Introduction aux bases de données relationnelles

L3Pro SCT – Bases de données et programmation

Mathieu Sassolas

IUT de Sénart Fontainebleau
Département Informatique

Année 2015-2016
Cours 1

UPEC
UNIVERSITÉ PARIS-EST CRÉTEIL
IUT
SENART
FONTAINEBLEAU

L'ubiquité des données

- L'informatique est la science du calcul.
- Le calcul utilise des données en entrée.
- Le calcul produit des données en sortie.

↪ Besoin de stocker ces données et d'y accéder facilement.

On utilise des bases de données.

Qu'est-ce qu'une base de donnée ?

N'importe quoi qui garde des données.

- Fichier texte brut.
- Fichier texte structuré :
  - Dictionnaire (paires clé-valeur) ;
  - XML ;
  - Valeurs séparées par des virgules (CSV).
- Tableau de données.
- Tables reliées entre elles ⇝ **Relations**

La manière d'accéder aux données est radicalement différente selon le modèle utilisé.

Structure du module

Chaque séance

~1h de cours, ~3h d'exercices (TD et/ou TP).

Planning du cours, par séance :

1. Introduction aux bases de données relationnelles.
2. Modélisation et création des tables.
3. Utilisation des BDD relationnelles.
4. Introduction à la programmation.
5. Programmer en interaction avec une BDD.
6. Du problème au programme – Contrôle.

Évaluation

- Les TP sont relevés via EPREL (coeff 1).
- Contrôle final en TP (coeff 3).

Ce qu'est et ce que n'est pas ce module

Vous y apprendrez

- Les bases du langage SQL.
- Les bases de la programmation objet.
- La conception d'une base de données.
- L'interrogation d'une base à travers un programme.

Vous n'y apprendrez pas

- Les bases de données non-relationnelles.
- La gestion du système de bases de données.
- Les mécanismes du fonctionnement interne de la base de donnée.
- Les problématiques des bases de données réparties.

Plan de la séance

1. Comment stocker de l'information ?
2. Le modèle relationnel
3. Lecture et écriture des données
   - Recherche de données
   - Mise à jour de données
   - Insertion/suppression de tuples
4. TD/TP

Plan de la scéance

1. Comment stocker de l'information ?
2. Le modèle relationnel
3. Lecture et écriture des données
   - Recherche de données
   - Mise à jour de données
   - Insertion/suppression de tuples
4. TD/TP

En deux mots

Les données sont stockées dans des tableaux.

- Chaque colonne du tableau est appelée un attribut.
- Chaque ligne du tableau est appelée un tuple.
- L'entête du tableau est appelé le schéma de la relation.
- L'ensemble des lignes est le contenu de la relation.

Un exemple simple

Médicament	Nom fournisseur	Ville
Xanax	Pfizer	New York
Depakine	Sanofi	Paris
Valium	Roche	Bâle
Plavix	Sanofi	Paris
Viagra	Pfizer	New York
Voltaren	Novartis	Bâle

Pourquoi « relationnel » ?

- Domaine : ensemble D de valeurs
  Par exemple : les entiers, une date, {Depakine,Plavix,Valium,Viagra,Voltaren,Xanax}
- Produit cartésien : $D_1 \times \cdots \times D_n$ ensemble des n-uplets (ou tuples) $(v_1, \ldots, v_n)$ avec $v_1 \in D_1$, $\ldots$ $v_n \in D_n$.
- La relation est un sous-ensemble du produit cartésien.
- Les attributs sont des noms donnés au rôle joué par le domaine.
- Le schéma est la liste des attributs avec leur domaine ; leur ordre n'a pas d'importance (on peut réordonner les colonnes).

⇝ Schéma d'une relation « Hospitalisation » :
IdPatient : entier ; Début : date, Fin : Date.

Pourquoi « relationnel » ?

