Processing SQL Statements with JDBC*

Bases de données

JDBC** (Java DataBase Connectivity***) est une interface de programmation créée par Sun Microsystems — depuis racheté par Oracle Corporation — pour les programmes utilisant la plateforme Java.
Elle permet aux applications Java d'accéder par le biais d'une interface commune à des sources de données pour lesquelles il existe des pilotes JDBC. Normalement, il s'agit d'une base de données relationnelle, et des pilotes JDBC sont disponibles pour tous les systèmes connus de bases de données relationnelles.

Bases de données

- ✓ Les bases de données (BDD) permettent de stocker des données.
- ✓ Système de fichiers contenant les données de votre application.
- ✓ Ces fichiers sont totalement transparents pour l'utilisateur

- ❑ Différence avec les fichiers classiques:
  - ✓ le fait que ce n'est pas vous qui les gérez : c'est votre BDD qui les organise, les range et, le cas échéant, vous retourne les informations qui y sont stockées.
  - ✓ Plusieurs utilisateurs peuvent accéder simultanément aux données dont ils ont besoin.
  - ✓ Les données sont ordonnées par « tables », c'est-à-dire par regroupements de plusieurs valeurs.
  - ✓ C'est vous qui créerez vos propres tables, en spécifiant quelles données vous souhaiterez y intégrer.

Bases de données -Quotidien

→Une base de données peut être vue comme une armoire à tiroirs dont vous spécifiez les noms et qui contiennent une multitude de fiches dont vous spécifiez aussi le contenu.

Bases de données -Exemple

Passager : Table

nom	prénom	n° vol	Nbr bagages	N° client
Einstein	Albert	45622	2	154565
Lavoisier	Antoine	45644	4	235002
Raimbault	Arthur	12896	2	544552
Poincaré	Henri	45644	3	781201
Lavoisier	Antoine	45644	1	785154
Einstein	Albert	75906	0	858547

Personne : Table

nom	prénom	age	civilité
Einstein	Albert	45	marié
Lavoisier	Antoine	41	marié
Planck	Max	52	veuf
Poincaré	Henri	45	marié
Raimbault	Arthur	25	célibataire

Dans cette base de données:
« 2 tables »
une dont le rôle est de stocker des informations relatives à des **personnes** (noms, prénoms, âges, civilites)
ainsi qu'une autre qui s'occupe de stocker des **passagers** (noms, prénoms, no de vol, bagages,node client).

- ✓ la BDD symbolise l'armoire
- ✓ chaque table représente un tiroir et
- ✓ chaque ligne de la table correspond à une fiche de ce tiroir !

Bases de données -Utilisation/Interrogation

Vous pouvez interroger les BDD→
en leur posant des questions via un langage précis.

ex.
« Donne-moi la fiche de la table `Personne` pour le nom `Lavoisier` » ;
•« Donne-moi la fiche de la table `Passagers` pour le nom `Einstein` » ;

•etc.

Le langage permettant d'interroger des bases de données: SQL
(Structured Query Language - « langage de requête structurée »).

Pour utiliser une BDD, vous avez besoin:
- ✓ la base de données et
- ✓ le SGBD (Système de Gestion de Base de Données).

Quelle base de données utiliser

Il existe plusieurs bases de données et toutes sont utilisées par beaucoup de développeurs.

PostgreSQL ;
MySQL ;
SQL Server ;
Oracle ;
Access.

Toutes ces bases de données :

- ✓ permettent d'effectuer les actions
- ✓ Chacune possède des spécificités :
  - certaines sont payantes (Oracle),
  - d'autres sont permissives avec les données qu'elles contiennent (MySQL),
  - d'autres encore sont dotées d'un système de gestion très simple à utiliser (MySQL)

❖ PostgreSQL → est **gratuit** et **complet**.

Creation

CREATE TABLE Passagers_Info
(Nom varying character[50], Prenom varying character[50], noVol varying character[50], age integer, designation text, salary integer);

Interrogation simple
SELECT

SELECT att1, att2, . . .
FROM nomTable ;

Recuperer les valeurs contenus dans la table nomTable, en ne gardant que les attributs att1, att2, ...

Schema : Passagers_Info(Prenom, Nom, noVol,nbrBaggages,NoClient)

Donner le nom et le prenom de chaque personne:
SELECT "Prenom", "Nom" FROM public."Passager_Info";

SELECT "Prenom", "Nom", "noVol", "nbrBaggages", "noClient"
FROM public."Passager_Info";
ou
SELECT * FROM public."Passager_Info";
Donner les informations sur chaque passager

**Pour le type « String » : character varying()[]

SELECT

SELECT att1, att2, . . .
FROM table1, table2, . . .
WHERE condition
ORDER BY atti , attj , . . .

Recuperation des donnees dans le FROM
Filtrage en utilisant la condition du WHERE
Tri des resultats restant suivant l'ordre specifie par ORDER BY

INSERT

INSERT INTO nomTable (att1, att2, . . .)
VALUES (val1, val2, …);

Ajouter les valeurs val1, val2 dans la table nomTable,
pour les attributs att1, att2, ...

Schema : Passagers_Info(Prenom, Nom, noVol,nbrBaggages,NoClient)

Ajouter les valeurs pour 1 passager :

INSERT INTO "Passager_Info"("Nom", "Prenom", "nbrBaggages", "noClient", "noVol")
VALUES ('{Einstein}', '{Albert}', 19,'{A1234567}', '{A156}') ;

Ajouter les valeurs pour 3 passager :

INSERT INTO "Passager_Info"("Nom", "Prenom", "nbrBaggages", "noClient", "noVol")
VALUES ('{Einstein}', '{Albert}', 19,'{A1234567}', '{A156}') ,
('{Lavoisier}', '{Antoine}',20, '{A1235787}', '{A150}'),
('{Raimbault}', '{Arthur}', 21,'{A1274567}', '{A159}');

DELETE

```
DELETE FROM Passagers_Info;
```

Supprimer toutes les lignes par la table ‘’Passagers_Info”

```
DELETE FROM Passagers_Info Condition;
```

Ex.

```
DELETE FROM Passagers_Info
WHERE noVol = A150 ;
```

PostgreSQL

https://blog.postgresql.fr/public/Doc_postgresql_9_0.pdf

http://www.enterprisedb.com/products-services-training/pgdownload

https://docs.postgresql.fr/8.4/pg84.pdf