Memento SQL

Tout est relation : on traite le quoi, pas le comment.
A la fin de chaque requête SQL, on écrit un point-virgule ;

Fonctions d'agrégation :

MIN(a) MAX(a) AVG(a) SUM(a) COUNT(a)

Projection : SELECT attributs *(sélection de colonnes…)*

avec fct d'agreg et/ou AS et/ou DISTINCT et/ou * et/ou FROM

```
SELECT nom, 2*AVG(note) , MAX(moyenne)      séparés par virgule
SELECT AVG(note ) AS moy                    renommage
SELECT DISTINCT nom                         enlève doublon
SELECT * FROM classe                        tous les attributs de classe
```

Sélection : WHERE conditions *(sélection de lignes…)*

AND, OR, NOT, IN, BETWEEN, <, <=, >, >=, =, <>, LIKE, _ et %

```
SELECT nom, note
FROM classe
WHERE (note > 15) OR (note IN (10,11)) AND (nom LIKE 'A%')
```

Jointure : FROM table1 JOIN table2 ON condition

JOIN = INNER JOIN, NATURAL JOIN si attributs communs

Pour lever toute ambigüité, en particulier dans le cas des auto-jointures : préfixer par le nom de la table.

```
SELECT …
FROM T1 NATURAL JOIN T2
```

Il y a beaucoup d'autres syntaxes possibles…

```
SELECT …
FROM T1 JOIN T2 JOIN T3
ON T1.a1 = T2.a2 AND T3.a3 = T2.b2
```

```
SELECT …
FROM T1 INNER JOIN T2
ON T1.a1 = T2.a2
```

Classement : ORDER BY attributs … (ASC ou DES)

```
ORDER BY note                       croissant par défaut
ORDER BY note DESC                  décroissant
ORDER BY note DESC, nom ASC         tri selon note decr. puis selon nom alphab.
```

Groupement : *(sélection de tuples…)*

GROUP BY attributs + HAVING conditions

Attention, les attributs doivent être communs à tout le groupe !

```
SELECT anonymat
FROM eleve
NATURAL JOIN DS
GROUP BY anonymat
HAVING SUM(note1+note2) BETWEEN 15.0 AND 20.0
```

Requêtes imbriquées : SELECT + FROM + WHERE + … SELECT…

Attention parenthèses obligatoires ! L'indentation permet d'y voir clair.

```
SELECT nom
FROM eleve
WHERE note = (
        SELECT MAX(note) FROM eleve)
```

Opérations ensemblistes : Combinent résultats de plusieurs SELECT

UNION ou INTERSECT ou EXCEPT ou UNION ALL (sans doublon)

```
SELECT nom FROM classe1                 Les SELECT ont le même nombre
UNION / INTERSECT / EXCEPT              d'attributs, de types compatibles
SELECT nom FROM classe2                 toutes orthographes de dupon*
WHERE LOWER(nom) LIKE 'dupon_'          insensible à la casse
```

Calcul :

```
SELECT
  (SELECT SUM(note*coeff) FROM table )
  /
  (SELECT SUM(coeff*1.) FROM table)
```

*Le *1. permet de forcer le type float pour* effectuer *effectuer une division décimale !*

NULL :

Attention à l'évaluation des attributs en présence de NULL !
Les opérateurs de comparaison habituels renvoient NULL et non pas vrai ou faux, quand l'une des entrées est NULL.