

SYSTÈME DE GESTION DE BASES DE DONNÉES

Langage de définition de données (LDD)

Structured Query Language

1 - langage d'interrogation de données (LID): SELECT

SELECT	NomColonne1 NomColonnep2	Colonne à projeter ou à calculer ou fonction d'agrégat
FROM	TABLE1, TABLE2,...	Tables utiles à la requête
WHERE	AND/OR	Jointures et restrictions
GROUP BY	NomColonne, ...	Regroupement de résultat d'opérations d'agrégat
HAVING	Condition	Restriction sur l'affichage des résultats d'opérations d'agrégat
ORDER BY	Colonne [ASC]/DESC, ...	Critères de tri

2 - Langage de définition de données (LDD) :

CREATE, ALTER, DROP

Les commandes de description de données

- Principaux objets d'une base de données
- Créer une base de données
- Créer une table
- Types de données des colonnes
- Modifier la définition d'une table
- Supprimer une table

Principaux objets d'une base de données

Objet	Description
Database	Une base de données relationnelles est un ensemble de tables organisées
Table	Unité de stockage élémentaire, composée de lignes et de colonnes
Vue	Représente de manière logique des sous-groupes de données (requête)
Index	Améliore les performances de certaines requêtes <i>comme pour la recherche dans un livre</i>

LDD : Création et suppression de bases de données

Création de bases de données :

```
CREATE DATABASE Nom_BD;
```

Exemple : création d'une BD nommée "dbeleve"

```
CREATE DATABASE dbeleve;
```

Suppression d'une base de données

```
DROP DATABASE Nom_BD;
```

Exemple : suppression de la BD nommée "dbeleve"

```
DROP DATABASE dbeleve;
```

LDD : Création et suppression de bases de données

EXEMPLE Création de bases de données :

```
CREATE DATABASE Nom_BD
```

```
CHARSET utf8;
```

Un jeu de caractères (character set ou charset) est un ensemble de lettres, signes de ponctuations et autres symboles auxquels on a associé un numéro de code. Par exemple, le jeu «ASCII » (American Standard Code for Information Interchange) donne le numéro 65 au A majuscule, le 44 à la virgule et le 13 au retour à la ligne.

LDD : Créer une table

Clause CREATE :

création d'une table dans une base de données :

```
CREATE TABLE Nom_table  
(Nom_colonne1 Type,  
Nom_colonne2 Type, ...);
```

LDD : Les types de données

Pour chaque colonne que l'on crée, il faut préciser le type de données que le champ va contenir. Celui-ci peut être un des types suivants :

Type de donnée	Syntaxe	Description
Type alphanumérique	CHAR(n)	Chaîne de caractères de longueur fixe n (n<16383)
Type alphanumérique	VARCHAR(n)	Chaîne de caractères de n caractères maximum (n<16383)
Type numérique	DECIMAL (n,[d])	Nombre de n chiffres [optionnellement d après la virgule] (autre SGBD NUMBER[(n[,d])])
Type numérique	SMALLINT (TY	Entier signé de 16 bits (-32768 à 32757)
Type numérique	INTEGER (INT)	Entier signé de 32 bits (-2E31 à 2E31-1)
Type numérique	FLOAT	Nombre à virgule flottante
Type date	DATE	Date sous la forme 16/07/16
Type horaire	TIME	Heure sous la forme 12:54:24.85
Type horaire	TIMESTAMP	Date et Heure
Et d'autres

LDD : Créer une table

Clause CREATE : création d'une table dans une base de données :

```
CREATE TABLE Nom_table  
(Nom_colonne1 Type,  
Nom_colonne2 Type, ...);
```

```
CREATE TABLE devoir(  
    codedevoir SMALLINT,  
    titredevoir VARCHAR(50),  
    datedevoir date,  
    PRIMARY KEY (codedevoir)  
) CHARSET utf8;
```

Exemple : Création d'une table nommée "prof", avec deux colonnes une pour l'identifiant de type entier comme clé primaire et une autre pour le nom de 50 caractères maximums :

```
CREATE TABLE prof(  
    codeprof SMALLINT NOT NULL,  
    nomprof VARCHAR(50) NOT NULL,  
    PRIMARY KEY (codeprof) )  
engine = InnoDB CHARSET utf8;
```

Supprimer une table

- Pour supprimer une table, tapez la commande :
- `DROP TABLE <nom de la table>`

Exemple : suppression de la table des devoirs:

```
DROP TABLE devoir
```

Exo05 Description des données

En **SQL** dans la base de données bd_test créer la table tbl_employe:

Nom	Type de données	Taille	Null	VALLEUR PAR DEFAULT
id	Numérique	7	Non	
Nom	Alphanumérique	50	Oui	
Prenom	Alphanumérique	40	Oui	
Service	Numérique	7	Oui	1

Les contraintes d'intégrité

Une contrainte d'intégrité permettent de s'assurer que les données saisies dans la base sont conformes aux données attendues. Ces contraintes doivent être exprimées dès la création de la table.

