Persistance des données

Chapitre SQL - DDL

He2b-ESI (CUV - NVS - SRE - ARO)

Année académique 2021 / 2022

Plan du chapitre

- Schéma conceptuel
  - CREATE
  - Contraintes d'intégrité
    - PRIMARY KEY
    - UNIQUE
    - CHECK
    - FOREIGN KEY
  - DROP
  - ALTER
- Schémas externes
- Schéma interne

DDL - Data Definition Language

Trois ensembles d'instructions :

- CREATE
- ALTER
- DROP

Agissant sur :

- Schéma conceptuel: Tables
- Schémas externes: Vues
- Schéma interne: Index, Cluster, ...
- ... Synonym, Sequence, User, ...

DDL - Schéma conceptuel - CREATE

Définition minimale :

```
CREATE TABLE Departement (
    dptNo   CHAR(3),
    dptLib  VARCHAR(20),
    dptMgr  CHAR(3),
    dptAdm  CHAR(3)
)
```

DDL - Schéma conceptuel

- **Chaînes de caractères**
  - Longueur fixe : CHARACTER(n) or CHAR(n)
  - Longueur variable : CHARACTER VARYING(n) or VARCHAR(n)
- **Chaînes de bits**
  - Longueur fixe : BIT(n) or CHAR(n)
  - Longueur variable : BIT VARYING(n) or VARCHAR(n)
- **Nombres**
  - Virgule fixe : INTEGER or INT, SMALLINT, BIGINT, NUMERIC(precision[,scale]) or DECIMAL(precision[,scale])
  - Virgule flottante : REAL, FLOAT(n), DOUBLE PRECISION
- **Heures et dates**
  - DATE : 2016-05-03
  - TIME : 15:54:22 Généralement par pas de 100 nanosecondes
  - TIMESTAMP : 2016-05-03 15:51:36
- **Autres types parfois disponibles**
  - CLOB, BLOB, BOOLEAN, ...

DDL - Schéma conceptuel - CREATE

- Caractère obligatoire d'une colonne : [[NOT] NULL]

```
CREATE TABLE Departement (
    dptNo   CHAR(3)      NOT NULL,
    dptLib  VARCHAR(20) NOT NULL,
    dptMgr  CHAR(3)      NOT NULL,
    dptAdm  CHAR(3)      NULL -- NULL peut être omis
)
```

- Valeur par défaut : DEFAULT value

```
CREATE TABLE Departement (
    dptNo   CHAR(3)      NOT NULL,
    dptLib  VARCHAR(20) DEFAULT 'valDef' NOT NULL,
    dptMgr  CHAR(3)      NOT NULL,
    dptAdm  CHAR(3)      DEFAULT 'D21')
```

Attention à l'ordre : DEFAULT puis [NOT] NULL.

DDL - Schéma conceptuel - Contraintes d'intégrités

- Assertions concernant des attributs que chacun des tuples d'une table devra respecter à « tout moment »
- Permettent d'exprimer au SGBD certaines contraintes d'intégrité relevées dans le monde réel
- Quatre types de déclaration de CI (explicites):
  - Clé primaire: **PRIMARY KEY**
  - Unicité: **UNIQUE**
  - Validation de condition: **CHECK**
  - Clé étrangère: **FOREIGN KEY**

DDL - Schéma conceptuel - Contraintes d'intégrités

Une contrainte se déclarera lors de la création de la table ou lors de sa modification

```
[CONSTRAINT nomContrainte] expression de la contrainte
```

- Si la clause CONSTRAINT n'est pas fournie, le système assignera lui-même un nom à la contrainte. **A éviter !!**
- La déclaration de la contrainte peut se faire au niveau d'un attribut ou au niveau de la table mais elle ne peut être faite au niveau de l'attribut que si celui-ci est le seul sur lequel porte la contrainte (clé primaire sur un attribut, unicité ne concernant qu'un attribut, . . .)

```
CREATE TABLE Table (
    attribut1   TYPE    CONSTRAINT ContSurUnAttr expr1,
    attribut2 ... ,
    CONSTRAINT ContSurTable expr2
)
```

