Conception de base de données

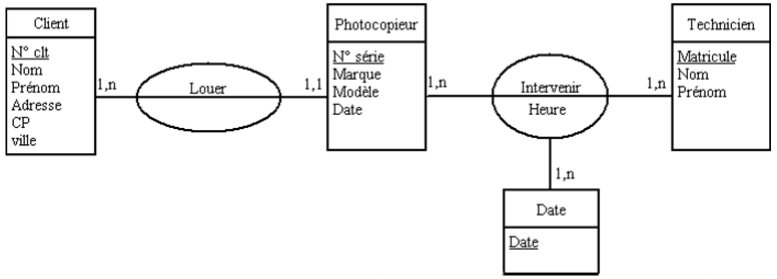
Traduction schéma relationnel ⇒ SQL

L3 Informatique

Antoine Spicher
antoine.spicher@u-pec.fr

Traduction EA ⇒ SR

- Big Picture…



« Réalité »
(Stock, salle / enseignement, compte, …)

Schéma conceptuel Entités/Associations

Conception

Schéma relationnel/logique

Client			
A	B	C	D
1	A	5	C
1	B	8	D

Produit		
A	B	E
1	B	9
2	B	7

SQL

Base de données
représentation de la réalité

Plan

- Types et domaines en SQL
- Création de table
- Définition de vue
- Modification de la table

Types et domaines en SQL – Types SQL

- Types de base
  - CHAR(n) : chaîne de caractères de longueur fixe n
  - VARCHAR(n) : chaîne de caractère de longueur variable avec taille max. n
  - INT / INTEGER : entier standard

    Précision dépendant de la machine
  - SMALLINT : sous-ensemble des entier (*short*)

    Précision dépendant de la machine
  - REAL, DOUBLE PRECISION : flottant standard

    Précision dépendant de la machine
  - NUMERIC(p, d) : nombre de taille fixe avec précision paramétrable
    - p : nombre de chiffres, d : position de la virgule
    - Exemple : NUMERIC(3,1) représente 44.5 mais pas 444.5 ou 0.32
  - FLOAT(n) : nombre flottant avec une précision d'au moins n décimales

Types et domaines en SQL – Types SQL

- Représentation du temps
  - DATE
    - Date au format : YYYY-MM-DD (ex. ‘2012-02-20’)
  - TIME / TIME(p)
    - p, précision pour les secondes (défaut, $p = 0$)
    - Horaire au format : HH:MM:SS.SSS (ex. ‘09:30:45.678’ avec $p = 3$)
  - TIMESTAMP / TIMESTAMP(p)
    - p, précision pour les secondes (défaut, $p = 6$)
    - Date avec horaire au format : YYYY-MM-DD HH:MM:SS.SSS
      (ex. ‘2012-02-20 09:30:45.6789’ avec $p = 4$)
  - Fonctions : CURRENT_DATE, LOCALTIME, EXTRACT(f FROM d)
    avec $f \in \{$year, month, day, hour, minute, second$\}$
  - INTERVAL
    Type des différences entre dates (nb. de jours) ou horaires (nb. de secondes)

Types et domaines en SQL – Déclaration

- Déclaration d'un nouveau *type* de données
  - Syntaxe

    CREATE TYPE *T* **AS** type **FINAL**

  - SQL est fortement typé
    - Erreur d'assignement (*Dollars* et *Euros* incompatibles, cf. ci-dessous)
    - Nécessité d'un *cast* pour effectuer des opérations : **CAST**(… **TO** …)
  - Exemple

```
CREATE TYPE TDollars AS NUMERIC(12,2) FINAL;

CREATE TYPE TEuros AS NUMERIC(12,2) FINAL;

CREATE TABLE service (
   nom      VARCHAR(20),
   budget TEuros);

… CAST (  (CAST (budget TO NUMERIC(12,2))) * 1.05 TO TEuros) …
```

Types et domaines en SQL – Déclaration

- Déclaration d'un nouveau *domaine* de données
  - Syntaxe

    CREATE DOMAIN *D* **AS** type [**DEFAULT** valeur] [**NOT NULL**] [**CHECK** condition]