Produit cartésien (à 72 éléments) de {Depakine,Plavix,Valium,Viagra,Voltaren,Xanax} × {Novartis,Pfizer,Roche,Sanofi} × {Bâle,New York,Paris}:

Depakine	Novartis	Bâle
Depakine	Novartis	New York
Depakine	Novartis	Paris
Depakine	Pfizer	Bâle
Depakine	Pfizer	New York
Depakine	Pfizer	Paris
Depakine	Sanofi	Bâle
Depakine	Sanofi	New York
Depakine	Sanofi	Paris
⋮	⋮	⋮
Xanax	Sanofi	Paris

Clefs et unicité

- Un tuple n'apparaît qu'une seule fois dans la relation.
- Cette unicité est souvent le fait d'un nombre restreint d'attribut :
  - Avec NumSécuritéSociale j'ai au plus un seul individu.
  - Avec (Type,Rue,Ville) j'ai au plus une seule voie.

↪ Ce ou ces attributs est la clef primaire de la relation.

- S'il n'y a pas de clef primaire naturelle, ce peut être seulement un numéro (incrémenté à chaque ajout dans la base, géré par le système de la BDD).

- Ces clefs peuvent alors être référencées par d'autres tables.
- On parle alors de clefs étrangères

Clefs et unicité

- Un tuple n'apparaît qu'une seule fois dans la relation.
- Cette unicité est souvent le fait d'un nombre restreint d'attribut :

Médicament	Fournisseur
Xanax	1651
Depakine	241
Valium	42
Plavix	241
Viagra	1651
Voltaren	315

Id	Nom	Ville
1651	Pfizer	New York
241	Sanofi	Paris
42	Roche	Bâle
315	Novartis	Bâle

- Ces clefs peuvent alors être référencées par d'autres tables.
- On parle alors de clefs étrangères

Plan de la scéance

Les opérations de base

- Recherche de tuples particuliers.
- Modification de valeurs.
- Ajout/suppression de tuples dans une table.

Le langage utilisé: SQL

- *Structured Query Language* (langage de requête structuré).
- Permet la gestion des tables autant que leur interrogation/modification.
- A été normalisé (plusieurs fois): SQL-86, SQL-89, SQL-92, SQL-99, SQL:2003, SQL:2008, SQL:2011.
- Implémenté par divers systèmes de bases de données: PostgreSQL, MySQL, Oracle, SQLite...
- Tous ces systèmes implémentent le même cœur avec des extensions ; aucune n'implémente complètement la norme.

Syntaxe globale

- Les commandes se terminent par ; (point-virgule)
- Commentaires :
```
<Du code SQL>; -- Un commentaire
<D'autre code SQL> /* Plus de commentaires
Car les commentaires /* imbriqués */ c'est fun.
N'est-ce pas? */ <La fin du code>;
```
- Pour donner des choses à exécuter (par exemple en TP...)
  - Lancer l'interpréteur interactif (pour faire le TP) :
    `psql <nom_de_la_base>`
  - Faire lire un fichier (pour exécuter des choses, souvent en début de TP) :
    `psql -f <le_fichier.sql> <nom_de_la_base>`

Dans l'interpréteur

- Méta-commandes : `\dt` pour lister les tables,
  `\d <nom_de_la_table>` pour connaître ses colonnes,
  `\q` pour quitter, `\?` pour de l'aide,
  `\i <nom_de_fichier>` pour lire un fichier SQL.
- Encodage : dépend du système.
  - Normalement tout est en UTF-8.
  - Les tables et colonnes peuvent avoir des noms UTF-8 (avec accents), mais ça reste dangereux.
- Casse (majuscule/minuscule).
  - Les mots clefs SQL ne sont pas sensibles à la casse, mais on a l'habitude de les mettre en majuscule.
  - Les tables, colonnes peuvent être sensible à la casse (dépend du système), on a l'habitude de les mettre en minuscule avec des _ en lieu d'espaces.