DDL - Schéma conceptuel - Identifiant primaire

```
[CONSTRAINT nomContrainte] PRIMARY KEY [(les att. de la PK)]
```

- Au niveau de la colonne :

```
CREATE TABLE Departement (
  dptNo   CHAR(3) NOT NULL CONSTRAINT dptPK PRIMARY KEY,
  dptLib  VARCHAR(20) NOT NULL,
  dptMgr  CHAR(3) NOT NULL,
  dptAdm  CHAR(3) )
```

- Au niveau de la table:

```
CREATE TABLE Departement (
  dptNo   CHAR(3) NOT NULL,
  dptLib  VARCHAR(20) NOT NULL,
  dptMgr  CHAR(3) NOT NULL,
  dptAdm  CHAR(3),
  CONSTRAINT dptPK PRIMARY KEY(dptNo) )
```

DDL - Schéma conceptuel - Identifiant primaire

```
[CONSTRAINT nomContrainte] PRIMARY KEY [(les att. de la PK)]
```

- Au niveau de la table:

```
CREATE TABLE Detail (
    ncom    CHAR(12) NOT NULL,
    npro    CHAR(15) NOT NULL,
    qcom    NUMBER(8,0),
    CONSTRAINT detailPK PRIMARY KEY (ncom, npro)
)
```

- Au niveau de la colonne : **Pas possible !!!**

DDL - Schéma conceptuel - Identifiant primaire

```
[CONSTRAINT nomContrainte] PRIMARY KEY [(les att. de la PK)]
```

- Les attributs constituant une clé primaire sont obligatoires et doivent donc être déclarés avec NOT NULL mais certains SGBD assument le NOT NULL pour les attributs constitutifs de la PK.
- Par souci de compatibilité, prenez l'habitude de spécifier NOT NULL pour ces attributs

DDL - Schéma conceptuel - Unicité

```
[CONSTRAINT nomContrainte] UNIQUE [(les att. de la UK)]
```

- Au niveau de la colonne :

```
CREATE TABLE Departement (
  dptNo   CHAR(3) NOT NULL CONSTRAINT dptPK PRIMARY KEY,
  dptLib  VARCHAR(20) NOT NULL CONSTRAINT dptLibUK UNIQUE,
  dptMgr  CHAR(3) NOT NULL,
  dptAdm  CHAR(3)
)
```

- Au niveau de la table:

```
CREATE TABLE Departement (
  dptNo   CHAR(3) NOT NULL CONSTRAINT dptPK PRIMARY KEY,
  dptLib  VARCHAR(20) NOT NULL,
  dptMgr  CHAR(3) NOT NULL,
  dptAdm  CHAR(3),
  CONSTRAINT dptLibUK UNIQUE(dptLib)
)
```

DDL - Schéma conceptuel - Unicité

```
[CONSTRAINT nomContrainte] UNIQUE [(les att. de la UK)]
```

- Au niveau de la table :

```
CREATE TABLE Participation (
  courseId  INT NOT NULL,
  coureurId INT NOT NULL,
  dossard   VARCHAR(20) NOT NULL,
  CONSTRAINT participationPK
        PRIMARY KEY(courseId, coureurId),
  CONSTRAINT dossardParCourseUK
        UNIQUE(courseId, dossard)
)
```

DDL - Schéma conceptuel - Unicité

```
[CONSTRAINT nomContrainte] UNIQUE [(les att. de la UK)]
```

- L'unicité sur des attributs facultatifs peut être gérée très différemment d'un SGBD à l'autre
  - refus
  - acceptation d'une seule valeur NULL
  - non prise en compte des valeurs NULL, des clés dont un composant au moins est NULL