  - Description
    - Alias pour simplifier/factoriser les spécifications de domaine
    - Spécification possible de contraintes d'intégrité (cf. les contraintes)
    - Les domaines ne sont pas des types (pas fortement typés)

      pas d'erreur d'assignement, pas de *cast* requis

  - Exemple

    CREATE DOMAIN *DEuros* **AS** NUMERIC(12,2) **NOT NULL CHECK** (*value* >= 0);

    CREATE TABLE *service* (*nom* VARCHAR(20), *budget DEuros*);

    … (*budget* * 1.05) …

    CREATE DOMAIN *Saison* **AS** VARCHAR(9) **NOT NULL**
    CHECK (*value* **IN** ('Printemps', 'Été', 'Automne', **OR** 'Hiver'));

Types et domaines en SQL – Valeur null

- « *Tous les types de données acceptent* null *pour valeur* »

- Interdiction de la valeur null
  - Utilisation du mot clé **NOT NULL** (déclaration de domaine et d'attribut)
  - Cf définition de contraintes

Plan

- Types et domaines en SQL
- Création de table
- Définition de vue
- Modification de la table

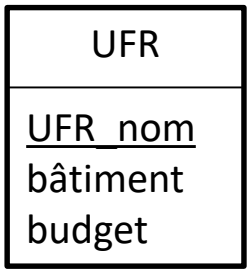
Création de table – Syntaxe générale

- Commande **CREATE TABLE**
  - Syntaxe

CREATE TABLE *R* (
attribut$_1$, attribut$_2$, …,
contrainte$_1$, contrainte$_2$, …)

  - Définition d'une nouvelle table à partir de
    - *R*, un nom
    - (attribut$_1$, attribut$_2$, …), une liste d'attributs
    - (contrainte$_1$, contrainte$_2$, …), une liste de contraintes

- Exemple simple



CREATE TABLE *UFR* (
UFR_nom VARCHAR(20),
bâtiment VARCHAR(15),
budget NUMERIC(12,2),
PRIMARY KEY (*UFR_nom*));

Création de table – Attribut

- Syntaxe générale
  - Syntaxe

    attribut ::= *A* domaine
    domaine ::= *D* | type [**DEFAULT** expr] [**NOT NULL**] [**CHECK** condition]

  - Description
    - Nouvel attribut *A* de type type ou défini par le domaine *D*
    - Possibilité d'affiner le type avec des contraintes (cf. définition des domaines)
      - **DEFAULT** expr

        Valeur par défaut donnée à l'insertion d'un tuple

        Si non précisée, la valeur par défaut est donnée par le type type
      - **NOT NULL**

        Attribut ayant une valeur toujours définie
      - **CHECK** condition

        Assertion sur la valeur de l'attribut

        Cf. les contraintes d'intégrité

Création de table – Attribut

- Syntaxe générale
  - Exemple
    - Création d'une table

```
CREATE TABLE UFR (
        UFR_nomVARCHAR(20) NOT NULL
        bâtiment VARCHAR(15),
        budget    NUMERIC(12,2) DEFAULT 0 NOT NULL CHECK (buget >= 0));
```

    - Valeur par défaut

```
INSERT INTO UFR(UFR_nom) VALUES ('UFR Sciences et technologie');
```

    - Valeur non-nulle

```
INSERT INTO UFR(UFR_nom) VALUES (NULL); -- Echec de la requête
```

    - Assertion arbitraire

```
INSERT INTO UFR(UFR_nom, budget) VALUES ('UFR Sciences et technologie', -1);
                                        -- Echec de la requête
```

Création de table – Contraintes

- Syntaxe générale
  - Syntaxe

    contrainte ::=[**CONSTRAINT** *nom*] spécification [différabilité]

  - Description
    - Possibilité de nommer une contrainte
    - Plusieurs spécifications différentes
      - Clés primaires et candidates
      - Contraintes référentielles
      - Contraintes d'intégrité
    - Possibilité de différer l'évaluation de la contrainte