Recherche de données : SELECT

Syntaxe de base

- SELECT * FROM <table> ;
- SELECT <colonnes> FROM <table> ;
- SELECT <colonnes> FROM <table> WHERE <condition> ;

Exemples

- SELECT * FROM medicaments ;
- SELECT nom_f, ville_f FROM fournisseurs ;
- SELECT nom_f FROM fournisseurs WHERE ville_f = 'Bâle' ;

IUT SENART FONTAINEBLEAU

Recherche de données : SELECT

Intro BD
M. Sassolas
L3Pro SCT – M7
Cours 1
Introduction
Le modèle relationnel
Lecture et écriture des données
SELECT
UPDATE
INSERT DELETE
TD/TP

medicaments

nom_m	fournisseur_m
Xanax	1651
Depakine	241
Valium	42
Plavix	241
Viagra	1651
Voltaren	315

fournisseurs

id_f	nom_f	ville_f
1651	Pfizer	New York
241	Sanofi	Paris
42	Roche	Bâle
315	Novartis	Bâle

nom_f	ville_f
Pfizer	New York
Sanofi	Paris
Roche	Bâle
Novartis	Bâle

nom_m	fournisseur_m
Xanax	1651
Depakine	241
Valium	42
Plavix	241
Viagra	1651
Voltaren	315

nom_f
Roche
Novartis

'Bâle' ;

15 / 29

IUT SENART FONTAINEBLEAU

Recherche dans plusieurs tables

Intro BD
M. Sassolas
L3Pro SCT – M7
Cours 1
Introduction
Le modèle relationnel
Lecture et écriture des données
SELECT
UPDATE
INSERT DELETE
TD/TP

SELECT ... FROM <table1>, <table2> ;

- Sélectionne dans le produit cartésien des tables.
- Si on veut utiliser plusieurs fois la même table : SELECT * FROM <table> AS t1, <table> AS t2 ;

↪ Utile également pour abréger des noms de tables ou si plusieurs tables ont une colonne du même nom.

↪ Permet aussi de renommer des colonnes

Relier les médicaments à la ville de leur fournisseur

```
SELECT m.nom_m AS medoc, f.ville_f AS ville
FROM medicaments AS m, fournisseurs AS f
WHERE m.fournisseur_m = f.id_f ;
```


IUT SENART FONTAINEBLEAU

Recherche dans plusieurs tables

Intro BD
M. Sassolas
L3Pro SCT – M7
Cours 1
Introduction
Le modèle relationnel
Lecture et écriture des données
SELECT
UPDATE
INSERT DELETE
TD/TP

SELECT ... F

- Sélectionne
- Si on veut
- FROM <t

↪ Utile égale

plusieurs t

↪ Permet au

medoc	ville
Xanax	New York
Depakine	Paris
Valium	Bâle
Plavix	Paris
Viagra	New York
Voltaren	Bâle

Relier les médicaments à la ville de leur fournisseur

```
SELECT m.nom_m AS medoc, f.ville_f AS ville
FROM medicaments AS m, fournisseurs AS f
WHERE m.fournisseur_m = f.id_f ;
```

16 / 29

Arranger les résultats : ORDER BY

Mettre de l'ordre : ORDER BY

SELECT * FROM <table> ORDER BY <colonne1> ASC, <colonne2> DESC ... ;

Lister les fournisseurs par ordre alphabétique inversé de leur ville puis par ordre alphabétique de leur nom

```
SELECT * FROM fournisseurs
ORDER BY ville_f DESC, nom_f ASC ;
```


IUT SENART FONTAINEBLEAU

Arranger les résultats : ORDER BY

Intro BD
M. Sassolas
L3Pro SCT – M7
Cours 1
Introduction
Le modèle relationnel
Lecture et écriture des données
SELECT
UPDATE
INSERT DELETE
TD/TP

fournisseurs

id_f	nom_f	ville_f
241	Sanofi	Paris
1651	Pfizer	New York
315	Novartis	Bâle
42	Roche	Bâle

Mettre de

SELECT

1> ASC,

Lister les fournisseurs par ordre alphabétique inversé de leur ville puis par ordre alphabétique de leur nom

```
SELECT * FROM fournisseurs
ORDER BY ville_f DESC, nom_f ASC ;
```

17 / 29

IUT SENART FONTAINEBLEAU

Arranger les résultats : LIMIT et OFFSET

Intro BD
M. Sassolas
L3Pro SCT – M7
Cours 1
Introduction
Le modèle relationnel
Lecture et écriture des données
SELECT
UPDATE
INSERT DELETE
TD/TP

Ne prendre qu'une partie des résultats : LIMIT et OFFSET

SELECT * FROM <table> LIMIT <max> OFFSET <nb_ign>

- Prend seulement les <max> résultats après en avoir ignoré <nb_ign> (prend les résultats 1+<nb_ign> à <max>+<nb_ign>).

