Numérique et science informatique
Classe de Terminale

Lycée hoche

année scolaire 2024-2025

Contents

1 Le modèle relationnel . . . 2
1.1 Un peu d'histoire . . . 2
1.2 Domaine . . . 2
1.3 shémas de relation . . . 2
1.4 Vision tabulaire . . . 3
1.5 Contraintes d'intégrité . . . 3
1.6 Clés (primaires) . . . 4
1.7 Clé étrangère . . . 4
1.8 Exemple de schéma de BD . . . 4
1.9 Valeurs manquantes . . . 5
2 La conception de bases de données relationnelles:Le modèle entité-association . . . 6
2.1 Entité et attribut . . . 6
2.2 Les associations . . . 7
2.3 Cardinalité d'une association . . . 8
2.4 Transformation de modèles . . . 8
3 Le langage SQL-Les bases . . . 10
3.1 Interrogation des données (SELECT) . . . 11
4 Langage de définition de données (DDL) . . . 14
4.1 Création de table . . . 14
4.2 Insertion . . . 15
4.3 Modification . . . 16
4.4 Suppression . . . 16
5 Conception de BD . . . 17
5.1 Anomalie d'insertion . . . 17
5.2 Anomalie de modification . . . 18
5.3 Anomalie de suppression . . . 18
5.4 Espace de stockage . . . 18
5.5 Performances . . . 18
5.6 Cohérence des données . . . 19
5.7 Qu'est-ce qu'un bon schéma de relation ? . . . 19

1 Le modèle relationnel

1.1 Un peu d'histoire

- Modèle de données proposé par Edgar T. Codd (IBM San-José)

  1969: Derivability, Redundancy, and Consistency of Relations Stored in Large Data Banks, IBM Research Report

  1970: A Relational Model of Data for Large Shared Data Banks, Communications of the ACM

- Basé sur la logique des prédicats et la théorie des ensembles
- Suppose que toute donnée peut être représentée par une relation n-aire
- Regroupe un ensemble de concepts pour formaliser la description d'articles de fichiers plats
- Différents langages de manipulation de données ayant des pouvoirs d'expression équivalents existent (calcul relationnel,algèbre relationnelle, . . .)

1.2 Domaine

Definition Un domaine est un ensemble de valeurs atomiques.

- Chaque domaine possède un nom et est associé à un type de données
- Le modèle relationnel dans sa forme initiale n'accepte que des valeurs atomiques (ni complexes, ni multivaluées)

Exemples :

- Entier
- Réel
- Chaîne de caractères
- Salaire $= \{4000 - 10000\}$
- Couleur $= \{rouge, vert, bleue\}$

1.3 shémas de relation

Définition

Un schéma de relation $R(A_1, A_2, ..., A_n)$ est constitué d'un nom R et d'une liste d'attributs $A_1, A_2, ..., A_n$. Un schéma de base de données est l'ensemble des schémas de relation.

- Chaque attribut A_i est le rôle joué par le domaine D dans le schéma de relation R.
- On note Dom(A_i) le domaine de l'attribut A_i
- Un schéma de relation définit l'intention de la relation

Exemples :

```
Etudiant(nom : Chaine, prenom : Chaine, numéro : Entier)
```

Etudiant est le nom de la relation .
nom,prenom,numéro désignent les attributs.

Ou encore

```
Voiture(immatriculation : Chaine, type : Chaine, couleur :Couleur)
```

Définition

Une relation r de schéma relationnel $R(A_1, A_2, ..., A_n)$ est un sous-ensemble du produit cartésien de la liste des domaines de R,i.e. $r \subseteq Dom(A_1) \times Dom(A_2) \times ... \times Dom(A_n)$.