Création de table – Contraintes et clés

- Clés primaires
  - Syntaxe

    spécification ::= **PRIMARY KEY** (*att_nom*$_1$, *att_nom*$_2$, …)

  - Description
    - Définition de la clé primaire
    - Sous-ensemble des attributs de la table
      - Les attributs ne peuvent pas prendre la valeur null (clause **NOT NULL**)
      - Unicité et existence de la clé

        Vérification à l'insertion d'un tuple, …
      - Exemple

UFR
UFR_nom bâtiment budget

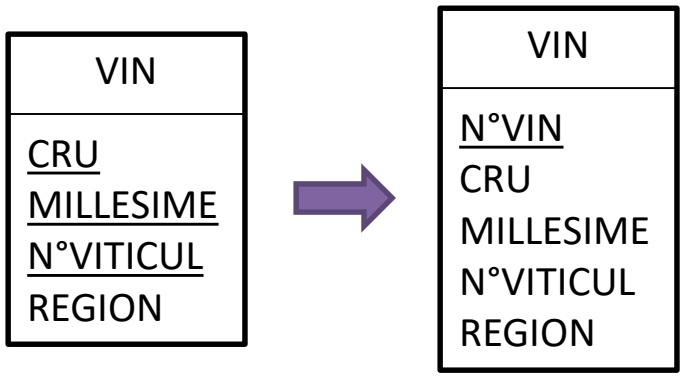
```
CREATE TABLE UFR (
        UFR_nom VARCHAR(20) NOT NULL
        bâtiment VARCHAR(15),
        budget   NUMERIC(12,2) NOT NULL CHECK (buget >= 0),
        PRIMARY KEY (UFR_nom));
```

Création de table – Contraintes et clés

- Clés candidates
  - Syntaxe

    spécification ::= **UNIQUE** (*att_name*$_1$, *att_name*$_2$, …)

  - Description
    - Définition d'un clé candidate (autre que la clé primaire)
    - Unicité uniquement de la clé (valeur null possible)
    - Exemple d'utilisation : *implantation d'une clé artificielle*
      - Ajout d'un attribut identifiant unique pour réduire la clé primaire
      - Schéma : **VIN**(N°VIN, CRU, MILLESIME, N°VITICUL, REGION)



CRU	MILLESIME	REGION	N°VITICUL
Julienas	97	Beaujolais	3

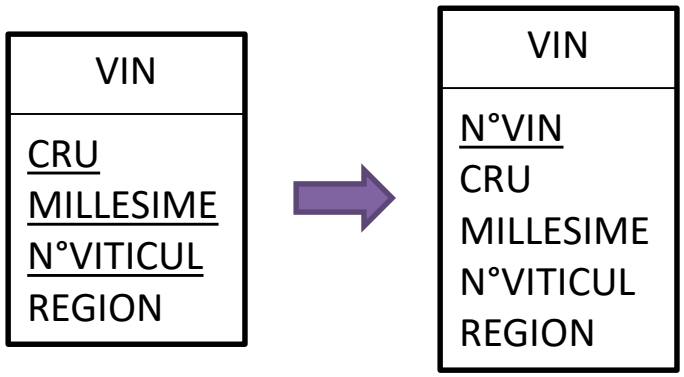
N°VIN	CRU	MILLESIME	REGION	N°VITICUL
100	Julienas	97	Beaujolais	3
101	Julienas	97	Beaujolais	3

Création de table – Contraintes et clés

- Clés candidates
  - Syntaxe

    spécification ::= **UNIQUE** (*att_name*$_1$, *att_name*$_2$, …)

  - Description
    - Définition d'un clé candidate (autre que la clé primaire)
    - Unicité uniquement de la clé (valeur null possible)
    - Exemple d'utilisation : *implantation d'une clé artificielle*
      - Ajout d'un attribut identifiant unique pour réduire la clé primaire
      - Schéma : **VIN**(N°VIN, CRU, MILLESIME, N°VITICUL, REGION)




```
CREATE TABLE VIN (N°VIN INTEGER NOT NULL,
        CRU VARCHAR(15) NOT NULL,
        MILLESIME NUMERIC(4,0) NOT NULL,
        N°VITICUL INTEGER NOT NULL,
        REGION NUMERIC(4,0),
        PRIMARY KEY (N°VIN)
        UNIQUE (CRU, MILLESIME, N°VITICUL));
```