DDL - Schéma conceptuel - Validation de condition

```
[CONSTRAINT nomContrainte] CHECK ( condition )
```

```
CREATE TABLE Employe (
   empNo           CHAR(3) NOT NULL,
   empNom          VARCHAR(50) NOT NULL,
   empDpt          CHAR(3) NOT NULL,
   empSal          NUMERIC(10,2) NOT NULL,
   empSexe         CHAR NOT NULL
      CONSTRAINT empSexeCK CHECK (empSexe IN ('M', 'F')),
   empDateEngage     DATE NOT NULL,
   empDateNaiss      DATE NOT NULL,
   CONSTRAINT empDateNaissCK
      CHECK (empDateEngage > empDateNaiss)
)
```

DDL - Schéma conceptuel - Clé étrangère

```
[CONSTRAINT nomContrainte]
    [FOREIGN KEY (les att. de la FK)]
    REFERENCES nomTableCible [(les att. clé référencée)]
```

- Les attributs de la clé source (la clé étrangère) doivent être compatibles 2 à 2 avec les attributs de la clé référencée (clé primaire ou clé unique de la table cible)
- Il est à remarquer que cette contrainte agit sur les deux tables :
  - la clé étrangère ne peut prendre ses valeurs que dans celles de la clé référencée
  - à priori, un tuple de la table cible ne peut pas être supprimé si il est référencé

DDL - Schéma conceptuel - Clé étrangère

```
[CONSTRAINT nomContrainte]
    [FOREIGN KEY (les att. de la FK)]
    REFERENCES nomTableCible [(les att. clé référencée)]
```

```
CREATE TABLE Employe (
      empNo   CHAR(3) NOT NULL PRIMARY KEY,
     ...
)

CREATE TABLE Departement (
     ...,
     dptMgr CHAR(3) NOT NULL
         CONSTRAINT dptMgrFK REFERENCES Employe (empNo),
     ...
)
```

DDL - Schéma conceptuel - Clé étrangère

```
[CONSTRAINT nomContrainte]
    [FOREIGN KEY (les att. de la FK)]
    REFERENCES nomTableCible [(les att. clé référencée)]
```

```
CREATE TABLE Departement (
  dptNo   CHAR(3) NOT NULL CONSTRAINT dptPK PRIMARY KEY,
  dptLib  VARCHAR(20) NOT NULL ,
  dptMgr  CHAR(3) NOT NULL,
  dptAdm  CHAR(3),
  CONSTRAINT dptLibUK UNIQUE(dptLib)
  CONSTRAINT dptAdmFK FOREIGN KEY(dptAdm)
        REFERENCES Departement (dptNo)
)
```

DDL - Schéma conceptuel - Clé étrangère

Modification de comportement en cas de suppression d'un tuple reprenant une clé référencée

```
[CONSTRAINT nomContrainte]
    [FOREIGN KEY (les att. de la FK)]
    REFERENCES nomTableCible [(les att. Clé référencée)]
    [ ON DELETE {RESTRICT | SET NULL | CASCADE} ]
```

- **RESTRICT** est l'option par défaut qui correspond au refus de suppression
- **SET NULL** demande, en cas de suppression du tuple référencé, de remplacer la valeur de la FK par NULL
- **CASCADE** demande en cas de suppression du tuple référencé de supprimer les tuples qui y font référence

DDL - Schéma conceptuel - Clé étrangère

1.
```
CREATE TABLE Departement (
      ..., dptMgr CHAR(3) NOT NULL CONSTRAINT ManagerFk
      REFERENCES Employe (empNo) ON DELETE RESTRICT,
      ... )
DELETE FROM Employe -- refus de suppression
```

2.
```
CREATE TABLE Departement (
      ..., dptMgr CHAR(3) NOT NULL CONSTRAINT ManagerFk
      REFERENCES Employe (empNo) ON DELETE SET NULL,
      ... )
DELETE FROM Employe -- tous les dptMgr sont à NULL
```

3.
```
CREATE TABLE Departement (
      ..., dptMgr CHAR(3) NOT NULL CONSTRAINT ManagerFk
      REFERENCES Employe (empNo) ON DELETE CASCADE,
      ... )
DELETE FROM Employe -- tous les départements qui font
                    -- référence aux employés sont supprimés
```

DDL - Schéma conceptuel - Contraintes d'intégrités

Quand sont contrôlées ces contraintes d'intégrité SQL ?