On distingue trois types de contraintes d'intégrité :

Contrainte de domaine : contrôle le format de la donnée saisie par rapport au domaine de valeur prévu pour l'attribut.

Exemple : le champ datedevvoir de la Table DEVOIR doit être une date au format JJ/MM/AAAA

Contraintes de relation (ou de table): concerne la clé primaire qui doit être unique.

Exemple : le champ codedevvoir est obligatoire et unique.

Contraintes d'intégrité référentielle : limite l'ajout ou la suppression des enregistrements liés par des clés étrangères.

Exemple : On ne peut supprimer la catégorie la matière M1 car il existe des notes de cette matière.

Contraintes

- NOT NULL
- PRIMARY KEY
- FOREIGN KEY
- UNIQUE (*slam1*)
- CHECK (*slam1*)

NOT NULL

- La contrainte NOT NULL interdit la présence de valeurs NULL dans la colonne à laquelle elle s'applique
- Par défaut, les colonnes peuvent contenir des valeurs NULL
- **Exemple** : créer la table groupe le nom ne doit pas être vide
- Codeg est une chaîne de 6 caractères et nomg une chaîne de 30 caractères .

```
CREATE TABLE groupe  
(codeg int,  
nomg varchar(30) NOT NULL)
```

PRIMARY KEY

- Une contrainte PRIMARY KEY crée une clé primaire pour la table
- La contrainte PRIMARY KEY est une colonne ou un ensemble de colonnes qui identifie de manière unique chaque ligne d'une table
- garantit qu'aucune colonne faisant partie de la clé primaire ne contient de valeur NULL
- Exemple : modifier la requête avec codeg comme clé primaire
CREATE TABLE groupe
(codeg varchar(6),
nomg varchar(30) **NOT NULL**,
PRIMARY KEY(codeg))

PRIMARY KEY

- En intégrant une contrainte.
- Exemple : modifier la requête avec codeg comme clé primaire
CREATE TABLE groupe
(codeg char(6),
nomg varchar(30) **NOT NULL**,
CONSTRAINT pk_groupe PRIMARY KEY(codeg))

FOREIGN KEY

- La contrainte FOREIGN KEY, ou contrainte d'intégrité référentielle, désigne une colonne ou une combinaison de colonnes comme étant une clé étrangère
- établit une relation avec une clé primaire
- Exemple :

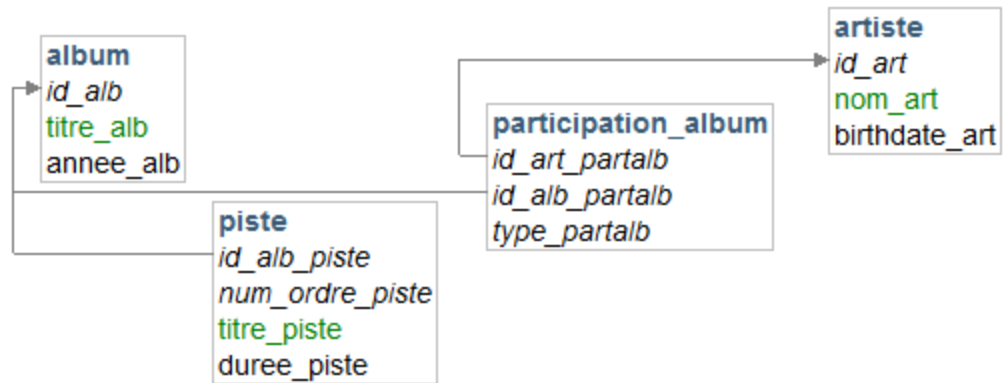
```
CREATE TABLE eleve
(
  cin varchar(12),
  nom varchar(30),
  codeg char(6),
  CONSTRAINT pk_élève PRIMARY KEY (cin),
  CONSTRAINT fk_groupe FOREIGN KEY (codeg)
  REFERENCES groupe(codeg)
);
```