Création de table – Contraintes référentielles

- Clés étrangères
  - Syntaxe

    spécification ::= **FOREIGN KEY** (*attribut*$_1$, *attribut*$_2$, …)
    REFERENCES *relation* [(*rel_att*$_1$, *rel_att*$_2$, …)]
    [**ON DELETE** action] [**ON UPDATE** action]

  - Description
    - Version SQL des contraintes référentielles
    - Sous-ensemble d'attributs référençant la clé primaire d'une autre table
    - Exemple
      - **VIN**(N°VIN, CRU, MILLESIME, N°VITICUL, REGION)
      - **VITICUL** (N°VITICUL, VNOM, VPRENOM, VVILLE)

Création de table – Contraintes référentielles

- Clés étrangères
  - Syntaxe

    spécification ::= **FOREIGN KEY** (*attribut*$_1$, *attribut*$_2$, …)
    REFERENCES *relation* [(*rel_att*$_1$, *rel_att*$_2$, …)]
    [**ON DELETE** action] [**ON UPDATE** action]

  - Description
    - Version SQL des contraintes référentielles
    - Sous-ensemble d'attributs référençant la clé primaire d'une autre table
    - Exemple

```
CREATE TABLE VITICUL (
   N°VITICUL INTEGER NOT NULL,
   VNOM VARCHAR(15),
   VPRENOM VARCHAR(15),
   VVILLE VARCHAR(15),
   PRIMARY KEY (N°VITICUL));
```

```
CREATE TABLE VIN (N°VIN INTEGER NOT NULL,
   CRU VARCHAR(15) NOT NULL,
   MILLESIME NUMERIC(4,0) NOT NULL,
   N°VITICUL INTEGER DEFAULT 1 NOT NULL,
   REGION NUMERIC(4,0),
   PRIMARY KEY (N°VIN),
   UNIQUE (CRU, MILLESIME, N°VITICUL),
   FOREIGN KEY (N°VITICUL) REFERENCES VITICUL);
```

Création de table – Contraintes référentielles

- Clés étrangères
  - Syntaxe

    action ::= **RESTRICT** | **NO ACTION** | **CASCADE** | **SET DEFAULT** | **SET NULL**

  - Description
    - Action induite dans les tables dépendantes suite à une suppression/màj
    - Exemple : … **FOREIGN KEY** (*N°VITICUL*) **REFERENCES** *VITICUL* …

DELETE FROM *VITICUL*
WHERE *N°VITICUL* = 2

?

N°VITICUL	VNOM	VPRENOM	VVILLE
1	Durand	Julien	Strasbourg
~~2~~	~~Dupond~~	~~Albert~~	~~Bordeaux~~
3	Guilain	Thomas	Bordeaux
…	…	…	…

N°VIN	CRU	MILLESIME	REGION	N°VITICUL
…	…	…	…	…
98	Lafite	2000	Beaujolais	2
99	Latour	2001	Beaujolais	3
100	Margaux	1999	Beaujolais	2
101	Margaux	2001	Beaujolais	2
…	…	…	…	…

Création de table – Contraintes référentielles

- Clés étrangères
  - Syntaxe

    action ::= **RESTRICT** | **NO ACTION** | **CASCADE** | **SET DEFAULT** | **SET NULL**

  - Description
    - Action induite dans les tables dépendantes suite à une suppression/màj
    - **RESTRICT/NO ACTION** : rien n'est fait (erreur possible…)
    - Exemple : … **FOREIGN KEY** (*N°VITICUL*) **REFERENCES** *VITICUL*

