BTS
BREVET DE TECHNICIEN SUPÉRIEUR
LYCÉE QUALIFIANT MOHAMED V
ESSAOUIRA

Modélisation de données et bases de données

Le langage SQL

Fatiha BENDAIDA

Introduction

Système de gestion de base de données (SGBD)

- **Définition (Base de données):** Une collection partagée de données en relation logique et une description des données, conçues pour satisfaire les besoins d'information d'une organisation.
- **Définition (Système de gestion de base de données)**
- Le logiciel qui permet à des utilisateurs de définir, créer, mettre à jour une base de données et d'en contrôler l'accès
  - langage de définition de données (LDD)
  - langage de manipulation de données (LMD)

Spécificités d’un SGBD

- Très grande quantité de données à gérer
- Besoin d’interroger, mettre à jour souvent, rapidement et
- efficacement ces données
- Contrôler la redondance d’information
- Partage des données / Accès concurrents
- Gérer les autorisation d’accès / Sécurit é des données
- Offrir des interfaces d’accès multiples
- Vérifier les contraintes d’intégrité
- Assurer la reprise après panne

Définitions

- **Relation:** Une table avec des colonnes et des lignes
- **Attribut:** Une colonne nommée de la relation
- **Tuple:** Une ligne dans une relation
- **Clé primaire:** La clé candidate choisie pour identifier de façon unique les tuples au sein de la relation
- **Clé étrangère:**Un ensemble d'attributs d'une relation qui correspond à une clé candidate d'une relation

Mysql

- MySQL derive directement de SQL (Structured Query Language)
- L’outil phpMyAdmin est développé en PHP et offre une interface pour l’administration des base de données phpMyAdmin est t ´ el échangeable ici : http://phpmyadmin.sourceforge.net
- cet outil permet de :
- créer de nouvelles bases
  - créer/modifier/supprimer des tables
  - afficher/ajouter/modifier/supprimer des tuples dans des tables
  - effectuer des sauvegardes de la structure et/ou des donnés effectuer des requˆetes
  - gérer les privilèges des utilisateurs

LDD

Langage de Définition de Données

- Le LDD permet de **définir et modifier la structure** d'une base de données. Il agit sur les objets (tables, index, vues, etc.) plutôt que sur les données elles-mêmes.

Créer une base de données

- Syntaxe:

```
CREATE DATABASE databasename;
```

- Exemple:

```
CREATE DATABASE testDB;
```

Supprimer une base de données

- Syntaxe:

```
DROP DATABASE databasename;
```

- Exemple:

```
DROP DATABASE testDB;
```

Créer une table

- Syntaxe:

```
CREATE TABLE table_name (
    column1 datatype,
    column2 datatype,
    column3 datatype,
   ....
);
```

- Exemple:

```
CREATE TABLE Persons (
    PersonID int,
    LastName varchar(255),
    FirstName varchar(255),
    Address varchar(255),
    City varchar(255)
);
```

Supprimer une table

- Syntaxe:

```
DROP TABLE table_name;
```

- Exemple:

```
DROP TABLE Shippers;
```

Vider une table

- Syntaxe:

```
TRUNCATE TABLE table_name;
```

- Exemple:

```
TRUNCATE TABLE costmers;
```

Modifier la structure d’une table
1-Ajouter une colonne

- Syntaxe:

```
ALTER TABLE table_name
ADD column_name datatype;
```

- Exemple:

```
ALTER TABLE Customers
ADD Email varchar(255);
```

Modifier la structure d’une table
2-Supprimer une colonne

- Syntaxe:

```
ALTER TABLE table_name
DROP COLUMN column_name;
```

- Exemple:

```
ALTER TABLE Customers
DROP COLUMN Email;
```

Modifier la structure d’une table
3-Renommer une colonne

- Syntaxe:

```
ALTER TABLE table_name
RENAME COLUMN old_name to new_name;
```

Modifier la structure d’une table
4-Modifier une colonne

- Syntaxe:

```
ALTER TABLE table_name
ALTER COLUMN column_name datatype;
```

Création des contraintes

- Syntaxe:

```
CREATE TABLE table_name (
    column1 datatype constraint,
    column2 datatype constraint,
    column3 datatype constraint,
    ....
);
```

- Exemple de contraintes:
  - NOT NULL - Garantit qu'une colonne ne peut pas contenir de valeur NULL (vide)
  - UNIQUE - Garantit que toutes les valeurs d'une colonne sont différentes (pas de doublons)
  - PRIMARY KEY - Combine NOT NULL et UNIQUE. Identifie de manière unique chaque ligne d'une table.
  - FOREIGN KEY - Maintient l'intégrité des liens entre tables en empêchant les actions qui les briseraient.
  - CHECK - Vérifie que les valeurs d'une colonne respectent une condition spécifique.
  - DEFAULT - Définit une valeur par défaut si aucune valeur n'est spécifiée pour la colonne.

Création des contraintes
1-La contrainte NOT NULL

- Exemple:

```
CREATE TABLE Persons (
    ID int NOT NULL,
    LastName varchar(255) NOT NULL,
    FirstName varchar(255) NOT NULL,
    Age int
);
```

Création des contraintes
2-La contrainte UNIQUE

- Exemple:

```
CREATE TABLE Persons (
    ID int NOT NULL UNIQUE,
    LastName varchar(255) NOT NULL,
    FirstName varchar(255),
    Age int
);
```

Création des contraintes
3-La contrainte PRIMARY KEY

- Exemple:

```
CREATE TABLE Persons (
    ID int NOT NULL PRIMARY KEY,
    LastName varchar(255) NOT NULL,
    FirstName varchar(255),
    Age int
);
```

Création des contraintes
4-La contrainte FOREIGN KEY

- Exemple:

```
CREATE TABLE Orders (
    OrderID int NOT NULL PRIMARY KEY,
    OrderNumber int NOT NULL,
    PersonID int,

    FOREIGN KEY (PersonID) REFERENCES Persons(PersonID)

);
```

Création des contraintes
5-La contrainte CHECK

- Exemple:

```
CREATE TABLE Persons (
    ID int NOT NULL,
    LastName varchar(255) NOT NULL,
    FirstName varchar(255),
    Age int,
    CHECK (Age>=18)
);
```

Création des contraintes
6-La contrainte DEFAULT

- Exemple:

```
CREATE TABLE Persons (
    ID int NOT NULL,
    LastName varchar(255) NOT NULL,
    FirstName varchar(255),
    Age int,
    City varchar(255) DEFAULT 'Sandnes'
);
```

Incrémentation automatique

- L'**auto-incrément** permet de générer automatiquement un numéro unique lorsqu'un nouvel enregistrement est inséré dans une table.
- Exemple:

```
CREATE TABLE Persons (
    Personid int NOT NULL AUTO_INCREMENT,
    LastName varchar(255) NOT NULL,
    FirstName varchar(255),
    Age int,
    PRIMARY KEY (Personid)
);
```

Le type DATE

- **DATE** - format YYYY-MM-DD
- **DATETIME** - format: YYYY-MM-DD HH:MI:SS
- **TIMESTAMP** - format: YYYY-MM-DD HH:MI:SS
- **YEAR** - format YYYY or YY