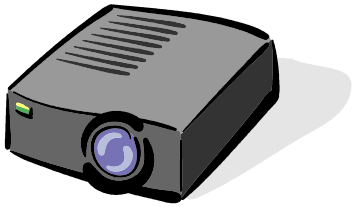
Modernisation et développement d'applications IBM i

Stratégies, technologies et outils

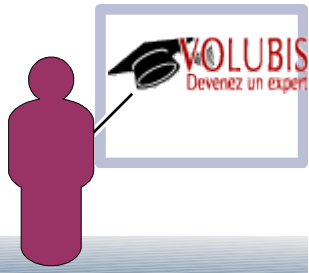
5 et 6 Avril 2012 – IBM Forum de Bois-Colombes



Volubis.fr

Conseil et formation sur OS/400, I5/OS puis IBM ***i*** depuis 1994 !

Christian Massé - cmasse@volubis.fr



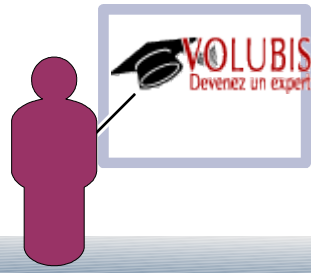
JNI, Utiliser Java en RPG

Il est possible de déclarer et d'utiliser depuis RPG4 :

- des méthodes Java
- des objets (retournés par Java ou destinés à être retournés)

Il est aussi possible de déclarer en RPG4 :

- des fonctions RPG4 devant être utilisées depuis java



JNI, Utiliser Java en RPG

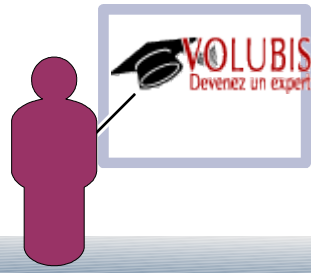
Déclaration d'un objet/java en RPG4

```
D javaChaine       S                 O   CLASS(*JAVA : 'java.lang.string')
```

cette définition peut être placée

- en tant que variable
- en tant que paramètre d'une procédure
- en tant que valeur retournée par une procédure

Ici, l'objet javaChaine va probablement être chargé par l'appel à une méthode sous la forme javaChaine = maMethode()



JNI, Utiliser Java en RPG

Déclaration d'une méthode

```
D maMethode       PR             ?   EXTPROC(*JAVA : 'nom de la classe' :
                                         'méthode' ou *CONSTRUCTOR)
```

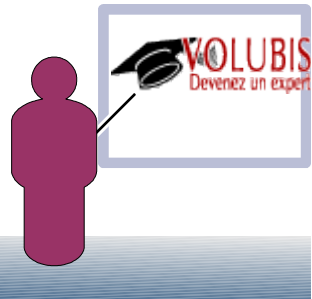
vous devez indiquer le type de donnée retourné

par exemple, la méthode suivante transforme l'objet String en suite d'octets :

```
D StringversChainePR      32767   VARYING
                                    EXTPROC(*JAVA : 'java.lang.String' : 'getBytes')
```

et celle ci transforme l'objet Date en Objet String

```
D DateversString  PR              O   CLASS(*JAVA : 'java.lang.String')
                                      EXTPROC(*JAVA : 'java.lang.Date' : 'toString')
```



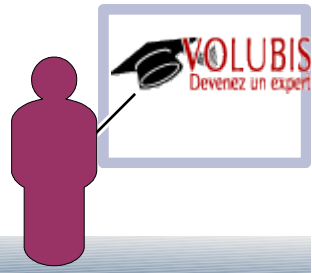
JNI, Utiliser Java en RPG

bien sur vous pourrez écrire :

```
R = StringversChaine( DateversString(objDate) ) ;
```

où objDate et R sont déclarés :

```
D objDate         S               O   CLASS(*JAVA : 'java.lang.Date')
D R               S             10A
```



JNI, Utiliser Java en RPG

Appel et passage de paramètres :

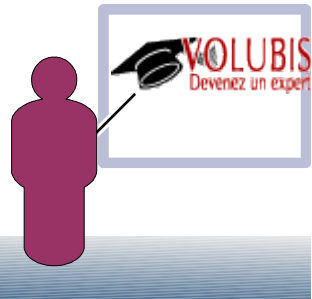
il existe deux types de méthode en Java.