- Par défaut, lors de l'exécution d'une requête DML ajoutant, supprimant ou modifiant un tuple
- Mais ...
- Toute spécification de contrainte peut être complétée d'une clause permettant d'éventuellement retarder le contrôle au commit de la transaction en cours.

```
DEFERRABLE [INITIALLY {DEFERRED | IMMEDIATE }]
```

- Le choix par défaut pourra être modifié dynamiquement par les programmes au travers de l'exécution de la commande suivante en début de transaction.

```
SET CONSTRAINT[S] { liste de contraintes | ALL }
                  { IMMEDIATE | DEFERRED }
```

DDL - Schéma conceptuel - Suppression de table

```
DROP TABLE nomTable [ CASCADE CONSTRAINTS ]
```

- Une suppression d'une table référencée par une autre table n'est pas permise : il faut d'abord supprimer les clés étrangères qui la prennent pour cible.
- **CASCADE CONSTRAINTS** permet d'éluder cette obligation. **A éviter !!!**

DDL - Schéma conceptuel - Modification de table

```
ALTER TABLE nomTable listes-de-clauses-de-modification
```

- ADD : ajout d'attribut, de contrainte
- MODIFY : modification d'attribut
- DROP : suppression d'attribut, de contrainte

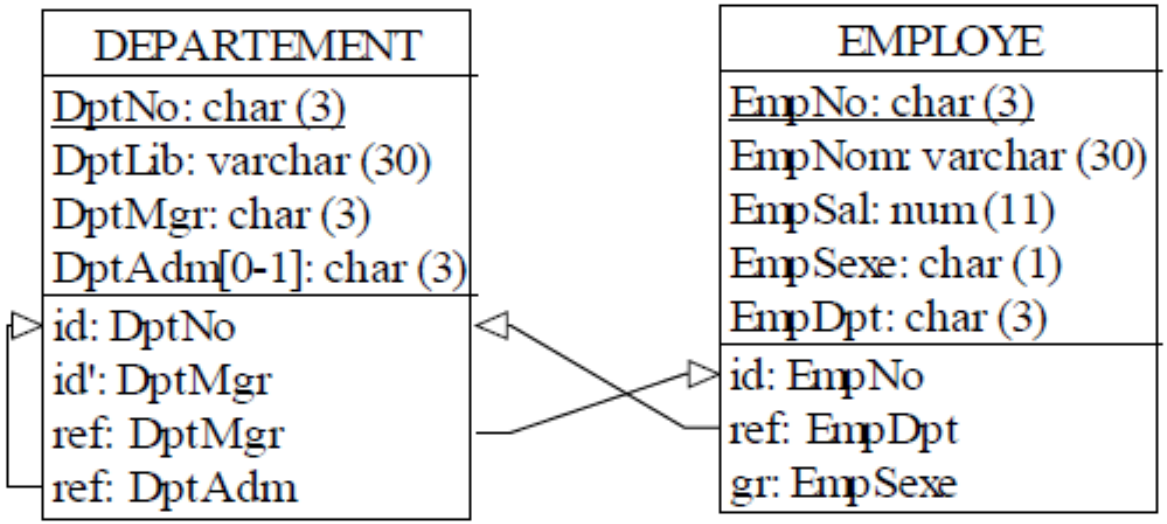
```
ALTER TABLE Employe ADD empDateNaissance DATE;
ALTER TABLE Employe MODIFY empDateNaissance NOT NULL;
ALTER TABLE Employe DROP COLUMN empDateNaissance;
ALTER TABLE Employe ADD CONSTRAINT salPositifCheck
                                    CHECK (empSal > 0) ;
ALTER TABLE Employe DROP CONSTRAINT salPositifCheck ;
```

