Semaine 03- Séance1

Retour sur la commande CREATE TABLE.

La contrainte de PRIMARY KEY: permet de définir une clé primaire sur la table. Lorsque la clé primaire est une clé primaire composée, la contrainte doit être définie au niveau de la table et non au niveau de la colonne. Les attributs faisant partie de la clé doivent être entre parenthèse. La contrainte de PRIMARY KEY assure également les contraintes de NOT NULL et UNIQUE

La contrainte CHECK :
Indique les valeurs permises qui peuvent être saisies pour la colonne (champ ou attribut) lors de l'entrée des données ou une condition à laquelle doit répondre une valeur insérée. La condition doit impliquer le nom d'au moins une colonne. Les opérateurs arithmétiques (+,*,/,-), les opérateurs de comparaisons et les opérateurs logiques sont permis.

La contrainte DEFAULT : indique la valeur par défaut que prendra l'attribut si aucune valeur n'est saisie.

La contrainte NOT NULL : indique que la valeur de la colonne ou de l'attribut est obligatoire. Si cette contrainte n'est pas précisée alors par défaut la valeur est NULL.

La contrainte UNIQUE : indique que les valeurs saisie pour les colonnes (champs ou attributs) doivent être unique Ce qui veut dire pas de Doublons.

La contrainte de FOREIGN KEY: cette contrainte indique que la valeur de l'attribut correspond à une valeur d'une clé primaire de la table spécifiée.

La clé primaire de l'autre table doit être obligatoirement crée pour que cette contrainte soit acceptée. La clé primaire de l'autre table et l'attribut défini comme clé étrangère doivent être de même type et de même longueur

On peut également préciser l'option ON DELETE CASCADE qui indique que les enregistrements (occurrences) soient détruits lorsque l'enregistrement correspondant à la clé primaire de la table référencée est supprimé. Si cette option n'est pas précisée alors aucun enregistrement ne sera supprimé de la table qui contient la clé primaire

Important :

La contrainte de clé étrangère est une contrainte sur la table. Elle ne peut être définie sur un attribut (colonne)

Exemple

```
CREATE TABLE programmes
(
codePrg VARCHAR2(3)
CONSTRAINT pk3 PRIMARY KEY,
nomProg VARCHAR2(20)
);
CREATE TABLE etudiants
(
NumAd NUMBER CONSTRAINT pketudiant PRIMARY KEY,
Nom VARCHAR2(20) NOT NULL,
Prenom VARCHAR2(20),
codePrg VARCHAR2(3),
CONSTRAINT fk1 FOREIGN KEY(codePrg) REFERENCES programme (codePrg)
);
```

Clé primaire composée·

Il arrive qu'une table ait besoin d'une clé primaire composée pour pouvoir identifier les enregistrements. Dans la plus part des cas, ces deux attributs qui composent la clé sont issus deux autres tables. La table qui contient la clé primaire composée est appelée «table de relation».

Exemple

```
CREATE TABLE Resultats
(
NumEtudiant NUMBER(10,0),
codeCours VARCHAR2(10),
Note NUMBER(5,2),
CONSTRAINT pkresultat PRIMARY KEY(NumEtudiant,codeCours)
);
```

La contrainte de clé primaire composées est une contrainte sur la table. Elle ne peut être définie sur un attribut (colonne)

Quelques fonctions SQL:

Les fonctions agissant sur les groupes

Ces fonctions sont utilisées pour traiter des groupes de rangées et d'afficher un seul résultat. Même si ce sont des fonctions de groupement, elles ne s'utilisent pas tout le temps avec la clause GROUP BY.

Les fonctions MIN et MAX: ce sont des fonctions qui s'utilisent pour afficher la valeur MIN (ou MAX) parmi l'ensemble des valeurs de la colonne indiquée.
Exemple

```
SELECT MAX (NOTE) FROM RESULTATS WHERE CODE_COURS ='KED';
```

Les fonctions AVG et SUM

AVG s'utilise pour obtenir une valeur moyenne des valeurs de la colonne indiquée
SUM s'utilise pour obtenir une valeur totale des valeurs de la colonne indiquée
Exemple

```
SELECT AVG (NOTE) FROM RESULTATS WHERE CODE_COURS ='KED';
```

Les fonctions VARIANCE et STDDEV: Pour calculer la variance et l'écart type sur les valeurs d'une colonne

La fonction COUNT: cette fonction permet de compter le nombre de lignes (rangées) qui répondent à un critère. La clause GROUP BY peut être utilisée ou non

Si une colonne est présente dans la clause SELECT alors elle doit être présente dans la clause GROUP BY

La clause GROUP BY: cette clause permet d'indiquer au système de regrouper des enregistrements selon des valeurs distincts qui existent pour les colonne spécifiées. La clause HAVING permet de mieux cibler les enregistrements spécifiés.