1/ les méthodes statiques ou méthodes de classe, elles sont déclarées

```
Static  typeretour nomMethode() {
}
```

elle ne s'appuient pas sur une instance pour réaliser le traitement,

vous devrez ajouter STATIC, lors du prototypage RPG.



JNI, Utiliser Java en RPG

Appel et passage de paramètres :

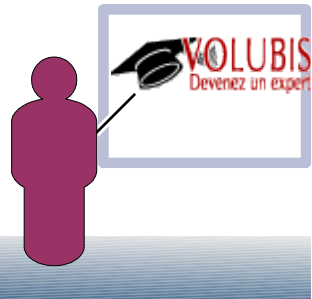
il existe deux types de méthode en Java.

2/ les méthodes qui s'appuient sur une instance (un objet en mémoire)

VOUS DEVREZ AJOUTER un paramètre aux paramètre(s) déclaré(s) dans le prototype: le nom de l'objet instancié.

-> en java : dat1.toString

-> en RPG4 : DateversString(dat1)



JNI, Utiliser Java en RPG

Lors de l'appel, RPG va vérifier si la JVM (machine virtuelle Java, "run-time" java) est démarrée, sinon, il la démarre.

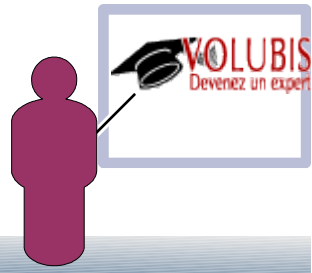
Vous pouvez la démarrer par pgm (et l'arrêter) en utilisant les API JNI.

Java possède sont propre ramasse-miettes (mécanisme qui permet à la JVM de détruire les objets de la mémoire quand ils ne sont plus utilisés)

Enfin, si vous utilisez des classes non standard, précisez bien la CLASSPATH par :

```
ADDENVVAR ENVVAR(CLASSPATH)
          VALUE('/mesclasses/:classes/autresclasses.jar')
```

WRKENVVAR pour voir les variables d'environnement



JNI, Utiliser Java en RPG

Vous pouviez fixer des options pour java, dans l'ordre suivant:

Dans le fichier indiqué par la variable d'env. QIBM_JAVA_PROPERTIES_FILE

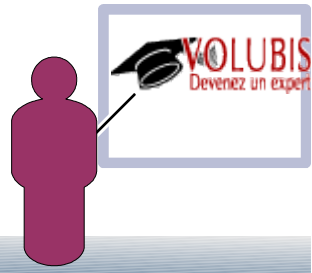
Dans le fichier SystemDefault.properties de la HOMEDIR de l'utilisateur

Dans le fichier SystemDefault.properties de /QIBM/UserData/JAVA400

dont java.class.path et java.version, pour la liste complète voyez
http://publib.boulder.ibm.com/infocenter/iseries/v6r1m0/topic/rzaha/sysprop2.htm

java.version n'est utilisée que pour les versions du JDK "classic", c'est à dire I5/OS, pour celles tournant sous PASE: 32 bits en 5.4, 32 ou 64 en 6.1

il faut renseigner JAVA_HOME (tjs variable d'environnement)



JNI, Utiliser Java en RPG

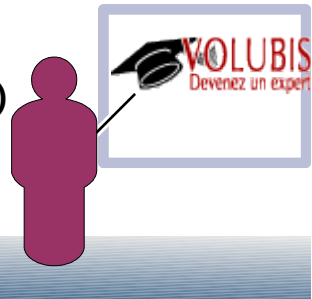
si vous n'indiquez pas de version particulière, le système regarde les versions de JDK installées et choisi dans cette ordre :

Option de 57xxJV1	java	JAVA_HOME
8 IBM tech. 5.0 32 bits	1.5	/Qopensys/QIBM/ProdData/JavaVM/jdk50/32bit
9 IBM tech. 5.0 64 bits	1.5	/Qopensys/QIBM/ProdData/JavaVM/jdk50/64bit
7 Classic 5.0	1.5	/QIBM/ProdData/Java400/jdk15
11 IBM tech. 6.0 32 bits	1.6	/Qopensys/QIBM/ProdData/JavaVM/jdk60/32bit
12 IBM tech. 6.0 64 bits	1.6	/Qopensys/QIBM/ProdData/JavaVM/jdk60/64bit
10 Classic 6	1.6	/QIBM/ProdData/Java400/jdk6
6 Classic 1.4	1.4	/QIBM/ProdData/Java400/jdk14