DELETE FROM *VITICUL*
WHERE *N°VITICUL* = 2

ON UPDATE RESTRICT ON DELETE RESTRICT …

N°VITICUL	VNOM	VPRENOM	VVILLE
1	Durand	Julien	Strasbourg
~~2~~	~~Dupond~~	~~Albert~~	~~Bordeaux~~
3	Guilain	Thomas	Bordeaux
…	…	…	…

N°VIN	CRU	MILLESIME	REGION	N°VITICUL
…	…	…	…	…
98	Lafite	2000	…jolais	2
99	Latour		…olais	3
100	Margaux		Beaujolais	2
101	Margaux	2001	Beaujolais	2
…	…	…	…	…

ERREUR

Création de table – Contraintes référentielles

- Clés étrangères
  - Syntaxe

    action ::= **RESTRICT** | **NO ACTION** | **CASCADE** | **SET DEFAULT** | **SET NULL**

  - Description
    - Action induite dans les tables référentes
    - **CASCADE** : les tuples dépendants sont détruits/mis-à-jours
    - Exemple : … **FOREIGN KEY** (*N°VITICUL*) **REFERENCES** *VITICUL*

DELETE FROM *VITICUL*
WHERE *N°VITICUL* = 2

N°VITICUL	VNOM	VPRENOM	VVILLE
1	Durand	Julien	Strasbourg
~~2~~	~~Dupond~~	~~Albert~~	~~Bordeaux~~
3	Guilain	Thomas	Bordeaux
…	…	…	…

ON UPDATE CASCADE ON DELETE CASCADE …

N°VIN	CRU	MILLESIME	REGION	N°VITICUL
…	…	…	…	…
~~98~~	~~Lafite~~	~~2000~~	~~Beaujolais~~	~~2~~
99	Latour	2001	Beaujolais	3
~~100~~	~~Margaux~~	~~1999~~	~~Beaujolais~~	~~2~~
~~101~~	~~Margaux~~	~~2001~~	~~Beaujolais~~	~~2~~
…	…	…	…	…

Création de table – Contraintes référentielles

- Clés étrangères
  - Syntaxe

    action ::= **RESTRICT** | **NO ACTION** | **CASCADE** | **SET DEFAULT** | **SET NULL**

  - Description
    - Action induite dans les tables référentes
    - **SET DEFAULT** : les tuples dépendants prennent la valeur par défaut
    - Exemple : … **FOREIGN KEY** (*N°VITICUL*) **REFERENCES** *VITICUL*

ON UPDATE SET DEFAULT ON DELETE SET DEFAULT …

DELETE FROM *VITICUL*
WHERE *N°VITICUL* = 2

N°VITICUL	VNOM	VPRENOM	VVILLE
1	Durand	Julien	Strasbourg
~~2~~	~~Dupond~~	~~Albert~~	~~Bordeaux~~
3	Guilain	Thomas	Bordeaux
…	…	…	…

N°VIN	CRU	MILLESIME	REGION	N°VITICUL
…	…	…	…	…
98	Lafite	2000	Beaujolais	**1**
99	Latour	2001	Beaujolais	3
100	Margaux	1999	Beaujolais	**1**
101	Margaux	2001	Beaujolais	**1**
…	…	…	…	…

Création de table – Contraintes référentielles

- Clés étrangères
  - Syntaxe

    action ::= **RESTRICT** | **NO ACTION** | **CASCADE** | **SET DEFAULT** | **SET NULL**

  - Description
    - Action induite dans les tables référentes
    - **SET NULL** : les tuples dépendants prennent la valeur null
    - Exemple : … **FOREIGN KEY** (*N°VITICUL*) **REFERENCES** *VITICUL*

DELETE FROM *VITICUL*
WHERE *N°VITICUL* = 2

ON UPDATE SET NULL ON DELETE SET NULL …

N°VITICUL	VNOM	VPRENOM	VVILLE
1	Durand	Julien	Strasbourg
~~2~~	~~Dupond~~	~~Albert~~	~~Bordeaux~~
3	Guilain	Thomas	Bordeaux
…	…	…	…

