âge de chaque personne ?

```
SELECT id_pers,nom,prenom,age
FROM PERSONNE
```

```
SELECT *
FROM PERSONNE
```

```
SELECT PERSONNE.*
FROM PERSONNE
```



id_pers	nom	prenom	age
0	Dujardin	Marc	20
1	Devos	Evelyne	43
2	Panahi	Mahmoud	38
3	Buzek	Elsa	50
4	Amalric	Jeanne	20
5	Pheonix	Arthur	32

Doublons possibles

- La relation correspondant à l'évaluation d'une requête interrogative peut contenir des **doublons** (des tuples identiques).

PERSONNE

id_pers	nom	prenom	age
0	Dujardin	Marc	20
1	Devos	Evelyne	43
2	Panahi	Mahmoud	38
3	Buzek	Elsa	50
4	Amalric	Jeanne	20
5	Pheonix	Arthur	32

- Quels sont les âges des personnes ?

```
SELECT age
FROM PERSONNE
```

age
20
43
38
50
20
32

Supprimer les doublons : SELECT DISTINCT

- Le mot clé DISTINCT peut être ajouté après le mot clé SELECT pour supprimer les doublons.

PERSONNE

id_pers	nom	prenom	age
0	Dujardin	Marc	20
1	Devos	Evelyne	43
2	Panahi	Mahmoud	38
3	Buzek	Elsa	50
4	Amalric	Jeanne	20
5	Pheonix	Arthur	32

- Quels sont les **différents** âges des personnes ?

```
SELECT DISTINCT age
FROM PERSONNE
```

age
20
43
38
50
32

Renommer les colonnes du résultat

PERSONNE

id_pers	nom	prenom	age
0	Dujardin	Marc	20
1	Devos	Evelyne	43
2	Panahi	Mahmoud	38
3	Buzek	Elsa	50
4	Amalric	Jeanne	20
5	Pheonix	Arthur	32

- Quels sont l'identifiant et le nom de chaque personne ? La colonne correspondant aux identifiants devra être nommée **identifiant**.

Utilisation du mot clé **AS** suivi du nouveau nom entre ".

```
SELECT id_pers AS "identifiant",nom
FROM PERSONNE
```

identifiant	nom
0	Dujardin
1	Devos
2	Panahi
3	[illegible]
4	Amalric
5	Pheonix

La clause ORDER BY : trier le résultat

- La clause optionnelle **ORDER BY** permet d'ordonner les tuples résultant de l'évaluation de la requête selon les valeurs de un ou plusieurs attributs.
- Les mots clés **ASC** et **DESC** sont utilisés pour un ordre respectivement ascendant ou descendant.

PERSONNE

id_pers	nom	prenom	age
0	Dujardin	Marc	20
1	Devos	Evelyne	43
2	Panahi	Mahmoud	38
3	Buzek	Elsa	50
4	Amalric	Jeanne	20
5	Pheonix	Arthur	32

- Quels sont l'identifiant, le nom et l'âge de chaque personne ? Le résultat sera trié **suivant l'ordre alphabétique des noms**.

```
SELECT id_pers,nom,age
FROM PERSONNE
ORDER BY nom ASC
```

id_pers	nom	age
4	Amalric	20
3	Buzek	50
1	Devos	43
0	Dujardin	20
2	Panahi	38
5	Pheonix	32

La clause ORDER BY : trier le résultat

- Résultat trié **suivant l'ordre alphabétique inverse des noms**.

```
SELECT id_pers,nom,age
FROM PERSONNE
ORDER BY nom DESC
```

id_pers	nom	age
5	Pheonix	32
2	Panahi	38
0	Dujardin	20
1	Devos	43
3	Buzek	50
4	Amalric	20

- Résultat trié **suivant l'ordre décroissant des âges et suivant l'ordre alphabétique des noms en cas d'égalité des âges.**

```
SELECT id_pers,nom,age
FROM PERSONNE
ORDER BY age DESC,nom ASC
```

id_pers	nom	age
3	Buzek	50
1	Devos	43
2	Panahi	38
5	Pheonix	32
4	Amalric	20
0	Dujardin	20

- Par défaut l'ordre est ascendant, le mot clé ASC est donc optionnel.