SF99572 level 7, apporte Java SE 7.0 en version 7

bien sur, les options non installées sont ignorées lors de la recherche, et vous pouvez toujours "forcer" une version par:

```
ADDENVVAR ENVVAR(JAVA_HOME)
 VALUE('/Qopensys/QIBM/ProdData/JavaVM/jdk60/64bit')
```



Projet JDBCR4

- UN programme RPG pouvant utiliser n'importe quelle classe Java (il y a des projets intéressant sur le net d'envoi de mail ou bien de génération de fichier Excel avec POI)

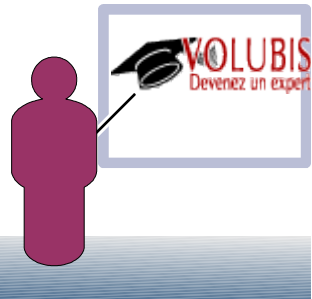
- la plupart des drivers JDBC d'aujourd'hui étant eux-même écrits en Java (driver de type 4)

Le projet JDBCR4 de **Scott Klement** propose une utilisation de JDBC en RPG basée sur un programme de service à compiler et téléchargeable à l'adresse:

http://systeminetwork.com/files/RpgAndJdbc.zip

la documentation est disponible à:

http://www.scottklement.com/presentations/External%20Databases%20from%20RPG.pdf



Projet JDBCR4

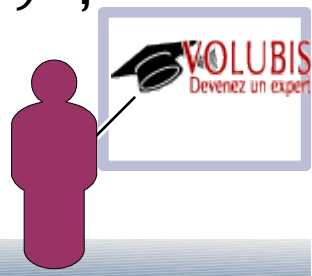
Pour utiliser un Driver JDBC celui-ci doit être de type 4 (écrit en Java), il est en général livré sous forme de fichier .jar

MySQL	mysql-connector-java-3.1.12-bin.jar
Oracle (thin)	ojdbc14.jar
SQL Server (version open source)	sqljdbc.jar jtds-1.2.5.jar
(les n° de version peuvent changer)	
DB2 for i:	jt400.jar
IBM DB2 (autre OS):	db2jcc.jar

Placez ce fichier dans un répertoire de l'IFS et modifiez votre CLASSPATH, par :

```
ADDENVVAR ENVVAR(CLASSPATH)
          VALUE('.:/java:/java/mysql-connector.jar') ,
```

par exemple.

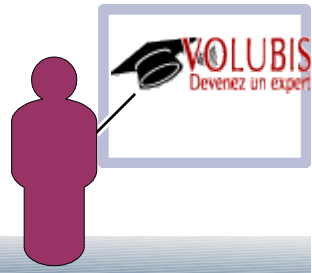


Projet JDBCR4

Écrivez ensuite un pgm java de test pour vérifier la validité du driver

il vous faut connaître la classe du driver

```
SQL Server:com.microsoft.sqlserver.jdbc.SQLServerDriver
jTDS       : net.sourceforge.jtds.jdbc.Driver
Oracle     :oracle.jdbc.OracleDriver
MySQL      :com.mysql.jdbc.Driver
DB2 for i  :com.ibm.as400.access.AS400JDBCDriver
DB2 Autre  :com.ibm.db2.jcc.DB2Driver
```



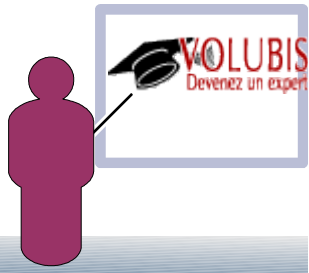
Projet JDBCR4

Écrivez ensuite un pgm java de test pour vérifier la validité du driver

il vous faut connaître la classe du driver
et l'URL de connexion

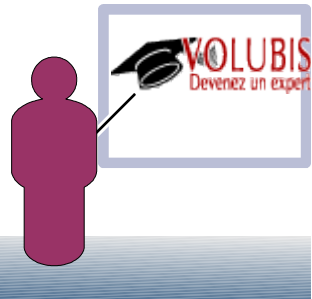
```
SQL Server:jdbc:sqlserver://myserver.example.com:1433
JTDS  :    jdbc:jtds:sqlserver://myserver.example.com:1433
Oracle:    jdbc:oracle:thin:@myserver.example.com:1521:myDataBase
MySQL:     jdbc:mysql://myserver.example.com/myDataBase
DB2 for i: jdbc:as400://myserver.example.com
DB2 Autre: jdbc:db2://myserver.example.com:50000/myDataBase
```