- Une relation r de schéma relationnel $R(A_1, A_2, ..., A_n)$ est donc un ensemble de tuples (n-uplets) $t = (v_1, v_2, ..., v_n)$ où $vi \in Dom(A_i)$
- On note $t[A_i, ..., A_j]$ la valeur du tuple t sur les attributs $A_i, ..., A_j$
- Une instance de schéma de relation définit l'extension de la relation

Exemple

Nom	Prénom	Numéro
Dupond	Jean	1215
Durand	Jasques	3256
Martin	Robert	2623

1.4 Vision tabulaire

- Une relation est un tableau (table) à deux dimensions
- Un tuple est une ligne de cette table
- Chaque colonne de la table est nommé indépendamment de son ordre
- Un attribut est le nom donné à une colonne de la table

1.5 Contraintes d'intégrité

Définition

Une contrainte d'intégrité restreint les relations acceptables pour un schéma de BD. C'est une propriété du schéma de la BD (pas des instances). Types de contrainte:

- *de domaine* : le salaire doit être compris entre 4000 et 100000.
- *d'entité* : exprime l'unicité et la non nullité
- *référentielle* : l'ensemble des valeurs d'un attribut doit être inclus dans l'ensemble des valeurs d'un autre attribut

1.6 Clés (primaires)

Définition

Une clé d'un schéma de relation R est un ensemble d'attributs $\{A_i, ..., A_j\}$ de taille minimale tel que : Pour toute relation r de R et pour tout $t, t' \in r$, $t[A_i, ..., A_j] \neq t'[A_i, ..., A_j]$.

- Une clé représente l'ensemble minimal d'attributs qui permet d'identifier de façon unique chaque tuple d'une relation
- Une clé n'est pas forcément unique On choisit en général une clé principale que l'on nomme clé primaire
- Tout schéma de relation doit posséder au moins une clé (contrainte d'entité)

Exemples :

- Nom, Prénom pour Etudiant
- Immatriculation pour Voiture

1.7 Clé étrangère

Définition:

Etant donné un schéma de relation R et un schéma de relation S,
l'expression $R[A_1, ..., A_n] \subseteq S[B_1, ..., B_n]$ est une clé étrangère si:
$\{B_1, ..., B_n\}$ est une clé de S et si pour tout r sur R et s sur S,
pour tout $t \in r$, il existe $t_0 \in s$ tel que $t[A_1, ..., A_n] = t'[B_1, ..., B_n]$.

- Une clé étrangère indique une relation entre deux relations
- Une clé étrangère **définit une contrainte d'intégrité référentielle**
  - lors d'une insertion dans la table référençant, la valeur des attributs doit exister dans la table référencée
  - lors d'une suppression dans la table référencée, les tuples référençant doivent être supprimés

1.8 Exemple de schéma de BD

Shémas de relation
Employé(nss, nom, prénom, salaire, département)
I Département(nom, localisation, responsable)

Clés
nss et nom, prénom pour Employé
nom pour Département

Clés étrangères

Employé[département] $\subseteq$ Département[nom]
Département[responsable] $\subseteq$ Employé[nss]

1.9 Valeurs manquantes

- L'absence d'une valeur dans une table est représenté par un marqueur spécial nommé NULL
  Problème de l'interprétation du marqueur NULL

  1. information pertinente mais inexistante pour le tuple
  2. information non pertinente pour le tuple
  3. information existante mais actuellement inconnue

- Les marqueurs NULL sont à éviter autant que possible
- Les marqueurs NULL sont également problématiques dans les requêtes (logique à trois valeurs)

2 La conception de bases de données relationnelles:Le modèle entité-association

- Ensemble de concepts pour la modélisation des données d'une application
- Ensemble de symboles graphiques associés.
- Proposé en 1976 par Peter Chen.
- Etendu vers des modèles généralisés puis vers l'objet.
- Différentes variantes de la notation.

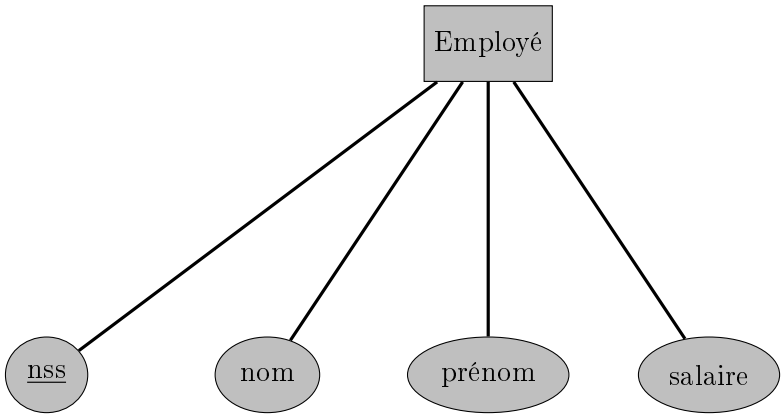
2.1 Entité et attribut

Définition

Une **entité** est un objet du monde réel qui peut être identifié et que l'on souhaite représenter.