Application

Les **types de données** sont généralement choisis parmi les suivants :

- **numériques entiers** : **TINYINT**[(length)], **SMALLINT**[(length)], **MEDIUMINT**[(length)], **INT**[(length)], **INTEGER**[(length)], **BIGINT**[(length)]
- **numériques réels** : **REAL**[(length,decimals)], **DOUBLE**[(length,decimals)], **FLOAT**[(length,decimals)], **DECIMAL**(length,decimals), **NUMERIC**(length,decimals)
- **dates, heures** : **DATE**, **TIME**, **TIMESTAMP**, **DATETIME**
- **chaînes de caractères** : **CHAR**(length), **VARCHAR**(length), **TINYTEXT**, **TEXT**, **MEDIUMTEXT**, **LONGTEXT**
- **binaire** (images, pdf...) : **TINYBLOB**, **BLOB**, **MEDIUMBLOB**, **LOB**
- **une valeur parmi un ensemble** : **ENUM**(value1,value2,value3,...)
- **plusieurs valeurs parmi un ensemble** : **SET**(value1,value2,value3,...)

Application



Les clés primaires

Définir une clé primaire simple (composée d'un seul champ) avec la clause **PRIMARY KEY placée juste après le champ concerné :**

Le schéma relationnel ci-dessus permet de représenter des albums de musique et leurs pistes. On peut associer un artiste soit à l'album entier, soit à une piste précise. Le type de participation est par exemple "Chanteur", "Compositeur", "Bassiste", etc.

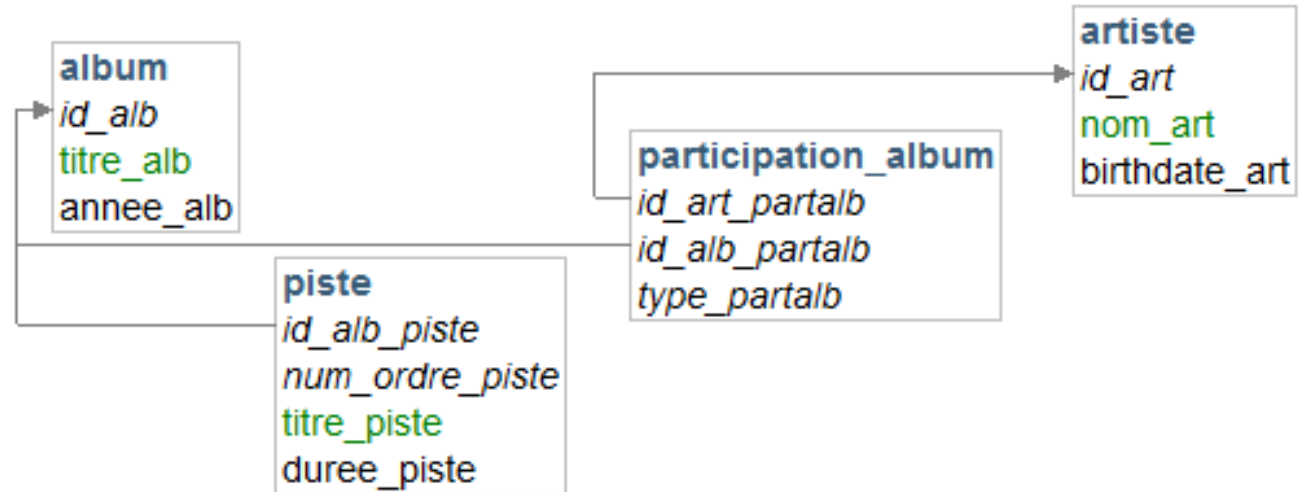
1- D'après ce schéma relationnel, donnez l'instruction SQL **CREATE TABLE qui permet de créer la table **Album**. Le champ **titre_alb** ne peut pas être **NULL** : un album a forcément un titre**

2 - Donnez l'instruction SQL **CREATE TABLE qui permet de créer la table **Artiste**. Le champ **nom_art** ne peut pas être **NULL** : un artiste a forcément un nom.**

Définir une clé primaire composée de plusieurs champs

3 - Donnez l'instruction SQL **CREATE TABLE qui permet de créer la table **Piste**. A ce stade, ne tenez pas compte des contraintes de clés étrangères. Le champ **titre_piste** ne peut pas être **NULL** : une piste a forcément un titre.**

