

# 420-KHG-LG, Semaine 04- Séance2

## Requêtes imbriquées (sous requêtes)

Une sous requête est une requête avec la commande SELECT imbriquée avec les autres commandes (UPDATE, INSERT, DELETE et CREATE)

Une sous-requête peut être utilisée dans les clauses suivantes :

- La clause WHERE d'une instruction UPDATE,DELETE et SELECT
- La clause FROM de l'instruction SELECT
- La clause VALUES de l'instruction INSERT INTO
- La clause SET de l'instruction UPDATE
- L'instruction CREATE TABLE.

### ***Utilisation d'une sous-requête avec la clause WHERE***

Ce type de sous-requête permet de comparer une valeur de la clause WHERE avec le résultat retourné par une sous-requête, dans ce cas on utilise les opérateurs de comparaison suivant : =, !=, <, <=, >, >=, et IN.

Cette requête ramène les numéro d'admission (NUMAD)des étudiants dont la note en KED est plus petite que celle de l'étudiant dont le numéro d'admission est 100 pour le même cours (CODE\_COURS)

```
SELECT NUMAD FROM RESULTATS
WHERE CODE_COURS='KED' AND NOTE <
  (SELECT NOTE FROM RESULTATS
   WHERE NUMAD=100 AND CODE_COURS='KED');
```

Quelle est-la requête qui ramène les noms des étudiants et non leur numéro ?

Réponse

```
SELECT E.NUMAD, E.NOM,R.NOTE FROM RESULTATS R, ETUDIANTS E
WHERE E.NUMAD= R.NUMAD
AND CODE_COURS='KED'
AND NOTE <
(SELECT NOTE FROM RESULTATS
WHERE NUMAD=100 AND CODE_COURS='KED');
```

- ✓ On utilise l'opérateur IN lorsque la sous requête retourne plus qu'une rangée (plus qu'un enregistrement)

Exemple1 : la requête suivante ramène tous les étudiants qui n'ont pas de notes

```
SELECT NUMAD,NOM, PRENOM
FROM ETUDIANTS
WHERE NUMAD NOT IN
(SELECT NUMAD FROM RESULTATS);
```

- ✓ On utilise l'opérateur ANY pour que la comparaison se fasse pour toutes les valeurs retournée. Le résultat est vrai si au moins une des valeurs répond à la comparaison
- ✓ On utilise l'opérateur ALL pour que la comparaison se fasse pour toutes les valeurs retournée. Le résultat est vrai si toutes les valeurs répondent à la comparaison
- ✓ On utilise l'opérateur EXISTS est similaire à l'opérateur IN pour déterminer l'existence ou non de lignes. EXISTS teste si la requête intérieur retourne des valeurs.

Exemple

```
SELECT * FROM clients WHERE EXISTS (SELECT numclient FROM commande);
```

Pour la table résultat suivante, les sorties des requêtes avec ANY et ALL sont les suivantes :

CODE_COURS	NUMAD	NOTE
KED	100	78
KED	101	65
KEG	100	88
KEG	101	78
KED	109	80
KED	107	78
KEG	107	55
KED	102	90

```
SELECT NUMAD FROM RESULTATS
WHERE NOTE >ALL
(SELECT NOTE FROM RESULTATS WHERE CODE_COURS ='KEG');
```

Donne le résultat

NUMAD
102

```
SELECT NUMAD FROM RESULTATS
WHERE NOTE >ANY
(SELECT NOTE FROM RESULTATS WHERE CODE_COURS ='KEG');
```

NUMAD
102
100
109
100
101
107
101

***Utilisation d'une sous-requête avec l'instruction INSERT***

Ce type de sous-requête permet d'insérer des données dans une table à partir d'une autre table. La sous requête est utilisée à la place de la clause VALUES de la requête principale et peut retourner plusieurs résultats.

```
INSERT INTO COURS_DU_SOIR  
(SELECT * FROM COURS  
WHERE CODE_COURS ='KEG');
```

Ou pour insérer toutes les lignes de la table COURS dans COURS\_DU\_SOIR

```
INSERT INTO COURS_DU_SOIR  
(SELECT * FROM COURS);
```

### ***Utilisation d'une sous-requête pour la création de table***

Ce type de sous requête permet de créer une table à partir d'une autre table. La nouvelle table contient les valeurs de la sous-requête

#### **EXEMPLE**

La requête suivante permet de créer la table des notes du cours KED

```
CREATE TABLE NOTEKED (NUMADMISSION,NOTEKED)  
AS SELECT NUMAD,NOTE FROM RESULTATS WHERE CODE_COURS='KED';
```

OU

```
CREATE TABLE NOTEKED (NUMADMISSION PRIMARY KEY ,NOTEKED)  
AS SELECT NUMAD,NOTE FROM RESULTATS WHERE CODE_COURS='KED';
```

#### **QUESTION**

Quelle-est la requête qui permet DE créer une table NOTESKED qui va avoir comme attributs NOMETUDIANT, PRENOMETUDIANT,TITRECOURS,NOTEKED

REPONSE

```
CREATE TABLE NOTESKED (NOMETUDIANT,  
PRENOMETUDIANT,TITRECOURS,NOTEKED)AS  
SELECT NOM, PRENOM,TITRE,NOTE  
FROM ETUDIANTS E INNER JOIN RESULTAT R ON E.NUMAD=R.NUMAD  
INNER JOIN COURS CR ON CR.CODE_COURS=R.CODE_COURS WHERE  
R.CODE_COURS='KED' ;
```

Remarque : Parfois, il est utile de créer une table ayants les mêmes attributs que la table principale mais sans avoir les valeurs de celle-ci. Dans ce cas il suffit de mettre une clause WHERE impossible dans la sélection.