- Le type d'entité correspond à l'ensemble des objets ayant des caractéristiques communes.
- Une instance d'entité correspond à un élément particulier du type d'entité.
- Une entité possède un ensemble de propriétés nommées attributs
- Dans le modèle de base, un attribut est simple (atomique).
- Dans les modèles étendus, un attribut peut être composé ou multivalué.
- Chaque entité possède un identifiant ou clé, i.e. un ensemble d'attributs qui permet de différencier une instance d'entité parmi les instances de son type d'entité.

Un exemple d'entité



2.2 Les associations

Les entités sont reliées ensemble par des associations.

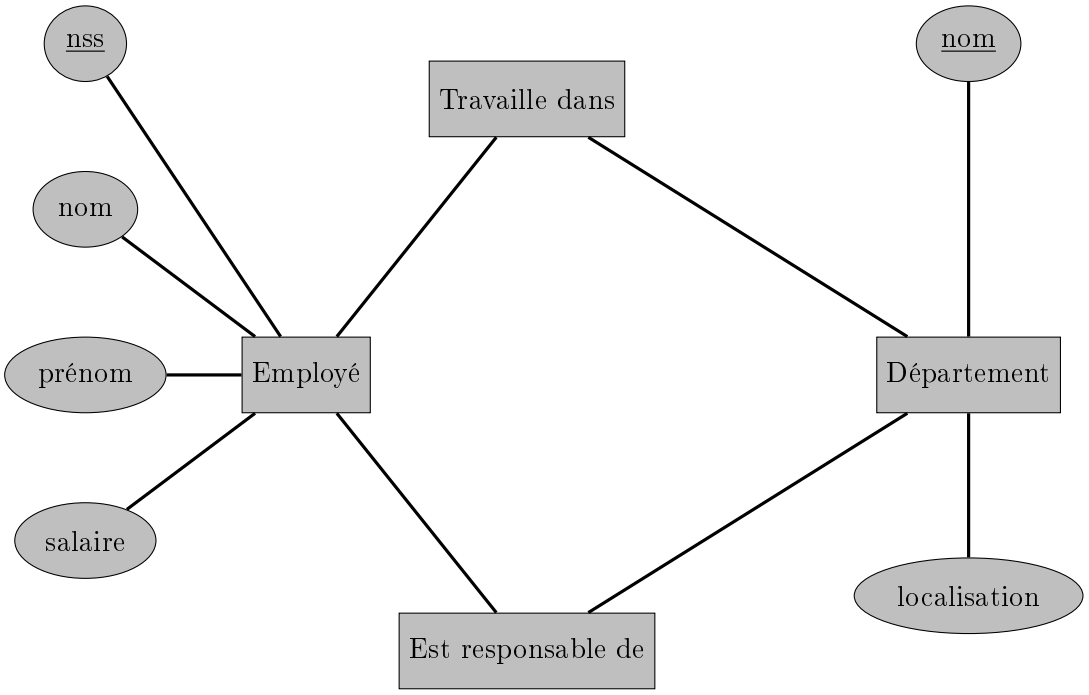
- Le type d'association R est un sous-ensemble du produit cartésien des types d'entités E_i participants, i.e. $R \subset E_1 \times E_2 \times ... \times E_n$ Une instance d'association r_i est un n-uplet $(e_1, ..., e_n)$ où e_i est de type E_i.
- Une association peut avoir des attributs
- Le degré d'une association est le nombre d'entités participantes
- Les associations les plus courantes sont de degré 2 (binaires)
- Une entité participant à une association peut avoir un rôle (le rôle qu'elle joue dans l'association)

2.3 Cardinalité d'une association

Définition

La cardinalité d'une association précise le nombre de fois où une instance d'entité peut participer à l'association.