Application



Les clés étrangères

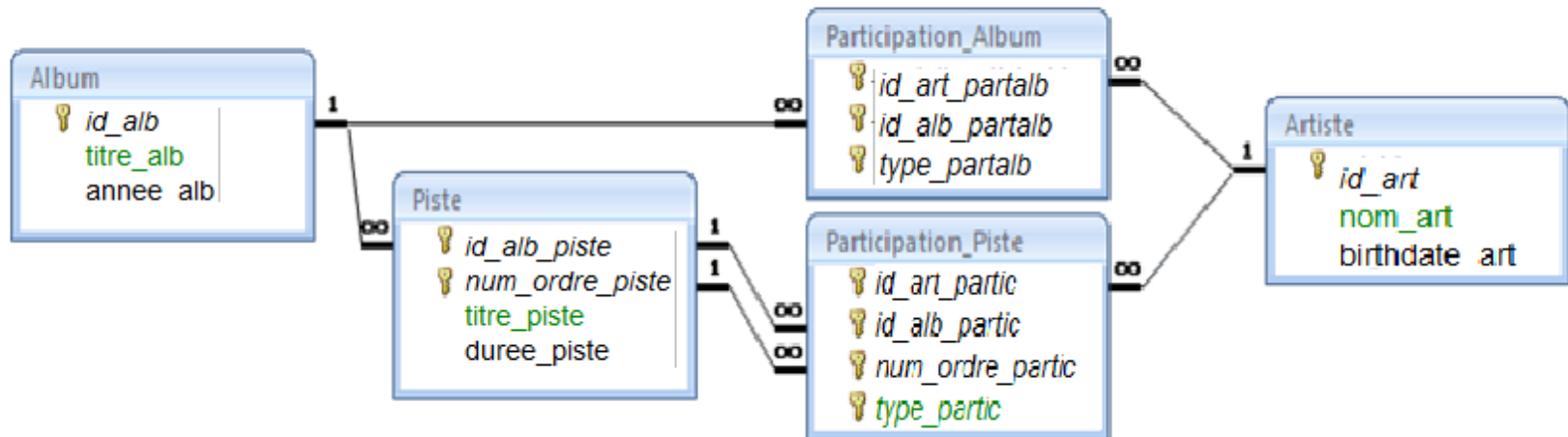
Définir une clé étrangère simple

Contrairement à la clé primaire qui peut être définie soit directement avec le champ, soit à la fin (après tous les champs), la clé étrangère est **forcément définie à la fin**

4 - Donnez l'instruction SQL **CREATE TABLE** qui permet de créer la table **Participation_Album**

5 - Donnez l'instruction SQL **CREATE TABLE** qui permet de créer la table **Piste**, en intégrant cette fois-ci les contraintes liées aux clés étrangères

