JDBC

(Java DataBase Connectivity)

DESS RSI
GL2

Trouvain Sébastien
Zinebi Tarik

Université de Reims Champagne Ardenne-UFR Sciences

Le : 22/10/2002

Sommaire

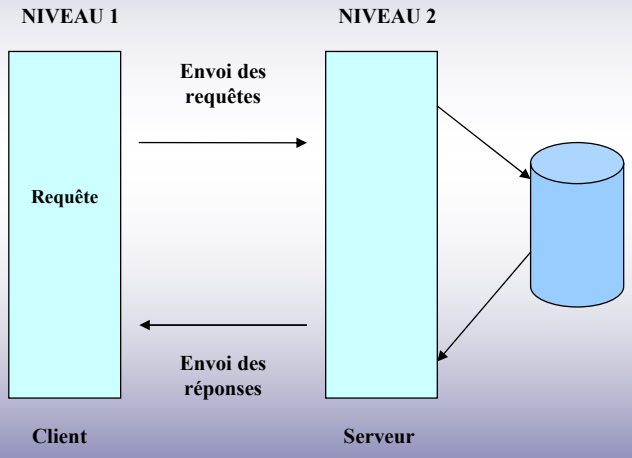
- ✓ Définition du JDBC
- ✓ Les modèles d'accès aux bases de données avec JDBC
- ✓ Utilité des drivers et leurs types
- ✓ Installation de JDBC
- ✓ Les étapes du travail avec une base de données avec JDBC
- ✓ Conclusion

Définitions

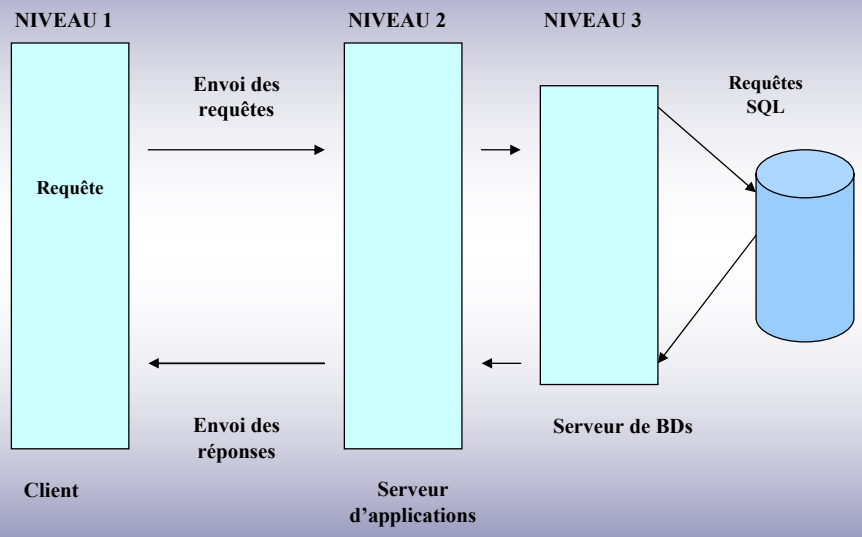
- JDBC permet l'accés à des bases de données avec le langage SQL, à partir d'un programme en Java
- Il est fourni par le paquetage java.sql
- JDBC est indépendant des SGBDs

Les Modèles d'accès aux BDs

1. Modèle à 2 couches



2. Modèle à 3 couches



Utilité des drivers

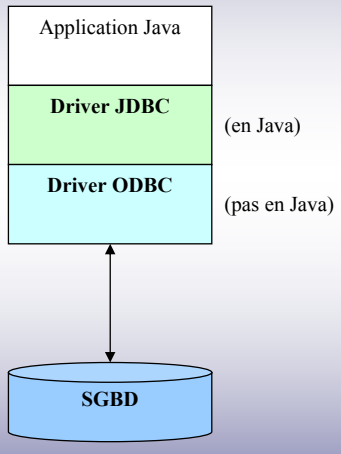
- Pourquoi des drivers ?
- Est ce que les drivers dépendent des SGBDs ?

Types de drivers

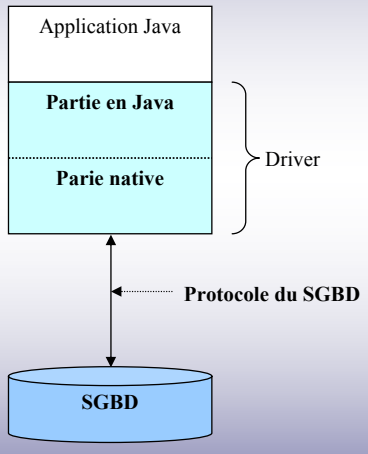
4 Types de drivers :

- Type 1 : pont JDBC-ODBC,
- Type 2 : Driver qui fait appel à des fonctions natives non Java (le plus souvent en langage C)de l'API du SGBD que l'on veut utiliser,
- Type 3 : driver qui permet l'utilisation d'un serveur middleware d'accès à plusieurs types de sources de données,
- Type 4 : driver écrit entièrement en Java, qui utilise le protocole du SGBD.

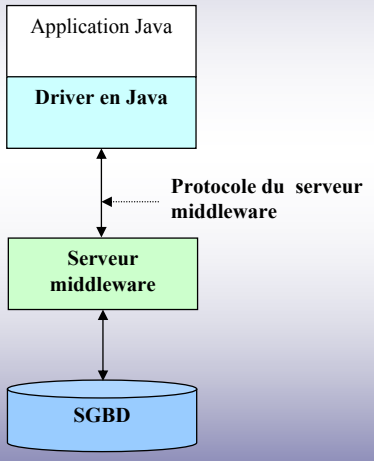
- Pont JDBC – ODBC (type 1)



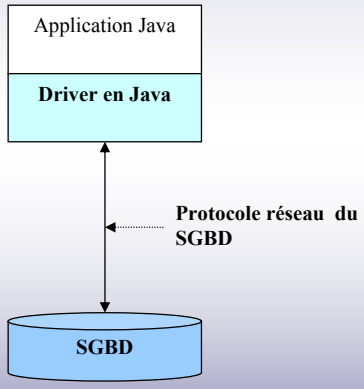
- Driver type 2 qui utilise une API native



- Driver de type 3 d'accès à un serveur middleware



- Driver de type 4 qui utilise le protocole réseau du SGBD



Installation : Contenu de java.sql

- Ce paquetage contient un grand nombre d'interfaces et quelques classes, comme le montre le tableau suivant :

Classes	Interfaces	Exceptions
	Array	
	Blob	
	CallableStatement	
	Clob	
	Connection	
Date	DatabaseMetaData	
DriverManager	Driver	BatchUpdateException
DriverPropertyInfo	PreparedStatement	DataTruncation
Time	Ref	SQLException
Timestamp	ResultSet	SQLWarning
Types	ResultSetMetaData	
	SQLData	
	SQLInput	
	SQLOutput	
	Statement	
	Struct	

Les étapes du travail

1. Charger le pilote
2. Ouvrir une connexion *(connection)*
3. Créer des instructions SQL *(Statement, PreparedStatement ou CallableStatement)*
4. Lancer l'execution de ces instructions :
   - interroger la base *(executeQuery()),*
   - ou mise à jour de la base *(executeUpdate()),*
   - ou tout autre ordre SQL *(execute()),*
5. Fermer la connexion *(close())*

Conclusion

L'**API JDBC** de Java fournit un moyen aux applications java d'accéder aux bases de données et d'écrire des applications de bases de données indépendantes d'une architecture matériel quelconque.