DDL - Schéma conceptuel - Sémantique

```
COMMENT ON TABLE nomTable IS 'description de la table'

COMMENT ON COLUMN nomTable.nomColonne
                    IS 'description de la colonne'
```

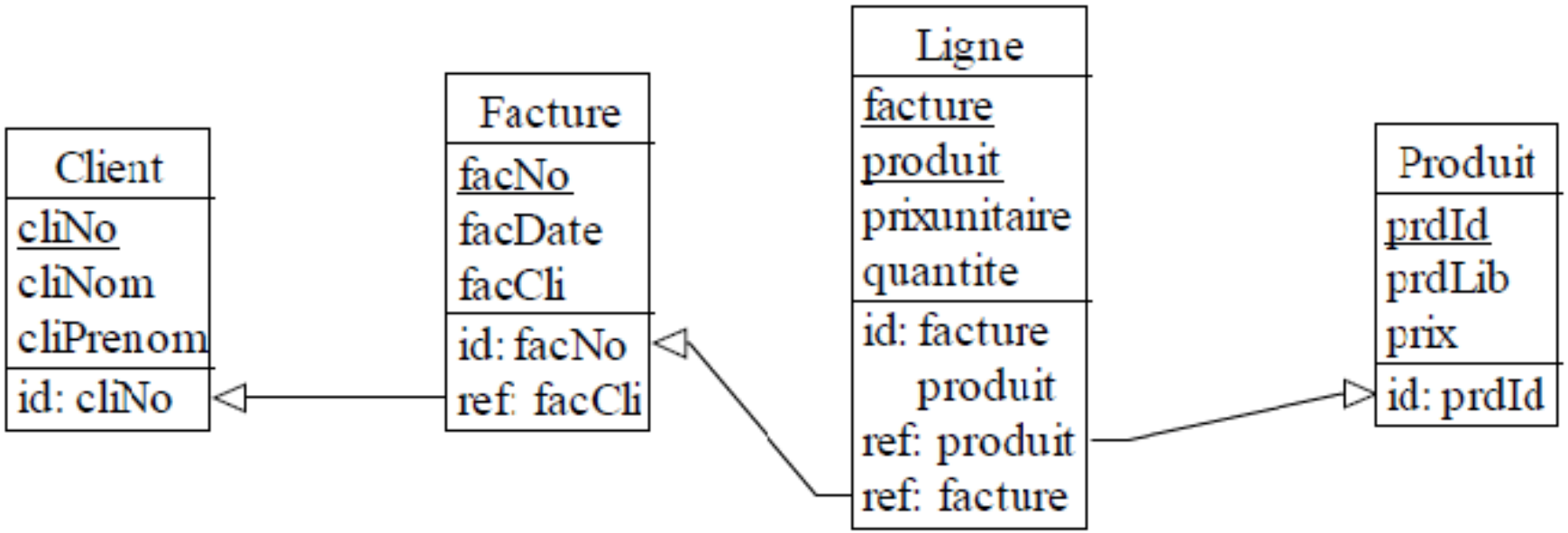
DDL - Schéma conceptuel - Exemple



Contraintes additionnelles

Un département ne peut s'administrer lui-même
Le sexe d'un employé peut valoir M ou F
Le Salaire d'un employé doit être compris entre 50000 et 150000

DDL - Schéma conceptuel - Exemple



DDL - Schémas externes - Vues

Création:

```
CREATE VIEW nom-de-vue [(liste d'attributs)] AS
    SELECT liste d'attributs ...
        [WITH CHECK OPTION]
```

Exemple :

```
CREATE VIEW DepCoord (no, lib, managerNom, masseSal) AS
    SELECT dptNo, dptLib, mgr.empNom, SUM(emp.empSal)
        FROM Departement
            JOIN Employe mgr ON dptMgr=mgr.empNo
            JOIN Employe emp ON dptNo=emp.empDpt
        GROUP BY dptNo, dptLib, mgr.empnom;
```