Limiter la requête précédente aux lignes 2 à 3

```
SELECT * FROM fournisseurs
ORDER BY ville_f DESC, nom_f ASC LIMIT 2 OFFSET 1 ;
```

Remarques

- N'a de sens que si on a précédé par un ORDER BY.
- OFFSET 0 peut être omis.
- LIMIT ALL ne limite rien et peut être omis.

IUT SENART FONTAINEBLEAU

Arranger les résultats : LIMIT et OFFSET

Intro BD
M. Sassolas
L3Pro SCT – M7
Cours 1
Introduction
Le modèle relationnel
Lecture et écriture des données
SELECT
UPDATE
INSERT DELETE
TD/TP

Ne prend

SELECT

- Prend

fournisseurs

id_f	nom_f	ville_f
1651	Pfizer	New York
315	Novartis	Bâle

Limiter la requête précédente aux lignes 2 à 3

```
SELECT * FROM fournisseurs
ORDER BY ville_f DESC, nom_f ASC LIMIT 2 OFFSET 1 ;
```

Remarques

- N'a de sens que si on a précédé par un ORDER BY.
- OFFSET 0 peut être omis.
- LIMIT ALL ne limite rien et peut être omis.

18 / 29

Arranger les résultats : DISTINCT

Éliminer les doublons

- SELECT DISTINCT <colonnes> FROM <table> ;
- SELECT DISTINCT (<colonnes_à_distinguer>) <colonnes_à_récupérer> FROM <table> ;

Trouver toutes les villes ayant un labo (avec le nom)

- SELECT DISTINCT ville_f FROM fournisseurs ;
- SELECT DISTINCT ON (ville_f) nom_f,ville_f FROM fournisseurs ;

Remarque

Le tuple effectivement choisi pour représenter l'unique valeur est arbitraire ; on peut affiner avec un ORDER BY.

IUT SENART FONTAINEBLEAU
Intro BD
M. Sassolas
L3Pro SCT – M7
Cours 1
Introduction
Le modèle relationnel
Lecture et écriture des données
SELECT
UPDATE
INSERT DELETE
TD/TP

Arranger les résultats : DISTINCT

fournisseurs

ville_f
New York
Paris
Bâle

fournisseurs

nom_f	ville_f
Pfizer	New York
Sanofi	Paris
Roche	Bâle

Pourquoi pas « Novartis » ?

Trouver toutes les villes ayant un labo (avec le nom)

- SELECT DISTINCT ville_f FROM fournisseurs ;
- SELECT DISTINCT ON (ville_f) nom_f,ville_f FROM fournisseurs ;

Remarque

Le tuple effectivement choisi pour représenter l'unique valeur est arbitraire ; on peut affiner avec un ORDER BY.

IUT SENART FONTAINEBLEAU
Intro BD
M. Sassolas
L3Pro SCT – M7
Cours 1
Introduction
Le modèle relationnel
Lecture et écriture des données
SELECT
UPDATE
INSERT DELETE
TD/TP

Les conditions du WHERE

- Condition sur la valeur contenue dans la colonne : ... `WHERE prix >= 5;`, ... `WHERE ville_f <> 'Bâle';`
- Combinaison booléennes avec AND et OR, et NOT : ... `WHERE ville_f = 'Bâle' OR ville_f = 'Paris';`
- Savoir si une colonne est `null` : `WHERE ville_f IS NULL;`
- Pour les nombres : $<$, $<=$, $=$, $>=$, $>$, $<>$, BETWEEN ... AND ...
- Pour les chaînes de caractères : $=$, $<>$ et expressions régulières avec LIKE ou SIMILAR TO :
  - _ est remplaçable par n'importe quel caractère ;
  - % est remplaçable par n'importe quelle chaîne de caractères ;
  - Avec SIMILAR TO : | (disjonction), * (itération), + (itération stricte), ? (optionnel), $\{m,n\}$ (répétition entre m et n fois), [...] (classe de caractères) peuvent être utilisés.