ORDER BY age DESC,nom ASC ↔ ORDER BY age DESC,nom

La clause WHERE

- La clause WHERE est optionnelle.
- La clause WHERE permet de spécifier une condition (une expression booléenne) qui sera évaluée sur chaque tuple du contenu spécifié par la clause FROM. **Seuls les tuples satisfaisant la condition seront gardés pour établir le résultat de la requête.**

PERSONNE

id_pers	nom	prenom	age
0	Dujardin	Marc	20
1	Devos	Evelyne	43
2	Panahi	Mahmoud	38
3	Buzek	Elsa	50
4	Amalric	Jeanne	20
5	Pheonix	Arthur	32

- Quels sont l'identifiant et le nom de chaque personne ayant un âge strictement inférieur à 35 ans ?

```
SELECT id_pers,nom
FROM PERSONNE
WHERE age<35
```

id_pers	nom
0	Dujardin
4	Amalric
5	Pheonix

Condition de sélection pour une clause WHERE

- Possibilité de comparer des valeurs d'attributs et des constantes à l'aide de différents opérateurs de comparaison :
  - **=** (égalité), **<** (strictement plus petit), **<=** (plus petit ou égal), **>** (strictement plus grand**), >=** (plus grand ou égal), **<>** (différent).
- Les opérateurs de comparaison ne doivent pas être utilisés avec la valeur NULL qui n'est pas en réalité une valeur mais un marqueur d'absence de valeur. Des opérateurs spécifiques devront être utilisés.
- Des conditions de sélection complexes peuvent être définies à partir des opérateurs logiques :
  - **AND** (et logique), **OR** (ou logique), **NOT** (négation logique).

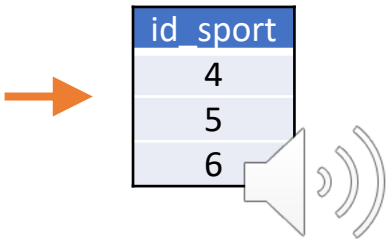
Condition de sélection pour une clause WHERE

SPORT

id_sport	nom_sport	categorie
0	'Marche'	'Athlétisme'
1	'Rugby'	'Collectif'
2	'Football'	'Collectif'
3	'Karaté'	'Art martial'
4	'Course à pied'	'Athlétisme'
5	'VTT'	'Cyclisme'
6	'Judo'	'Art martial'
7	'Volley-ball'	'Collectif'

- Quels sont les identifiants des sports supérieurs ou égaux à 4 associés à des sports d'une catégorie différente de la catégorie 'Collectif' ?

```
SELECT id_sport
FROM SPORT
WHERE (id_sport >= 4) AND (categorie <> 'Collectif')
```



id_sport
4
5
6

Les prédicats IS NULL et IS NOT NULL

- Possibilité de tester l'absence et la présence d'une valeur avec les prédicats **IS NULL** et **IS NOT NULL**.

PRATIQUE

id_pers	id_sport	nb_heures
0	2	2.00
1	4	1.50
1	5	3.50
2	2	4.00
3	0	**NULL**
4	3	2.50
4	6	2.00
5	0	**NULL**
5	4	**NULL**
5	7	3.00

- Quels sont les identifiants des personnes et des sports pour lesquels **nous ne connaissons pas** le nombre d'heures pratiquées ?

```
SELECT id_pers,id_sport
FROM PRATIQUE
WHERE nb_heures IS NULL
```

id_pers	id_sport
3	0
5	0
5	4

Les prédicats IS NULL et IS NOT NULL

PRATIQUE

id_pers	id_sport	nb_heures
0	2	2.00
1	4	1.50
1	5	3.50
2	2	4.00
3	0	**NULL**
4	3	2.50
4	6	2.00
5	0	**NULL**
5	4	**NULL**
5	7	3.00

- Quels sont les identifiants des personnes et des sports pour lesquels **nous connaissons** le nombre d'heures pratiquées ?

```
SELECT id_pers,id_sport
FROM PRATIQUE
WHERE nb_heures IS NOT NULL
```

id_pers	id_sport
0	2
1	4
1	5
2	2
4	3
4	6
5	7

À Bientôt …