SQL

1. LOGIQUE TRIVALUÉ

On a 3 valeurs logique possible `TRUE, FALSE, NULL`

`NULL` correspond a une valeur **non connu**

Opératons : `AND, OR, NOT`

1.1. AND

AND	**TRUE**	**NULL**	**FALSE**
TRUE	TRUE	NULL	FALSE
NULL	NULL	NULL	FALSE
FALSE	FALSE	FALSE	FALSE

1.2. OR

OR	**TRUE**	**NULL**	**FALSE**
TRUE	TRUE	TRUE	TRUE
NULL	TRUE	NULL	NULL
FALSE	TRUE	NULL	FALSE

1.3. NOT

NOT	**-**
TRUE	FALSE
NULL	NULL
FALSE	TRUE

- Dans une requête, la condition du `WHERE` ne selectionne que les lignes qui donne une valeur `TRUE` pour la condition. (`NULL` et `FALSE` éliminés)
- Dans les `CHECK` des `CREATE TABLE` `NULL` et `TRUE` sont acceptés.
- Dans une comparaison, si une des deux valeurs est `NULL` ça retourne `NULL`

```
SELECT NULL = NULL; => NULL

SELECT 2 = NULL; => NULL
```

```
SELECT NULL IS NULL; => TRUE

SELECT 2 IS NULL; => FALSE

SELECT NULL IS NOT NULL; => FALSE

SELECT 2 IS NOT NULL; => TRUE
```

`IS, IS NOT` retourne `TRUE, FALSE` jamais `NULL`

`IS DISTINCT, FROM`

`IS NOT DISTINCT FROM` retourne `TRUE, FALSE`

Et compare deux valeurs

```
SELECT 2 IS DISTINCT FROM NULL; => TRUE

SELECT NULL IS DISTINCT FROM NULL; => FALSE
```

```
ligne_commande (*no_fact#, *id_produit#, quant, satisfaction)
```

`satisfaction` peut être `NULL`

```
SELECT *
FROM ligne_commande
WHERE satisfaction > 3 OR satisfaction <= 3;
```

Retourne toute les lignes où `stisfaction` n'est pas égala à `NULL`

Equivalent à :

```
SELECT *
FROM ligne_commande
WHERE satisfaction IS NOT NULL;
```

```
SELECT 3 in (NULL, 1, 2); => NULL

SELECT NULL in (NULL, 1, 2); => NULL

SELECT 3 in (NULL, 1, 2, 3); => TRUE

SELECT (1, NULL) in ((NULL,1) (2, 3)); => NULL

SELECT (1, NULL) in ((3,1) (2, NULL)); => FALSE
```

```
(1,NULL) = (2, NULL) <=>  1 = 2 AND NULL = NULL => FALSE
```

```
produit(*id_produit, desc_produit, prix)
magasin(*id_magasin, nom_magasin, adresse)
stock(*(id_produit#, id_magasin#), quantite)
facture(*no_facture, id_magasin#, nom_client, adresse, date)
ligne_commande(*(no_facture#, id_produit#), quant, satisfaction)
```

`adresse, satisfaction` peuvent être `NULL`

Requête d'aggrégation (suite)

1. Valeur NULL

```
SELECT AVG(satisfaction), SUM(satisfaction), COUNT(*), COUNT
(satisfaction)
FROM ligne_commande
```

satisfaction
NULL
1
3
NULL
5
NULL

AVG	SUM	COUNT	COUNT
3	9	6	3

id_prod	id_mag	quantite
100	1	10
100	2	15
101	1	3
102	2	4
102	3	5
102	4	2

```
SELECT id_prod, SUM(quantite)
FROM ___ GROUP
BY id_prod;
```

id_prod	SUM
100	25
101	3
102	10

2. Syntax générale

```
SELECT ____ FROM ___
[WHERE cond du where]
[GROUP BY attributs
 HAVING cond du having]
```

Dans le select, on ne peut mettre que des attributs qui sont dans le group by et/ ou des fonctipn d'agrégations

2.1. Autre Exemple :

Pour chause produit, (id+description) la moyenne des satisfaction:

```
SELECT id_produit, desc_produit,AVG(satisfaction)
GROM ligne_comm_ NATURAL JOIN produit AS p
GROUP BY (p.)id_produit,desc_produit;
```

marchera car `id_produit` est clef primaire de la table produit dans laquelle est `desc_prduit`

3. HAVING

Permet de mettre des condition sur les valeurs des fonctions d'aggregation.

Les produits dqui ont une moyenne de satisfaction <= 2

```
SELECT id_produit, desc_produit
FROM produit NATURAL JOIN ligne_commande
GROUP BY id_produit, desc_produit HAVING AVG(satisfaction) <= 2
```

Produit N.J.ligne_commande

id_produit		**satifaction**	***res***
100		1	AVG(1,5)
100		2	"
101		3	-AVG(3,5)-
101		4	"
102		NULL	-AVG(NULL)-
102		NULL	"
103		4	-AVG(4)-
103		NULL	"

RESULTAT

id_produit	
100	----

les chaises et tabourets dont la moyenne de satisfaction est <= 2 on affichera: id_produit, description et quantité commandée.

```
SELECT id_produit, desc_produit,SUM(quantite)
FROM produit NATURAL JOIN ligne_commande
WHERE desc_produit IN ('chaise','tabouret')
GROUP BY id_produit, desc_produit HAVING AVG(satisfaction) <= 2
```

- `WHERE` : condition sur les lignes individuelles (Avant `GROUP BY`)
- `HAVING` : condition sur fonctions d'aggregations et attributs du `GROUP BY`

Autres exemple

Pour chaque magasin qui stock au moins 100 chaises et tabourets le nombre de référence (chaise et tabouret) en stock

```
SELECT id_magasin, nom_magsin, COUNT(DISTINCT id_produit)
FROM (magasin NATURAL JOIN Stock) NATURAL JOIN produit
WHERE desc_produit IN ('chaise', 'tabouret')
GROUP BY id_magasin, nom_magsin HAVING SUM(quantite) >= 10;
```

Jointure

id_mag	----	id_prod	desc_produit	quantite
1		100	table *(1)	10
1		101	ch	20
1		102	tab	30
2		101	ch	100
3		100	table *(1)	20
4		101	ch	3
4		102	tab	4

`*(1) supprimé avec le WHERE`

id_mag	SUM	COUNT	
1	50	2	**OK SUM >= 10**
2	100	1	**OK SUM >= 10**
4	7	2	**NON**