

# Bases de données

Souhila KACI

Partie 2

# Le langage SQL

- SQL : **Strucured Query Language** (Langage de requêtes structurées).
- SQL est un langage de requêtes basé sur l'**algèbre relationnelle**.
- SQL permet
  - la description du schéma de la base de données : création des tables, suppression des tables, création des contraintes d'intégrité.
  - la modification du contenu des tables : ajout, suppression, modification des tuples des tables.
  - l'interrogation sur le contenu des tables : projection, sélection, jointure, tri, agrégation.
- SQL est un langage normé (norme ANSI).
- Les éditeurs de SGBDR proposent diverses extensions à cette norme.

- Description du schéma de la base de données : Langage de Description des Données.
- Modification du contenu des tables : Langage de Modification des Données.
- Interrogation sur le contenu des tables : Langage d'Interrogation des Données.

- Le résultat d'une requête interrogative sera toujours une table/une relation.
- Une requête interrogative (simple) est de la forme :  
**SELECT**  $\langle$ liste d'attribut(s) $\rangle$  : colonnes/attributs que l'on veut garder pour affichage  
**FROM**  $\langle$ liste de table(s) $\rangle$  : tables sur lesquelles porte la requête
- Par la suite nous verrons qu'une requête interrogative peut contenir d'autres clauses que les clauses **SELECT** et **FROM**.

# La projection

- Afficher certaines colonnes d'une table.
- Les colonnes sélectionnées doivent obligatoirement appartenir à la table.

```
SELECT TABLE.att1, TABLE.att2 ... FROM TABLE;
```

```
SELECT ProduitVrac.designation FROM ProduitVrac;
```

ProduitVrac	
codePV	designation
'P01'	'sucre'
'P02'	'poivre'
'P03'	'sel'

⇒

designation
'sucre'
'poivre'
'sel'

Lorsqu'il n'y a pas d'ambiguïté pour la désignation d'un attribut il est possible d'omettre le nom de la table dans sa désignation :

```
SELECT designation FROM ProduitVrac;
```

- Le caractère générique \*
- Il sert à sélectionner toutes les colonnes pour la projection.

`SELECT * FROM ProduitVrac;`

ProduitVrac	
codePV	designation
'P01'	'sucre'
'P02'	'poivre'
'P03'	'sel'

⇒

codePV	designation
'P01'	'sucre'
'P02'	'poivre'
'P03'	'sel'

- Par défaut les doublons ne sont pas supprimés.

```
SELECT codePV FROM ProduitCond;
```

ProduitCond			
codePC	Poids	Volume	codePV
'C012'	2.3	0.69	'P01'
'C253'	1	0.25	'P01'
'C258'	2	0.4	'P01'
'C693'	2	0.45	'P02'

⇒

codePV
'P01'
'P01'
'P01'
'P02'



# Suppression des doublons

- Pour supprimer les doublons : Utiliser le mot clé **DISTINCT**

**SELECT DISTINCT** codePV FROM ProduitCond;

ProduitCond			
codePC	Poids	Volume	codePV
'C012'	2.3	0.69	'P01'
'C253'	1	0.25	'P01'
'C258'	2	0.4	'P01'
'C693'	2	0.45	'P02'

⇒

codePV
'P01'
'P02'

SELECT DISTINCT codePC,codePV FROM ProduitCond;

ProduitCond			
codePC	Poids	Volume	codePV
'C012'	2.3	0.69	'P01'
'C253'	1	0.25	'P01'
'C258'	2	0.4	'P01'
'C693'	2	0.45	'P02'

⇒

codePC	codePV
'C012'	'P01'
'C253'	'P01'
'C258'	'P01'
'C693'	'P02'

- Par défaut, le résultat des requêtes n'est pas ordonné.
- Heureusement, il est possible de choisir un ordre.
- On peut ordonner suivant plusieurs colonnes.
- Le mot clé : **ORDER BY**
- Pour chaque critère, le tri peut être ascendant ou descendant DESC.
- Par défaut il est ascendant.
- Le tri peut également être fait à partir d'une expression construite à partir d'attributs.