Suppression:

```
DROP VIEW nom_vue
```

DDL - Schémas externes - Vues

- Table virtuelle « contenant » le résultat d'un SELECT
- Ne stocke pas les données
- Fait référence aux tables de base à travers une requête SELECT
- La requête SELECT est exécutée à chaque référencement de la vue

DDL - Schémas externes - Vues

- Permet l'indépendance logique aux données
- Cache des données aux utilisateurs $\rightarrow$ Sécurité supplémentaire
- Simplifier l'utilisation de tables (nom complexe, nombreuses colonnes,...)
- « Sauvegarder » des requêtes SELECT sous un nom
- Masquer des jointures fréquemment utilisées

DDL - Schémas externes - Manupilation des vues

Idéalement, toute manipulation permise sur une table devrait être permise sur une vue.

Ceci est bien évidemment un désir irréalisable.

Exemple: pas de modification, suppression ou ajout possible sur cette vue

```
CREATE VIEW masseSal(masse) AS
    SELECT SUM(empSal) FROM Employe
```

Il y a donc des restrictions (qui peuvent varier fortement d'un SGBD à l'autre)

DDL - Schémas externes - Manupilation des vues

Restrictions en sélection

- On ne peut réaliser de groupage sur un attribut défini par le biais d'une fonction synthétique (SUM, AVG,...)
- De plus, un tel attribut ne peut être l'argument d'une fonction synthétique

```
CREATE VIEW masseSal(dpt, masse, nbEmp) AS
    SELECT empdpt, SUM(empSal), COUNT(*)
        FROM Employe
        GROUP BY empdpt

SELECT nbEmp, COUNT(*)
    FROM masseSal      GROUP BY nbEmp -- interdit

SELECT AVG(masse) FROM masseSal -- interdit
```

(mais accepté par exemple par Oracle)

Restrictions en mise à jour (INSERT, UPDATE, DELETE)

- Une mise à jour doit être faite sur une seule ligne des tables de base. Donc la vue ne peut pas avoir
  - d'instructions ensemblistes (UNION, INTERSECT, EXCEPT)
  - de fonctions synthétiques
  - de clause GROUP BY

Restrictions en mise à jour (INSERT, UPDATE, DELETE)

- Pour la mise à jour d'une vue avec jointure, la vue doit préserver l'unicité de la clé des tables de base.

```
CREATE VIEW chef AS
    SELECT dptno, dptlib, dptmgr,
           empno, empnom, empsal, empsexe, empdpt
      FROM employe e
          JOIN departement d ON d.dptno = e.empdpt;
```

- La table de base departement n'est pas clé-préservée
- La table de base employe est clé-préservée

Restrictions

- En ajout:
  - Les attributs de la table non présents dans la vue seront affectés de valeur NULL
- En suppression et modification:
  - La vue ne doit pas contenir de clause DISTINCT
  - La clause WHERE ne peut contenir un select imbriqué corrélé
- En modification:
  - Les expressions ne peuvent être modifiées (par ex. salaire *12)

Clause **WITH CHECK OPTION**

Pour interdire aux utilisateurs qui ont le droit UPDATE ou INSERT sur la vue d'effectuer des manipulations qui génèrent des tuples qui ne sont pas visualisables au travers de la vue.

DDL - Schéma interne

Spécification de

- caractéristiques de stockage : fichiers, disque, extends,dataSpace, ...
- chemins d'accès : index, clusters, ...

Uniquement la notion d'index cette année et seulement le type le plus courant d'index : le B-Tree (arbre B) arbre balancé. Ces notions sont abordées aux laboratoires spécifiques à la section Gestion.

Crédits

Slides pour Persistance des données à l'ESI,
école supérieure d'informatique.

Crédits

Linux, pandoc, beamer, LaTeX