Application

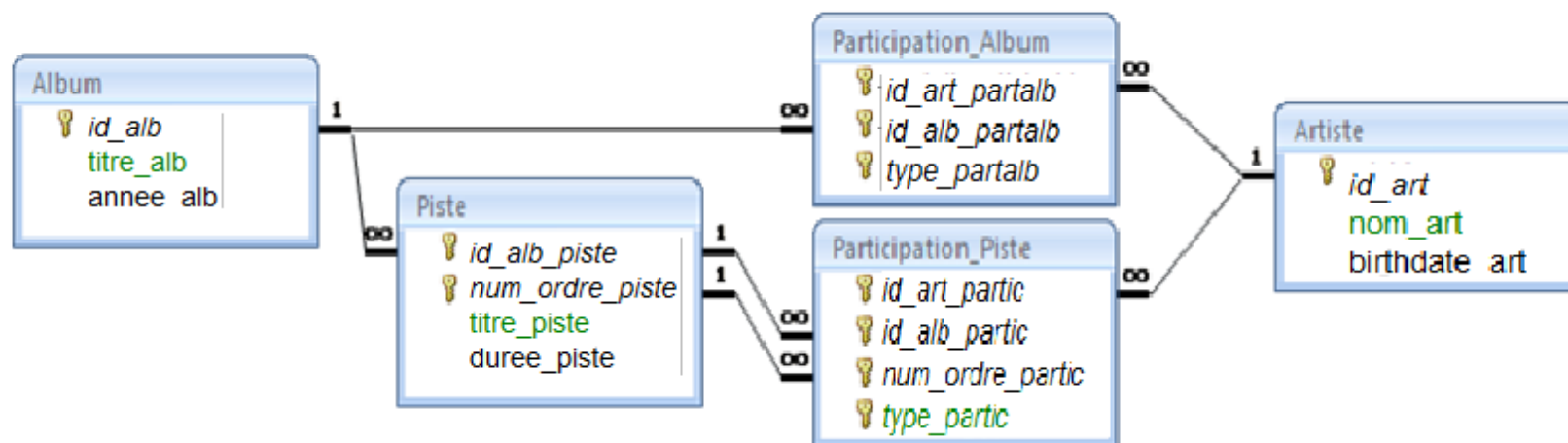


Définir une clé étrangère multiple

Lorsqu'on a une clé primaire composée (c'est à dire que **plusieurs champs** forment la clé primaire), **alors les clés étrangères pointant vers cette clé primaire seront composées**.

Puisque la clé étrangère est définie à la fin, la syntaxe est exactement la même lorsque la clé étrangère est composée.

6 - Donnez l'instruction SQL **CREATE TABLE** qui permet de créer la table **Participation_Piste**.



SQL

Définition de données

Manipulation de la BD

ALTER TABLE *table*

ADD COLUMN *champ* *type*[(*taille*)] [**NOT NULL**]

Ajouter une colonne

MODIFY COLUMN *champ* *type*[(*size*)]

Modifier une colonne

ADD CONSTRAINT

Ajouter une contrainte

- Clé primaire
- Clé étrangère
- etc.

...

DROP COLUMN *nom_du_champ*

DROP CONSTRAINT *nom_de_la-contrainte*

La suppression d'un champ n'est possible que si le champ ne fait pas l'objet d'une contrainte d'intégrité



Ajout de contrainte à une table

```
ALTER TABLE nom_table ADD PRIMARY KEY (nom_col1,...);
```

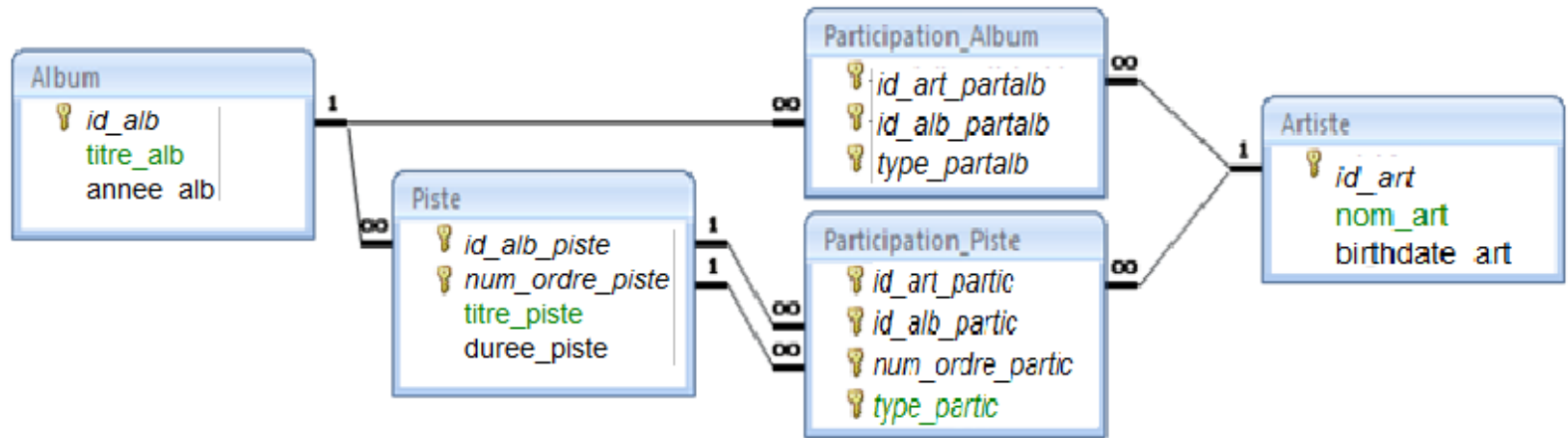
```
ALTER TABLE nom_table DROP PRIMARY KEY;
```

```
ALTER TABLE nom_table
```

```
    ADD CONSTRAINT nom_contrainte FOREIGN KEY (attr1) REFERENCES table2(attr2);
```

```
ALTER TABLE nom_table DROP FOREIGN KEY nom_contrainte;
```

Application



5 - Donnez l'instruction SQL **ALTER TABLE** qui permet d'intégrer la table **Piste** les contraintes liées aux clés étrangères.