N°VIN	CRU	MILLESIME	REGION	N°VITICUL
…	…	…	…	…
98	Lafite	2000	Beaujolais	**NULL**
99	Latour	2001	Beaujolais	3
100	Margaux	1999	Beaujolais	**NULL**
101	Margaux	2001	Beaujolais	**NULL**
…	…	…	…	…

Création de table – Contraintes d'intégrité

- Vérification de propriétés *ad-hoc*
  - Syntaxe

    spécification ::= **CHECK** condition
    condition ::= **EXISTS** (requête) | expr relop expr | (condition)
    | **NOT** condition | condition **AND** condition | …

  - Description
    - Spécification d'une propriété quelconque
    - Cf. la traduction SQL des contraintes d'intégrité

Création de table – Différabilité

- Différabilité
  - Syntaxe

    différabilité ::= **INITIALLY DEFERRED** | **DEFERRABLE** | **NOT DEFERRABLE**

  - *Transaction*
    - Certaines fonctionnalités ne peuvent être réalisées qu'à travers une succession d'opérations SQL
      - Comment s'assurer que la cohérence de la base ?
      - Comment annuler en cas d'échec ?
    - Utilisation de la notion de transaction
      - Définition : « *Suite d'opérations effectuées sur une base de données* »
      - Atomicité, entrelacement des transactions
    - Syntaxe SQL

      START TRANSACTION

      Liste de manipulations / requêtes SQL

      COMMIT (si aucune erreur) ou **ROLLBACK** (s'il y a des erreurs)

Création de table – Différabilité

- Différabilité
  - Syntaxe

    différabilité ::= **INITIALLY DEFERRED** | **DEFERRABLE** | **NOT DEFERRABLE**

  - *Transaction*
    - Exemple : *Comptabilité en partie double*
      - Chaque opération sur 2 comptes (crédité vs. débité) avec un bilan nul
      - Relation Operation(n°op, type, montant, compte)

```
START TRANSACTION
INSERT INTO Operation(n°op, type, montant, compte)
        VALUES (201, 'débit', 100€, 'Compte marchandise');
INSERT INTO Operation(n°op, type, montant, compte)
        VALUES (201, 'crédit', -100€, 'Compte chéquier');
COMMIT
```

Création de table – Différabilité

- Différabilité
  - Syntaxe

    différabilité ::= [**NOT DEFERRABLE** | **DEFERRABLE**
    [**INITIALLY DEFERRED** | **INITIALLY IMMEDIATE**]]

  - Différabilité des contraintes pendant les transactions
    - Quand évaluer une contrainte
      - **NOT DEFERRABLE** : la contrainte est vérifiée immédiatement
      - **DEFERRABLE** : la contrainte peut être différée
      - **INITIALLY DEFERRED** : différable mais vérifiée immédiatement
      - **INITIALLY IMMEDIATE** : différable et vérifiée à la fin
    - Exemple

      … **CHECK** (**EXISTS** (**SELECT** '' **FROM** (**SELECT** **SUM**(*montant*) **AS** *tot* **FROM** *Operation*) = 0) **WHERE** *tot* = 0)) **DEFERRABLE INITIALLY DEFERRED** …

Plan

- Types et domaines en SQL
- Création de table
- Définition de vue
- Modification de la table

Définition de vues

- Présentation
  - Définition

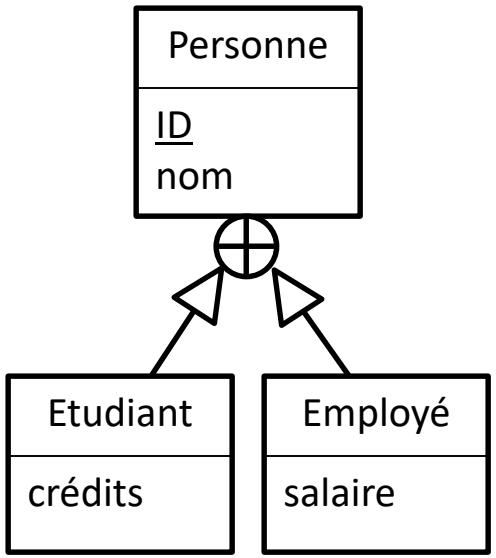
    « *Relation virtuelle définie à partir d'une requête* »
  - Intérêts
    - Cacher de l'information à l'utilisateur (gestion de droit, cf. SQL GRANT)
    - Retrouver les concepts du MCD disparu
  - Syntaxe