**ORDER BY** nom\_col1 | num\_col1 [DESC] ...

**SELECT \* FROM** Commande **ORDER BY** DateCom;  
**SELECT \* FROM** Commande **ORDER BY** 4;

Commande			
numCom	codePC	qte	DateCom
'CD01'	'C012'	25	12/05/2002
'CD02'	'C693'	14	09/11/2002
'CD03'	'C012'	4	03/07/2002
'CD04'	'C012'	11	02/06/2002
'CD05'	'C693'	71	09/01/2003

⇒

numCom	codePC	qte	DateCom
'CD01'	'C012'	25	12/05/2002
'CD04'	'C012'	11	02/06/2002
'CD03'	'C012'	4	03/07/2002
'CD02'	'C693'	14	09/11/2002
'CD05'	'C693'	71	09/01/2003

**SELECT \* FROM Commande ORDER BY  
codePC,DateCom DESC;**

Commande			
numCom	codePC	qte	DateCom
'CD01'	'C012'	25	12/05/2002
'CD02'	'C693'	14	09/11/2002
'CD03'	'C012'	4	03/07/2002
'CD04'	'C012'	11	02/06/2002
'CD05'	'C693'	71	09/01/2003

⇒

numCom	codePC	qte	DateCom
'CD05'	'C693'	71	09/01/2003
'CD02'	'C693'	14	09/11/2002
'CD03'	'C012'	4	03/07/2002
'CD04'	'C012'	11	02/06/2002
'CD01'	'C012'	25	12/05/2002

**SELECT \* FROM Commande ORDER BY codePC  
ASC, DateCom DESC;**

Commande			
numCom	codePC	qte	DateCom
'CD01'	'C012'	25	12/05/2002
'CD02'	'C693'	14	09/11/2002
'CD03'	'C012'	4	03/07/2002
'CD04'	'C012'	11	02/06/2002
'CD05'	'C693'	71	09/01/2003

⇒

numCom	codePC	qte	DateCom
'CD03'	'C012'	4	03/07/2002
'CD04'	'C012'	11	02/06/2002
'CD01'	'C012'	25	12/05/2002
'CD05'	'C693'	71	09/01/2003
'CD02'	'C693'	14	09/11/2002

- Permet l'affichage de certaines lignes, qui vérifient un critère donné.
- Le critère est une expression booléenne plus ou moins compliquée.
- Le mot clé **WHERE**

SELECT att1,att2 ... FROM table WHERE condition;

SELECT \* FROM ProduitCond WHERE (codePV='P01');

ProduitCond			
codePC	Poids	Volume	codePV
'C012'	2.3	0.69	'P01'
'C253'	1	0.25	'P01'
'C258'	2	0.4	'P01'
'C693'	2	0.45	'P02'

⇒

codePC	Poids	Volume	codePV
'C012'	2.3	0.69	'P01'
'C253'	1	0.25	'P01'
'C258'	2	0.4	'P01'

Opérateur	Numérique	Chaîne de caractères	Date/Heure
=	égal	identique	en même temps
<>	différent	différent	pas en même temps
>	supérieur	supérieure	après
<	inférieur	inférieure	avant
		...	

- Et les connecteurs logiques habituels : AND, OR, NOT.
- Attention aux priorités.
- Il en existe d'autres, nous les verrons plus tard.



# Un petit exercice (1)

Voiture	<u>Immatriculation</u>	Marque	Annee	Prix	<u>IdProprio</u>
	'1111AA01'	'Toyota'	1997	16 000	'Id01'
	'2222BB02'	'Peugeot'	2000	31 200	'Id01'
	'3333CC03'	'Fiat'	1997	2 000	'Id03'
	'4444DD13'	'Fiat'	1995	30 300	'Id02'
	'5555EE62'	'Renault'	1997	21 000	'Id02'
	'6666FF59'	'Opel'	1999	2 900	'Id01'
	'7777ZZ75'	'Ford'	1998	22 222	'Id03'

Personnes	<u>IdProprio</u>	Nom	Prenom	Naissance
	'Id01'	'Martin'	'Paul'	01/02/1967
	'Id02'	'Duval'	'Jean'	03/09/1980
	'Id03'	'Dupond'	'Laurence'	01/01/1945
	'Id04'	'Durand'	'Julie'	03/03/1985

## Un petit exercice (2)

- 1 Liste des immatriculations.
- 2 Liste des voitures de 1996.
- 3 Voitures qui appartiennent au proprio Id01.
- 4 Liste des voitures entre 10000 et 20000 euros.
- 5 Différentes marques de voitures.

- Pour des raisons de commodité ou de clarté, il est possible de renommer les colonnes de la table résultat.
- Le mot clé **AS**

```
SELECT codePC AS Code_Conditionné FROM Commande;
```

Commande			
numCom	codePC	quantite	DateCom
'CD01'	'C012'	25	12/05/2002
'CD02'	'C693'	14	09/11/2002
'CD03'	'C012'	4	03/07/2002
'CD04'	'C012'	11	02/06/2002
'CD05'	'C693'	71	09/01/2003

⇒

Code_Conditionné
'C012'
'C693'
'C012'
'C012'
'C693'

- Pour des raisons de commodité ou de clarté, il est possible de renommer les tables dans une requête.
- On utilise aussi le mot clé **AS**