Exemples

```
SELECT CODEPRG, COUNT(CODEPRG)
FROM ETUDIANTS
GROUP BY CODEPRG;
```

CODEPRG	COUNT(CODEPRG)
430	2
420	4
410	3

```
SELECT CODEPRG, COUNT(CODEPRG)
FROM ETUDIANTS
GROUP BY CODEPRG
HAVING CODEPRG ='420';
```

Cette requête calcule le nombre total d'étudiants.

```
SELECT COUNT(*)
FROM ETUDIANTS;
```

Étape1 : Création des tables :

1- Créer la table Etudiants avec les colonnes suivantes :

Colonnes	Types et contratines
Numad	Number, Primary Key
Nom	Varchar2(40) not null
Prenom	Varchar2(40)
CodeP	Number(3), clé étrangère faisant référence à CodeP de la table Programmes

2- Créer la table Programmes avec les colonnes suivantes

Colonnes	Types et contratines
CodeP	Number(3), Primary Key
NomProgramme	Varchar2(40) not null

3- Créer la table Cours avec les colonnes suivantes

Colonnes	Types et contratines
CodeCours	Char(3), Primary Key
TitreCours	Varchar2(40) not null
CodeP	Number(3), clé étrangère faisant référence à CodeP de la table Programmes

4- Créer la table Resultats avec les colonnes suivantes :

Colonnes	Types et contratines
CodeCours	Char(3),
Numad	Number
Note	Number(5,2)

CodeCours et Numad forment la clé primaire composée. Elles sont également clé étrangères.

Étape 2 : Insertion des enregistrements

Insérer les enregistrements suivants dans Etudiants

	NUMAD	NOM	PRENOM	CODEP
1	10	PATAOCHE	ALAIN	420
2	11	POIRIER	JUTEUX	420
3	12	FAFAR	CHANTAL	420
4	13	SIMPSON	CHRISTIAN	410
5	14	LEFOU	DUVILLAGE	430
6	15	SATURNE	ALAIN	410
7	16	BIEN	SIMON	(null)
8	17	GOUIN	SÉBASTIEN	(null)

Insérer les enregistrements suivants dans Programmes

	CODEP	NOMPROGRAMME
1	420	INFORMATIQUE
2	410	ADMINISTRATION
3	430	SANTÉ ANIMALE
4	440	GÉNIE INDUSTRIELLE

Insérer les enregistrements suivants dans Cours

	CODECOURS	TITRECOURS	CODEP
1	KED	CONCEPTION DES BASES DE DONNÉES	420
2	KA5	DÉVELOPPEMENT D'INTERFACES	420
3	KEE	STRUCTURES DE DONNÉES	420
4	EDC	ADMINISTRATION 101	410
5	KHG	Intégration des bases de données	420

Insérer les enregistrements suivants dans Resultats.

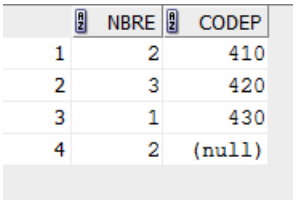
	NUMAD	CODECOURS	NOTE
1	10	KED	65
2	10	KA5	85
3	10	KEE	60
4	11	KED	85
5	11	KA5	70
6	11	KEE	75
7	12	KED	60
8	12	KA5	77
9	12	KEE	75
10	10	KHG	65
11	11	KHG	85
12	12	KHG	60

Étape 3 : Questions

1- Que se passe-t-il si vous essayez d'insérer l'enregistrement suivant dans la table Etudiants :

INSERT INTO ETUDIANTS VALUES (20,'ROY','SÉBASTIEN',520);?

2- Que se passe-t-il si vous essayez de supprimer le programme dans le code est 420 de la table Programmes?
3- Que se passe-t-il si vous essayez de Modifier le code est 420 dans la table Programmes?
4- Que se passe-t-il si vous essayer de supprimer l'étudiant dont le NUMAD est 10 de la table Etudiants?
5- Écrire les requêtes suivantes :
    a. Liste des étudiants en informatique. (le codep est 420) ;
    b. Liste des étudiants qui n'ont pas de notes
    c. Combien d'étudiants avons-nous au total ?
    d. Combien d'étudiants avons-nous dans le programme informatique (le codeP est 420) ?
    e. Combien d'étudiants avons-nous dans chaque programme ? (vous devez obtenir ceci)



	NBRE	CODEP
1	2	410
2	3	420
3	1	430
4	2	(null)

    f. Quelle est la moyenne des étudiants dans le cours de KHG ?
    g. Quelle est la moyenne des notes selon le code cours ?
    h. Quelle est la plus haute et la plus basse note en KHG?