IUT SENART FONTAINEBLEAU
Intro BD
M. Sassolas
L3Pro SCT – M7
Cours 1
Introduction
Le modèle relationnel
Lecture et écriture des données
SELECT
UPDATE
INSERT DELETE
TD/TP

Les conditions du WHERE

- `ville_f LIKE 'B%'` ⇝ dont la ville commence par B.
- `ville_f LIKE '_a%'` ⇝ dont la ville a a comme deuxième lettre.
- `ville_f LIKE '%sur%'` ⇝ dont la ville contient 'sur'.
- `ville_f SIMILAR TO '% (lès|sur|en) %'` ⇝ dont la ville contient ' lès ', ' sur ' ou ' en '.
- Pour les nombres : $<$, $<=$, $=$, $>=$, $>$, $<>$, BETWEEN ... AND ...
- Pour les chaînes de caractères : $=$, $<>$ et expressions régulières avec LIKE ou SIMILAR TO :
  - _ est remplaçable par n'importe quel caractère ;
  - % est remplaçable par n'importe quelle chaîne de caractères ;
  - Avec SIMILAR TO : | (disjonction), * (itération), + (itération stricte), ? (optionnel), $\{m,n\}$ (répétition entre m et n fois), [...] (classe de caractères) peuvent être utilisés.

Les conditions du WHERE, suite

Avec les mot clef IN, EXISTS, ANY (ou SOME) et ALL

On peut filtrer en utilisant le résultat d'un autre SELECT.

- IN : La valeur se trouve parmi celle renvoyée par le SELECT.
  Il faut que le SELECT ait le même nombre de colonnes !
- EXISTS : Le SELECT retourne au moins un tuple.
- ANY/SOME : Précédé d'un opérateur ($<$, $<=$, $=$, $>=$, $>$, $<>$), cherche si un résultat du SELECT vérifie la condition.
- ALL : Précédé d'un opérateur ($<$, $<=$, $=$, $>=$, $>$, $<>$), cherche si tous les résultats du SELECT vérifient la condition.

Exemples de WHERE complexes

Fournisseurs qui ont au moins 1 médicament

- SELECT * FROM fournisseurs WHERE id_f IN (SELECT fournisseur_m from medicaments) ;
- SELECT * FROM fournisseurs WHERE EXISTS (SELECT * from medicaments WHERE fournisseur_m = fournisseurs.id_f) ;
- SELECT * FROM fournisseurs WHERE id_f = ANY (SELECT fournisseur_m from medicaments) ;

Rues qui ont un homonyme

SELECT DISTINCT type_v,nom_v FROM voies AS v1 WHERE (v1.type_v,v1.nom_v) IN (SELECT type_v,nom_v FROM voies AS v2 WHERE v1.ville_v <> v2.ville_v) ;

Exemples de WHERE complexes, suite

Produits moins cher qu'un concurrent du même type

SELECT nom_p FROM produits WHERE prix_p < ANY (SELECT prix_c from concurrents WHERE concurrents.type = produits.type) ;

Produits les plus compétitifs de leur type

SELECT nom_p FROM produits WHERE prix_p < ALL (SELECT prix_c from concurrents WHERE concurrents.type = produits.type) ;

Mise à jour de données : UPDATE

Syntaxe de base

- UPDATE `<table>` SET `<colonne>` = `<valeur>` ;
- UPDATE `<table>` SET `<colonne>` = `<valeur>` WHERE `<condition>` ;

Remarque

WHERE : même conditions que pour un SELECT.

Exemples

- -- UPDATE medicaments SET fournisseur_m = 315 ;
- UPDATE medicaments SET fournisseur_m = 315 WHERE nom_m = 'Valium' ;
- UPDATE medicaments SET fournisseur_m = 42 WHERE fournisseur_m = 1651 ;

Mise à jour de données : UPDATE

`medicaments` (avant)

nom_m	fournisseur_m
Xanax	1651
Depakine	241
Valium	42
Plavix	241
Viagra	1651
Voltaren	315

`medicaments` (après)

nom_m	fournisseur_m
Xanax	42
Depakine	241
Valium	315
Plavix	241
Viagra	42
Voltaren	315

Exemples

- -- UPDATE medicaments SET fournisseur_m = 315 ;
- UPDATE medicaments SET fournisseur_m = 315 WHERE nom_m = 'Valium' ;
- UPDATE medicaments SET fournisseur_m = 42 WHERE fournisseur_m = 1651 ;

Mise à jour complexes

On peut...