SELECT codePC **AS** Code\_Conditionné FROM Commande **AS** CMD where CMD.quantite>20;

Commande			
numCom	codePC	quantite	DateCom
'CD01'	'C012'	25	12/05/2002
'CD02'	'C693'	14	09/11/2002
'CD03'	'C012'	4	03/07/2002
'CD04'	'C012'	11	02/06/2002
'CD05'	'C693'	71	09/01/2003

  

Code_Conditionné
'C012'
'C693'

⇒

# Attributs Calculés

## Création de nouvelles colonnes

- On peut créer de nouvelles colonnes qui sont “construites” à partir de colonnes existantes.

```
SELECT Poids*0.9 AS PoidsNet FROM ProduitCond;
```

ProduitCond			
codePC	Poids	Volume	codePV
'C012'	2.3	0.69	'P01'
'C253'	1	0.25	'P01'
'C258'	2	0.4	'P01'
'C693'	2	0.45	'P02'

⇒

PoidsNet
2,07
0,9
1,8
1,8

```
SELECT codePC,Poids,Volume,Volume/Poids AS Densité  
FROM ProduitCond;
```

ProduitCond			
codePC	Poids	Volume	codePV
'C012'	2.3	0.69	'P01'
'C253'	1	0.25	'P01'
'C258'	2	0.4	'P01'
'C693'	2	0.45	'P02'

⇒

CodePC	Poids	Volume	Densité
'C012'	2.3	0.69	0,3
'C253'	1	0.25	0.25
'C258'	2	0.4	0,2
'C693'	2	0.45	0,22

# Attributs calculés

## Fonctions d'agrégation

- Obtention de valeurs agrégées sur une colonne
  - Somme SUM, moyenne AVG
  - Minimum MIN, Maximum MAX
  - Nombre de lignes COUNT

ProduitCond			
codePC	Poids	Volume	codePV
'C012'	2.3	0.69	'P01'
'C253'	1	0.25	'P01'
'C258'	2	0.4	'P01'
'C693'	2	0.45	'P02'

- $\text{MIN}(\text{codePC}) = \text{'C012'}$
- $\text{SUM}(\text{Poids}) = 7.3$
- $\text{Max}(\text{Volume}) = 0.69$
- $\text{Count}(\text{codePC}) = 4$



- **SELECT SUM(c1),AVG(c2) ... FROM ... WHERE ...;**
- Ceci nous donne, pour un nombre quelconque de lignes, **une seule et unique valeur.**
- Nous obtenons un résultat agrégé à partir de l'ensemble des valeurs d'une colonne.

**SELECT SUM(Poids) FROM ProduitCond;**

ProduitCond			
codePC	Poids	Volume	codePV
'C012'	2.3	0.69	'P01'
'C253'	1	0.25	'P01'
'C258'	2	0.4	'P01'
'C693'	2	0.45	'P02'

⇒

Sum(Poids)
7.3

```
SELECT COUNT(*) AS nbP, AVG(Volume) AS volMoyen,  
MIN(Volume) AS volMin FROM ProduitCond;
```

ProduitCond			
codePC	Poids	Volume	codePV
'C012'	2.3	0.69	'P01'
'C253'	1	0.25	'P01'
'C258'	2	0.4	'P01'
'C693'	2	0.45	'P02'

⇒

nbP	VolMoyen	VolMin
4	0.45	0.25

- Le résultat d'un opérateur d'agrégation fournit une unique valeur.
- Une table à une seule ligne.

**SELECT codePC, SUM(Poids) FROM ProduitCond;**

ProduitCond			
codePC	Poids	Volume	codePV
'C012'	2.3	0.69	'P01'
'C253'	1	0.25	'P01'
'C258'	2	0.4	'P01'
'C693'	2	0.45	'P02'



**IMPOSSIBLE**

- Lorsqu'une requête combine des opérations de sélection et de calcul, la sélection est toujours faite **AVANT** le calcul.

**SELECT SUM(Poids) FROM ProduitCond WHERE Volume < 0.5;**

ProduitCond			
codePC	Poids	Volume	codePV
'C012'	2.3	0.69	'P01'
'C253'	1	0.25	'P01'
'C258'	2	0.4	'P01'
'C693'	2	0.45	'P02'

✓  
✓  
✓

⇒

Sum(Poids)
5

```
SELECT SUM(Poids) AS pBrut, SUM(Poids)*0.9 AS pNet  
FROM ProduitCond WHERE Volume*2 < 1;
```

ProduitCond		
codePC	Poids	Volume
'C012'	2.3	0.69
'C253'	1	0.25
'C258'	2	0.4
'C693'	2	0.45