- On rencontre généralement les cardinalités un à un (one to one ou 1 :1), un à plusieurs (one to many ou 1 :N) ou plusieurs à plusieurs (many to many ou N :N)
- La cardinalité maximum précise le nombre maximum de fois où une instance d'entité peut participer à l'association (1 ou N)
- La cardinalité minimum précise le nombre minimum de fois où une instance d'entité peut participer à l'association (0 ou 1)



2.4 Transformation de modèles

La transformation de modèle est le processus permettant de traduire un modèle M 1 en un modèle M 2 Utilisée à plusieurs niveaux pour la conception de BD

1. Transformation du modèle conceptuel vers le modèle logique
2. Transformation du modèle logique vers le modèle physique

Transformation des entités

- Chaque entité E est transformée en une relation R dont les attributs sont les attributs de E .
- La clé primaire de R est choisie parmi les identifiants de E

Transformation des associations

- (Cas général) Une association A entre les entités $E_1, ..., E_n$ est transformée en une relation R dont les attributs sont les identifiants des E_i et les éventuels attributs de A
- La clé primaire de R est l'ensemble des identifiants des E i
- (Cas des associations binaires 1 :1) Les deux participantes peuvent être fusionnées (envisageable surtout si les deux cardinalité minimums sont à 1)
- (Cas des associations binaires 1 :1 ou 1 :N) La clé de l'entité du côté N peut être ajoutée à la relation représentant l'entité du côté 1.

Exemple de Transformation:

1. L'entité Employé devient la relation *employes* qui possède les attributs *nss* (clé primaire), *nom*, *prenom* ,*salaire* .
2. L'entité Département devient la relation *departements* qui possède les attributs *nom* (**clé primaire**) et localisation.
3. L'association 1 :N Travaille dans entre *Employé* et *Département* est représentée par une **clé étrangère** dans *employes*.
4. L'association 1 :N Est responsable entre *Employé* et *Département* est représentée par une clé étrangère dans *departement* .

3 Le langage SQL-Les bases

- SELECT.
- WHERE
- FROM
- DISTINCT
- ORDER BY
- IS (NOT) NULL
- GROUP BY
- Les fonctions
- Opérations ensemblistes

LE SQL (Structured query Language).Plusieurs facettes:

- des opérations de définition de données
- des opérations de manipulation de données
- et aussi pour la sécurité, les sessions, les transactions, . . .
- SQL-2 : 1992 (ANSI)
- SQL-3 : 1999
- 2003
- 2006

- 2008
- 2011
- 2016

3.1 Interrogation des données (SELECT)

Syntaxe

SELECT liste d'attributs FROM liste de tables WHERE conditions GROUP BY liste d'attributsi HAVING conditions ORDER BY liste d'attributs

SELECT Spécifie le schéma de sortie (projection)
FROM Précise les tables impliquées (produit cartésien)
WHERE Fixe les conditions que doivent remplir les tuples résultats (restriction/sélection et jointures)
GROUP BY Attributs de partitionnement
HAVING Conditions sur les partitions
ORDER BY Critères de tris de la sortie

Opérateurs de restriction

- Comparaison <, <=, =, >=, >, <>
- Logique AND, OR
- Valeur manquante IS NULL, IS NOT NULL
- Textuel LIKE
- Intervalle BETWEEN
- Liste IN Imbrication de blocs IN, EXIST, NOT EXIST, ALL, SOME, ANY

Exemples de requêtes SELECT (I)

– Nom et prénom des employés
– triés en ordre croissant
– selon le nom puis le prénom:

```
SELECT nom, prenom FROM employes
ORDER BY nom ASC, prenom ASC;
```

– Nom des employés (sans doublons):

```
SELECT DISTINCT nom FROM employes;
```

– Tous les attributs des employés:

```
SELECT * FROM employes;
```

– Nom et prénom des employés qui gagnent plus de 2000:

```
SELECT nom, prenom FROM employes
WHERE salaire > 2000;
```

Exemples de requêtes SELECT (II)

– Numéro de sécurité sociale, nom et prénom des employés
– qui gagnent entre 2000 et 3000 et
– dont le nom commence par 'B'

```
SELECT nom, prenom FROM employes
WHERE salaire BETWEEN 2000 AND 3000
AND nom LIKE 'B%';
```

– Nom des employés des départements 'Ventes' – ou 'Marketing' SELECT e.nom AS nom FROM employes e, departements d

WHERE d.nom = e.departement – Jointure
AND d.nom IN ('Ventes', 'Marketing');
– Nom des employés non affectés à un département
SELECT nom FROM employes
WHERE departement IS NULL;