### ***Utilisation d'une sous-requête avec la clause SET***

Dans ce cas le résultat retourné par la sous-requête doit contenir une seule rangée.

```
UPDATE RESULTATS SET NOTE =  
(SELECT AVG(NOTE) FROM RESULTATS WHERE CODE_COURS ='KEG')  
WHERE NOTE <60 AND CODE_COURS ='KEG';
```

### ***Requêtes corrélées***

Question : comment avoir tous les étudiants qui sont dans le même programme que l'étudiant dont le nom est PATOCHE

```
SELECT * FROM ETUDIANTS  
WHERE CODEPRG = (SELECT CODEPRG FROM ETUDIANTS WHERE  
NOM='PATOCHE');
```

NUMAD	NOM	PRENOM	CODEPRG
100	PATOCHE	ALIAN	420
101	FAFAR	CHANTALE	420
109	FAVRE	NICOLAS	420
107	JACOB	YANICK	420

## Question

Comment peut-on avoir les étudiants (nom et prénoms) qui ont la même note que l'étudiant numéro 100 et dans le cours KED

## Réponse

```
SELECT * FROM ETUDIANTS
WHERE NUMAD IN
      (SELECT NUMAD FROM RESULTATS WHERE CODE_COURS ='KED' AND
NOTE =
      (SELECT NOTE FROM RESULTATS WHERE CODE_COURS='KED' AND
NUMAD =100));
```

NUMAD	NOM	PRENOM	CODEPRG
100	PATOCHE	ALIAN	420
107	JACOB	YANICK	420

## Exercices

### Exercice 1 : Utiliser les tables syemp et sydept

1. Qui a le salaire plus élevé que celui de l'employé Jones ?
2. Qui a le plus le plus petit salaire de tous les employés ?
3. Écrire la sous-requête qui permet d'afficher les salaires de tous les employés dont le salaire se situe entre le plus bas et le plus haut des salaires versés aux employés du département numéro 30.
4. Écrire la sous-requête qui permet d'afficher les salaires de tous les employés dont le salaire se situe entre le plus bas et le plus haut des salaires versés aux employés du département 'SALES';
5. Afficher tous les employés qui travaillent dans le même département que l'employé dont le nom est ALLEN
6. Afficher les noms des employés et leurs départements.

7. Afficher les noms des employés et leurs départements y compris les départements qui n'ont pas d'employées
8. Afficher les noms des employés et leurs départements y compris les employés qui n'ont pas de départements
9. Qui sont les employés qui travaillent dans un département dans la localité est à DALLAS?
10. Afficher les noms des départements qui ont à leur emploi plus qu'un commis
11. Exécuter la requête suivante

```
SELECT EMPNO, ENAME, SAL FROM SYEMP  
WHERE SAL > ALL (SELECT SAL FROM SYEMP WHERE DEPTNO = 30);
```

12. Dans la requête précédente remplacer le terme ALL par ANY, puis exécuter la requête.

Notez ce que vous remarquez :

## **Exercice 2 : Utiliser vos propres tables (étudiants, programmes ...)**

1. Qui sont les étudiants qui sont dans le même programme qu'ALAIN PATAOCHE ?
2. Afficher tous les étudiants (nom, prénom) qui n'ont pas de note.
3. Afficher le nom, le prénom et la note des étudiants qui ont une plus petite note en KHG que la moyenne des notes en KHG.
4. Quel est le nom et le prénom de l'étudiant ayant la plus haute note en KHG ?
5. Afficher le nom et le prénom des étudiants en KHG dont la note est plus petite que toutes les notes de KHG.

6. Afficher Le nom et le prénom des étudiants en KHG dont la note est plus petite qu'au moins une note de KHG
7. Combien de notes sont saisies pour le cours de 'KED'?
8. Mettre à jour le code programme de BIEN SIMON par le même code programme que SATURNE ALAIN
9. Mettre à jour le code programme de GUOIN SÉBASTIEN par le même code programme qu'ALAIN PATAOCHE.
10. Qui sont les trois meilleurs étudiants ?
11. Ecrire la requête qui permet de créer la table NotesBd qui va contenir les notes des étudiants dans le cours : Intégration des bases de données. Les colonnes de la table seront (nom, prenom et note) ;
12. Ecrire la requête qui supprime les étudiants qui ne sont dans aucun programme.(ne pas exécuter)