Initiation à SQL

1. Requête :

interroge la base de données et retourne une table

1.1. Retourner le contenu d'une table

```
SELECT * FROM nom_table
```

ex : `SELECT * FROM produit;`

Les lignes sont données dans un ordre quelconque

`'*' : designe toutes les colonnes;`

1.2. N'obtenir que certaines colonne

```
SELECT nom_col1,nom_col2 FROM nom_table;
```

ex : `SELECT desc_produit FROM produit;`

1.3. Selectionner certaines lignes

```
SELECT nom_col1,nom_col2 FROM nom_table WHERE 'condition';
```

ne retourne que les ligne respectant la condition **ex** :`SELECT desc_produit,prix FROM produit WHERE prix <= 50;`

1.4. Remarque

Les conditions peuvent porter sur des colonnes qu'on ne veut pas.

ex :

```
SELECT id_magasin,nom_magasin FROM magasin WHERE adresse='Lyon';
```

2. Opération de comparaison :

`<,<=,..., =, <> <=> !=, BETWEEN`

2.1. Comparaison chaîne de caractères

Égalité :

égalité : `=, LIKE <, >, <=` **(Ordre lexicographique)**

Recherche inéxacte :

- `%` remplace 0, 1 ou plusieurs caractère
- `-` remplace 1 et un seul caractère

ex :`SELECT * FROM produit WHERE desc_produit LIKE 'tab%'`

3 lignes 'tab'ouret,'tab'ouret,'tab'le

2.2. Recherche dans un ensemble fini de valeur :

`IN`

```
SELECT desc_produit,prix FROM produit WHERE desc_produit IN ('chaise',
'tabouret', 'fauteuil');
```

2.3. Connecteur logique

AND, OR, NOT

ex : Les produits qui sont des tabouret ou des chaises à plus de 35 euros

```
SELECT * FROM produit WHERE (desc_produit LIKE 'tabouret' OR
desc_produit LIKE 'chaise') AND prix > 35
```

2.4. Eliminer les doublons

```
SELECT DISTINCT desc_produit FROM produit;
```

tabouret, chaise, fauteuil.

2.5. Requête d'agrégation simple

ex : quantité total en stock

```
SELECT SUM(quantite) FROM stock;
```

ex : Moyenne des prix des tabourets

```
SELECT AVG(prix) FROM produit WHERE desc_produit = 'tabouret'
```

Autres fonctions

MIN, MAX, COUNT

Avec DISTINCT

On considère la table A

```
SELECT SUM(x), SUM(DISTINCT x) FROM A;
```

COUNT

compte le nombre de ligne

```
SELECT COUNT(*), COUNT(desc_produit),COUNT(DISTINCT desc_produit) FROM
produit;
= [5, 5, 4]
```

3. Complements

Donner des noms aux colonne de la table résultat et faire des opération sur les colonnes.

ex : Simuler une Augmentation de 10% des prix

```
SELECT id_produit, desc_produit, prix * 1,1 AS nouveau_prix FROM
produit;
```

3.1. Remarque

- Ne modifie pas la table.
- On ne peut pas utiliser le nom de colonne 'nouveau_prix' dans le WHERE de la requête.

Par-contre on peut utiliser:

```
SELECT id_produit, desc_produit, prix * 1,1 AS nouveau_prix FROM
produit WHERE prix * 1.1 > 100;
```