Exemples de requêtes SELECT (III)

– Moyenne des salaires des employés

SELECT AVG(salaire) FROM employes;
– Moyenne des salaires des employés
– par localisation du département

SELECT d.localisation AS Adresse, AVG(salaire) AS Moyenne

FROM employes e, departements d
WHERE d.nom = e.departement – Jointure
GROUP BY d.localisation

– Moyenne des salaires des employés
– dont le nom commence par 'B'
– pour les départements de plus de 10 employés:

SELECT departement, AVG(salaire) FROM employes
WHERE nom LIKE 'B%'
GROUP BY departement
HAVING COUNT(departement) > 10;

Statut de NULL

NULL n'est ni égal, ni différent d'une autre valeur

- Une comparaison classique entre une valeur et NULL ne retourne ni vrai, ni faux mais inconnu
- Les tests spécifiques IS NULL et IS NOT NULL servent à vérifier si une valeur est manquante

4 Langage de définition de données (DDL)

Langage pour la modification du schéma d'une base de données

Opérations

1. I Création d'un objet (CREATE)
2. Suppression d'un objet (DROP)
3. Modification d'un objet (ALTER)

4.1 Création de table

Syntaxe
CREATE TABLE table (definitions de colonnes , contraintes de table)

- Chaque définition de colonne comporte le nom de la colonne, son type et éventuellement une spécification de contraintes d'intégrité
- Il est également possible de définir des contraintes d'intégrité sur la table (plus général)

Quelques types de données:

CHAR(taille) : Chaîne de taille fixe (remplissage avec des ' ')
VARCHAR(taille) Chaîne de taille variable
INTEGER : Entier sur 32 bits
NUMERIC(nbChiffres, nbDecimales) : Nombre en précision fixe (nbChiffres chiffres dont nbDecimales après la virgule)
DECIMAL: Equivalent à NUMERIC
REAL :Nombre en virgule flottante simple précision
DOUBLE PRECISION : Nombre en virgule flottante double précision
DATE/TIME/TIMESTAMP : Date/heure/Date et heure

Types de contraintes d'intégrité

- UNIQUE Pas de doublons
- NOT NULL Pas de valeur manquante
- PRIMARY KEY Clé primaire (équivalent à NOT NULL et UNIQUE)
- FOREIGN KEY/REFERENCES Clé étrangère
- CHECK Contrainte de domaine

Exemple DROP TABLE departement CASCADE;
DROP TABLE employes CASCADE;

CREATE TABLE departements(
nom VARCHAR(50) PRIMARY KEY,
localisation VARCHAR(100) DEFAULT 'Paris',
responsable CHAR(4));

CREATE TABLE employes(
nss CHAR(4) PRIMARY KEY,
nom VARCHAR(20),
prenom VARCHAR(20),
salaire INTEGER CHECK(salaire BETWEEN 1000 AND 4000),
departement VARCHAR(50) REFERENCES departements(nom),
CONSTRAINT employes_uniq_nom_prenom UNIQUE(nom, prenom));

ALTER TABLE departements
ADD CONSTRAINT
departements_fk_responsable
FOREIGN KEY (responsable)
REFERENCES employes(nss);

4.2 Insertion

Syntaxe

INSERT INTO table (liste d'attributs)
VALUES (liste de valeurs)

- Ajoute des tuples dans une table
- On peut préciser la liste des attributs à renseigner : la valeur des autres attributs est fixée à la valeur par défaut de ou à NULL
- VALUES ajoute un tuple à la fois
- L'utilisation d'une requête à la place de VALUES permet d'insérer plusieurs tuples en une seule opération (Bulk insert)

Exemple d'insertion
INSERT INTO employes
VALUES('1234', 'Dupond', 'Jean', 2500, 'Ventes');

```
INSERT INTO employes(nss, nom, prenom)
VALUES('1234', 'Dupond', 'Jean');
```

```
INSERT INTO employes
SELECT nss, nom, prenom, 1500, 'Formation'
FROM candidats
WHERE evaluation > 15;
```

4.3 Modification

Syntaxe
UPDATE table SET liste d'affectationsWHERE condition

- Modifie les valeurs des tuples d'une table
- L'utilisation d'une liste de valeurs permet de modifier la valeur de plusieurs attributs
- La clause WHERE définit les tuples qui seront mis à jour