✓  
✓  
✓

⇒

pBrut	pNet
5	4.5

Opérateur	Numérique	Chaîne	Date/Heure
Count	Nombre de valeurs connues		
MAX	Le plus grand	Le plus grand	Le plus tard
MIN	Le plus petit	Le plus petit	Le plus tôt
SUM	La somme	#####	#####
AVG	La moyenne	#####	#####

**SELECT COUNT(\*) as Nb FROM ProduitCond;**

ProduitCond			
codePC	Poids	Volume	codePV
'C012'	2.3	0.69	'P01'
'C253'	1	0.25	'P01'
'C258'	2	0.4	'P01'
'C693'	2	0.45	'P02'

⇒

Nb
4



**SELECT COUNT(Poids) as Nb FROM ProduitCond;**

ProduitCond			
codePC	Poids	Volume	codePV
'C012'	2.3	0.69	'P01'
'C253'	1	0.25	'P01'
'C258'	2	0.4	'P01'
'C693'	2	0.45	'P02'

⇒

Nb
4

**SELECT COUNT(DISTINCT Poids) as Nb FROM  
ProduitCond;**

ProduitCond			
codePC	Poids	Volume	codePV
'C012'	2.3	0.69	'P01'
'C253'	1	0.25	'P01'
'C258'	2	0.4	'P01'
'C693'	2	0.45	'P02'

⇒

Nb
3

- Un attribut donné sous certaines conditions peut avoir la valeur NULL.
- Une valeur NULL correspond à l'absence d'une valeur.
- La valeur NULL n'est pas la chaîne de caractères vide ' ' ou la valeur 0 !
- Pour tester si la valeur d'un attribut A est NULL on ne doit pas effectuer le test **A=NULL** (cela retournera systématiquement faux, comme tout autre prédicat non spécifique à cette valeur).

- Pour tester si la valeur d'un attribut A est NULL on doit effectuer le test **A IS NULL**.
- Pour tester si la valeur d'un attribut A n'est pas NULL on peut effectuer le test **A IS NOT NULL**.
- Lors de l'utilisation de ORDER BY (ASC) les valeurs NULL sont présentées en premier (en dernier avec DESC).

# La valeur NULL

ProduitVrac	
codePV	designation
'P01'	'sucre'
'P02'	'poivre'
'P03'	NULL

SELECT \* FROM ProduitVrac WHERE designation=NULL;

**FAUX**

SELECT \* FROM ProduitVrac WHERE designation is NULL;

ProduitVrac	
codePV	designation
'P01'	'sucre'
'P02'	'poivre'
'P03'	NULL

⇒

codePV	designation
'P03'	NULL

SELECT \* FROM ProduitVrac WHERE designation is NOT NULL;

ProduitVrac	
codePV	designation
'P01'	'sucre'
'P02'	'poivre'
'P03'	NULL

⇒

codePV	designation
'P01'	'sucre'
'P02'	'poivre'

Les fonctions d'agrégat COUNT( ), MIN( ), AVG( ) ne prennent pas en compte la valeur NULL.

```
SELECT COUNT(designation) FROM ProduitVrac;
```

ProduitVrac	
codePV	designation
'P01'	'sucre'
'P02'	'poivre'
'P03'	NULL

Résultat : 2

## Un petit exercice (1)

Voiture	<u>Immatriculation</u>	Marque	Annee	Prix	<u>IdProprio</u>
	'1111AA01'	'Toyota'	1997	16 000	'Id01'
	'2222BB02'	'Peugeot'	2000	31 200	'Id01'
	'3333CC03'	'Fiat'	1997	2 000	'Id03'
	'4444DD13'	'Fiat'	1995	30 300	'Id02'
	'5555EE62'	'Renault'	1997	21 000	'Id02'
	'6666FF59'	'Opel'	1999	2 900	'Id01'
	'7777ZZ75'	'Ford'	1998	22 222	'Id03'

Personnes	<u>IdProprio</u>	Nom	Prenom	Naissance
	'Id01'	'Martin'	'Paul'	01/02/1967
	'Id02'	'Duval'	'Jean'	03/09/1980
	'Id03'	'Dupond'	'Laurence'	01/01/1945
	'Id04'	'Durand'	'Julie'	03/03/1985

## Exercice (2)

- 1 Afficher le nombre total de voitures.
- 2 Afficher le nombre total de FIAT.
- 3 Afficher le prix moyen et le prix total de l'ensemble des voitures en francs et en euros.
- 4 Idem, mais uniquement pour les voitures de moins de 1996.
- 5 Afficher le nombre de voitures qui coûtent moins de 20 000 euros.
- 6 Afficher le nombre de voitures que possède Laurence Dupond.