- Mettre à jour plusieurs colonnes à la fois
- Utiliser le résultat d'une première requête comme nouvelle valeur.

Exemples

- UPDATE fournisseurs SET (ville_f,nom_f)=('Nanterre','New Sanofi') WHERE nom_f='Sanofi' ;
- UPDATE fournisseurs SET ville_f='Boulogne', nom_f='Old Sanofi' WHERE nom_f='New Sanofi' ;
- UPDATE medicaments SET fournisseur_m = (SELECT fournisseur_m FROM medicaments AS m WHERE m.nom_m = 'Voltaren') WHERE fournisseur_m = 1651 ;

Insertion de données : INSERT

Syntaxe

- INSERT INTO <table> VALUES <tuple> ;
- INSERT INTO <table> VALUES <tuple1>, <tuple2>, ... ;
- INSERT INTO <table> (<colonne_1>,...,<colonne_n>) VALUES (<val_colonne_1>,...,<val_colonne_n>) ;

Exemples

- INSERT INTO fournisseurs VALUES (471,'GSK','Londres') ;
- INSERT INTO medicaments VALUES ('Amaryl',241), ('Ventolin',471) ;
- INSERT INTO fournisseurs (nom_f) VALUES ('Merck') ;

`medicaments` (avant)

nom_m	fournisseur_m
Xanax	1651
Depakine	241
Valium	42
Plavix	241
Viagra	1651
Voltaren	315

`fournisseurs` (avant)

id_f	nom_f	ville_f
241	Sanofi	Paris
1651	Pfizer	New York
315	Novartis	Bâle
42	Roche	Bâle

`medicaments` (après)

nom_m	fournisseur_m
Xanax	1651
Depakine	241
Valium	42
Plavix	241
Viagra	1651
Voltaren	315
Amaryl	241
Ventolin	471

`fournisseurs` (après)

id_f	nom_f	ville_f
241	Sanofi	Paris
1651	Pfizer	New York
315	Novartis	Bâle
42	Roche	Bâle
471	GSK	Londres
327	Merck	

Suppression de données : DELETE

Syntaxe

- DELETE FROM <table> ;
- DELETE FROM <table> WHERE <condition> ;

Exemples

- -- DELETE FROM fournisseurs ;
- DELETE FROM fournisseurs WHERE ville_f='Bâle' ;
- DELETE FROM fournisseurs WHERE ville_f IS NULL ;

IUT SENART FONTAINEBLEAU
Intro BD
M. Sassolas
L3Pro SCT – M7
Cours 1
Introduction
Le modèle relationnel
Lecture et écriture des données
SELECT
UPDATE
INSERT DELETE
TD/TP

Suppression de données : DELETE

`fournisseurs` (avant)

id_f	nom_f	ville_f
241	Sanofi	Paris
1651	Pfizer	New York
315	Novartis	Bâle
42	Roche	Bâle
471	GSK	Londres
327	Merck	

`fournisseurs` (après)

id_f	nom_f	ville_f
241	Sanofi	Paris
1651	Pfizer	New York
471	GSK	Londres

Exemples

- -- DELETE FROM fournisseurs ;
- DELETE FROM fournisseurs WHERE ville_f='Bâle' ;
- DELETE FROM fournisseurs WHERE ville_f IS NULL ;

IUT SENART FONTAINEBLEAU
Intro BD
M. Sassolas
L3Pro SCT – M7
Cours 1
Introduction
Le modèle relationnel
Lecture et écriture des données
TD/TP

Plan de la scéance

IUT SENART FONTAINEBLEAU
Intro BD
M. Sassolas
L3Pro SCT – M7
Cours 1
Introduction
Le modèle relationnel
Lecture et écriture des données
TD/TP

Fin du cours

↳ C'est l'heure du TD ↵