Exemple de modification

```
UPDATE employes
SET salaire = 3000
WHERE nom = 'Dupond' AND prenom = 'Jean';
```

```
UPDATE employes
SET departement = 'Ventes', salaire = salaire * 1.2
WHERE departement = 'Marketing' AND salaire < 2000;
```

4.4 Suppression

Syntaxe
DELETE FROM table WHERE condition

- Supprime certains tuples d'une table
- La clause WHERE définit les tuples à supprimer

Exemple de suppression

```
DELETE FROM employes
WHERE departement = 'Marketing';
DELETE FROM employes;
```

5 Conception de BD

Comment regrouper les attributs dans des schémas de relation ?

- Tous les choix de schémas de relation sont-ils équivalents ?
- Qu'est-ce qu'un « bon » schéma de relation ?
- Comment choisir les « bons » schémas de relation ?

Exemple
Soient deux schémas de BD :

1.
2. Personne(Nom, CP, Ville) Nom: Nom de la personne
   CP: Code postal
   Ville: Nom de la ville

3. Personne(Nom, CP), Ville(CP, Nom)

   relation Personne
   Nom: Nom de la personne
   CP: Code postal

   relation Ville
   CP: Code postal
   Nom: Nom de la ville

5.1 Anomalie d'insertion

L'ajout d'une entité impose également d'ajouter une autre entité
Sur l'exemple,

dans le schéma 1, ajouter une personne sans connaître son adresse impose d'insérer deux valeurs manquantes (NULL)
dans le schéma 1, ajouter l'information 63000, Clermont-Ferrand impose d'ajouter une personne

5.2 Anomalie de modification

Un changement nécessite de modifier plusieurs tuples Sur l'exemple,

changer Clermont-Ferrand en Clermont dans le schéma 1 nécessite de modifier tous les tuples où la valeur apparaît

5.3 Anomalie de suppression

La suppression d'une entité provoque la suppression d'une autre
Sur l'exemple,

dans le schéma 1, supprimer la dernière personne d'une ville provoque la suppression de la ville

5.4 Espace de stockage

Le choix des schémas de relation a un impact sur la place occupée par les données Exemple

- Supposons qu'il y ait un million de personnes et 10000 villes
- Personne(Nom, CP) est en commun dans les deux schémas
- Supposons que CP occupe 4 octets et Nom (de la ville) 50 octets
- Pour le schéma 1, le stockage du nom de la ville occupe donc 50 millions d'octets
- Pour le schéma 2, la relation Ville occupe $10000 \times (4 + 50) = 540000$ octets soit environ 49Mo de moins

5.5 Performances

-
- L'accès à des données réparties dans plusieurs tables nécessite de réaliser des jointures
- Les données réparties dans plusieurs tables occupent généralement moins de place

Sur l'exemple,
avec le schéma 2, il faut réaliser une jointure entre Personne
et Ville pour avoir l'adresse d'une personne avec le schéma 1, récupérer les noms des personnes nécessite de parcourir 50Mo de plus

5.6 Cohérence des données

Les saisies multiples peuvent entraîner des erreurs dans les données
Sur l'exemple,

saisir à chaque fois le nom de la ville conduira à avoir des orthographes différentes pour une même ville

5.7 Qu'est-ce qu'un bon schéma de relation ?

- Mesures informelles
- Simplicité: Concevoir des schémas de relation dont l'interprétation est simple
- Redondance: Eliminer la redondance permet d'éviter les anomalies de mise à jour et de réduire l'espace de stockage

- Valeurs manquantes: La présence de valeurs manquantes peut indiquer un problème de conception
- Tuples parasites Après la décomposition d'un schéma, les jointures ne doivent pas générer de données non présentes initialement

Valeurs manquantes
Sémantiques

Valeur existante mais inconnue actuellement

Valeur inapplicable

par exemple l'âge d'une personne
par exemple, le nom de jeune fille pour un homme
Pas d'information
on ne sait pas dans laquelle des deux catégories précédentes se trouve la valeur

En pratique

- Seule une valeur NULL est utilisée
- Logique à trois valeurs (vrai, faux et inconnue)
- Extension de l'algèbre relationnelle
- Attention aux résultats de certaines requêtes (sélection, jointure, . . .)