    CREATE TABLE *R* **AS** requête
  - Exemple

    CREATE TABLE *Etudiant* (
    ID INT, *nom* CHAR(10), *crédits* INT, **PRIMARY KEY** (*ID*));

    CREATE TABLE *Employé* (
    ID INT, *nom* CHAR(10), *salaire* INT, **PRIMARY KEY** (*ID*));

    CREATE VIEW *Personne* **AS**
    (**SELECT** *ID*, *nom* **FROM** *Etudiant*) **UNION** (**SELECT** *ID*, *nom* **FROM** *Employé*);



Définition de vues – Représentation

- *View expansion*
  - Utilisable dans n'importe quelle requête

    dont dans la définition de nouvelles vues
  - Vue = alias pour une requête : macro-expansion
    - Relation calculée à chaque appel
    - Limitation : définition récursive
  - Exemple
    - Utilisation de la vue

```
SELECT p.nom
FROM   Personne p, Association a
WHERE p.ID = a.PersID AND a.ID = 123
```

    - Version *expansée*

```
SELECT p.nom
FROM   ( (SELECT ID, nom FROM Etudiant) UNION
              (SELECT ID, nom FROM Employé) ) p, Association a
WHERE p.ID = a.PersID AND a.ID = 123
```

Définition de vues – Représentation

- Vue matérialisée
  - Représentation stockée

    Technique de représentation alternative, peu répandue
  - Maintenance laissée à la charge du SGBD
    - Mise à jour à chaque modification d'une dépendance
    - Mise à jour à la première utilisation

Définition de vues – Mise-à-jour d'une vue

- Vue et modification
  - Problème d'ambiguïté
    - Quelle valeur utilisée pour les attributs non-spécifiés
    - Quelle valeur modifiée dans le cas d'une jointure
  - Vue *updatable*
    - La requête est de forme simple
    - La clause FROM concerne une seule table
    - La clause SELECT contient uniquement des noms d'attributs (pas de calcul ni d'agrégation, pas de DISTINCT)
    - Les attributs non spécifiés dans la clause SELECT ne sont pas NOT NULL (en particulier, ils ne font pas partie de la clé primaire)
    - Pas de clause GROUP BY ou HAVING

Plan

- Types et domaines en SQL
- Création de table
- Définition de vue
- Modification de la table

Modification des tables – LMD

- Ajout d'un tuple
  - Syntaxe

    INSERT INTO *R* [(*attr*$_1$, *attr*$_2$, …)] **VALUES** (expr$_1$, expr$_2$, …)

- Mise-à-jour d'un tuple
  - Syntaxe

    UPDATE *R* **SET** *attr*$_1$ = expr$_1$, *attr*$_2$ = expr$_2$, … **WHERE** condition

- Suppression d'un tuple
  - Syntaxe

    DELETE FROM *R* [**WHERE** condition]

Modification des tables – LDD

- Ajout d'une colonne / contrainte
  - Syntaxe : **ALTER TABLE** *R* **ADD** { attribut | contrainte }
- Suppression d'une colonne
  - Syntaxe : **ALTER TABLE** *R* **DROP COLUMN** [**IF EXISTS**] *attr*
- Suppression d'une contrainte
  - Syntaxe
    ALTER TABLE *R* **DROP** { **CONSTRAINT** [**IF EXISTS**] *cstr* | **PRIMARY KEY** }
- Suppression d'une table
  - Syntaxe : **DROP TABLE** [**IF EXISTS**] *R* [**RESTRICT** | **CASCADE**]
- Suppression d'une vue
  - Syntaxe : **DROP VIEW** [**IF EXISTS**] *R* [**RESTRICT** | **CASCADE**]

-- FIN --