

# T.P. XI - B. d. d. Modifications, Jointures, Agrégats

## I - Modification de bases de données

### I.1 - Création d'une table

#### Définition 1 - CREATE TABLE ...

La requête `CREATE TABLE nom (col1 domaine1, col2 domaine2)` permet de créer la table `nom` dont les attributs sont `col1` et `col2` en spécifiant leur domaine.

\* Pour créer la table `Entreprise` on pourrait utiliser :

```
CREATE TABLE Entreprise (  
    NomEntreprise TEXT,  
    Siren TEXT,  
    AdresseEntreprise TEXT,  
    CPEntreprise TEXT,  
    VilleEntreprise TEXT,  
    SecteurActivite TEXT,  
    ContactEntreprise TEXT,  
    PRIMARY KEY (Siren))
```

\* Pour créer la table `Stagiaire` on pourrait utiliser :

```
CREATE TABLE Stagiaire (  
    NirStagiaire TEXT,  
    NomStagiaire TEXT,  
    PrenomStagiaire TEXT,  
    DatenaissanceStagiaire DATE,  
    AdresseStagiaire TEXT,  
    CPStagiaire TEXT,  
    VilleStagiaire TEXT,  
    TelStagiaire TEXT,  
    SirenEntreprise TEXT,  
    PRIMARY KEY (NirStagiaire),  
    FOREIGN KEY (SirenEntreprise)  
        REFERENCES Entreprise(Siren))
```

### I.2 - Modification d'une table

#### Définition 2 - Ajout d'un enregistrement

La requête `INSERT INTO table (col1, col2, ...) VALUES (val1, val2, ...)` permet d'insérer dans `table` l'enregistrement dont l'attribut `col1` vaut `val1`, l'attribut `col2` vaut `val2`,...

\* Ajout d'un stagiaire :

```
INSERT INTO Stagiaire  
(NirStagiaire, NomStagiaire, PrenomStagiaire,  
    DatenaissanceStagiaire, AdresseStagiaire,  
    CPStagiaire, VilleStagiaire, TelStagiaire,  
    SirenEntreprise)  
VALUES  
("214053122002379", "Filine", "Agathe",  
    2014-05-31, "3_rue_du_boulot", "31500",  
    "Toulouse", "06.25.56.48.75", "571261524")
```

\* Ajout d'une entreprise :

```
INSERT INTO Entreprise  
(NomEntreprise, Siren, AdresseEntreprise,  
    CPEntreprise, VilleEntreprise,  
    SecteurActivite, ContactEntreprise)  
VALUES  
("Lycée_Saint_Sernin", "193100419",  
    "3_place_Saint_Sernin", "31000", "Toulouse",  
    "Education", "0310041b@ac-toulouse.fr")
```

#### Définition 3 - Suppression d'un enregistrement

La requête `DELETE FROM table WHERE conditions` permet de supprimer tous les enregistrements de table qui satisfont

## conditions

- \* Suite à une erreur de saisie, pour supprimer toutes les formations dont la date de début est au 1<sup>er</sup> janvier 2023 :

```
DELETE FROM Formation
WHERE DateDebutFormation = 2023-01-01
```

- \* Suite à des avaries dans les locaux, pour supprimer toutes les formations ayant lieu à Agen :

```
DELETE FROM Formation
WHERE LieuFormation = "Agen"
```

- \* Suite à sa démission, pour supprimer toutes les formations assurées par le formateur dont le numéro de sécurité sociale est 1763104325012 :

```
DELETE FROM Anime
WHERE NirFormateur = "1763104325012"
```

## Définition 4 - Modification des enregistrements

La requête `UPDATE table SET col1 = nvelle_valeur WHERE conditions` permet de modifier l'attribut `col1` en y attribuant la valeur `nvelle_valeur` pour tous les enregistrements de `table` qui satisfont les conditions.

- \* La nouvelle responsable des formations Sécurité sera Marie :

```
UPDATE Formation
SET ResponsableFormation = "Marie"
WHERE DesignationFormation = "Sécurité"
```

- \* L'entreprise de numéro Siren "571261524" déménage au "12 rue des congés" à "Castres" avec le code postal "81100"

```
UPDATE Entreprise
SET AdresseEntreprise = "12_rue_des_congés",
    CPEntreprise = "81100",
    VilleEntreprise = "Castres"
WHERE Siren = "571261524"
```

## II - Jointures

Dans la base Formation, la table **Inscrit** permet de connaître le numéro NIR des stagiaires inscrits à une formation mais elle ne permet pas de les contacter. En effet, le contact des stagiaires se trouve dans la table **Stagiaire**. Il faut donc coupler ces deux tables pour obtenir cette information. Dans les bases de données, cette opération s'appelle une *jointure*.

## Définition 5 - ... JOIN ... ON ...

La requête `SELECT * FROM table1 INNER JOIN table2` permet d'afficher le contenu des tables `table1` et `table2` dans lesquelles une jointure naturelle a été faite sur leur attribut commun.

- \* Pour obtenir les noms et numéros de téléphone de stagiaires inscrits à une formation :

```
SELECT NomStagiaire, TelStagiaire
FROM Stagiaire
INNER JOIN Inscrit
```

- \* Pour obtenir le nom des stagiaires ainsi que le nom de l'entreprise à laquelle ils sont rattachés :

```
SELECT NomStagiaire, NomEntreprise
FROM Stagiaire
INNER JOIN Entreprise
```

## III - Fonctions d'agrégation

## Définition 6 - Agrégation

Les fonctions d'agrégation permettent d'obtenir des statistiques sur un ensemble d'enregistrements. On pourra retenir les fonctions MIN, MAX, SUM, AVG (moyenne se dit *average* en anglais), COUNT(\*).

- \* Pour compter le nombre d'entreprises présentes dans la base de donnée :

```
SELECT COUNT(*)  
FROM Entreprise
```

\* Pour obtenir le coût de la formation la moins chère :

```
SELECT MIN(CoutFormation)  
FROM Formation
```

\* Pour obtenir le coût de la formation la plus chère :

```
SELECT MAX(CoutFormation)  
FROM Formation
```

\* Pour obtenir le coût moyen d'une formation :

```
SELECT AVG(CoutFormation)  
FROM Formation
```