Bien sur vous pouvez (en allant à la "pêche" aux informations) utiliser des drivers non cités ici, comme celui pour PostGreSql (org.postgresql.Driver)



Projet JDBCR4

```
import java.sql.*;


public class testMySql
{
    public static void main (String[] parameters)
    {


        String mysqlurl  = "jdbc:mysql://localhost/maBase";
        String userid    = "root";
        String password  = "xxxxx";
        Connection connection   = null;


        try {
            DriverManager.registerDriver(new com.mysql.jdbc.Driver());


            connection = DriverManager.getConnection ( mysqlurl,
                                                       userid,
                                                       Password );

 . . . / . . .
```



Projet JDBCR4

Passons ensuite à l'écriture RPG

Scott fourni un source à compiler en tant que programme de service **JDBCR4**

→ CRTRPGMOD mylib/JDBCR4 SRCFILE(xxx/QRPGLESRC) DBGVIEW(*LIST)

→ CRTSRVPGM SRVPGM(mylib/JDBCR4) EXPORT(*SRCFILE) SRCFILE(xxx/QSRVSRC)

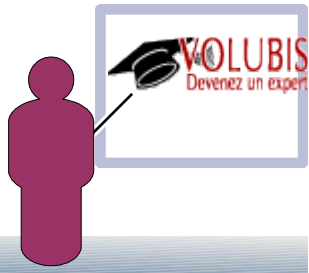
→ CRTBNDDIR BNDDIR(mylib/JDBC)

→ ADDBNDDIRE BNDDIR(mylib/JDBC) OBJ((JDBCR4 *SRVPGM))

→ vos programmes devront commencer par :

```
     H DFTACTGRP(*NO)
     H BNDDIR('JDBC')

     D
      /copy jdbc_h
```

*Nul besoin d'aller voir le contenu du *SRVPGM sauf à être curieux ou à déboguer finement.*

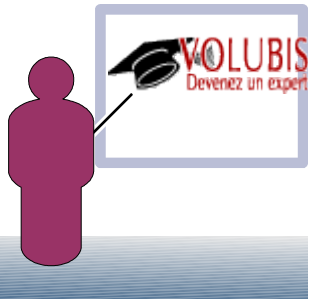


Projet JDBCR4

Passons ensuite à l'écriture RPG

La première fonction à utiliser est JDBC_Connect

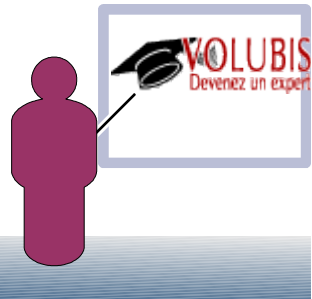
```
 /copy JDBC_H

D userid          s              50a
D passwrd         s              50a
D conn            s                    like(Connection)


 /free

   userid = 'root';
   passwrd = 'xxxxx';
   conn = JDBC_Connect('com.mysql.jdbc.Driver'
                       :'jdbc:mysql://localhost/maBase'
                       :%trim(userid)
                       :%trim(passwrd) );
    if (conn = *NULL);
         errorMsg = 'impossible de se connecter à MYSQL !';
         // Afficher le message.
    endif;


 /end-free
```



Projet JDBCR4

Si vous avez besoin de propriétés particulières à l'utilisation du driver écrivez plutôt :

```
D userid          s              50a
D passwrd         s              50a
D conn            s                    like(Connection)
D prop            s                    like(Properties)
 /free

   userid = 'root';
   passwrd = 'xxxxx';
   prop = JDBC_Properties();

   JDBC_setProp(prop: 'user' : %trim(userid) );
   JDBC_setProp(prop: 'password' : %trim(passwrd));
   JDBC_setProp(prop: 'connectTimeout': '60' );

   conn = JDBC_ConnProp('com.mysql.jdbc.Driver'
                        :'jdbc:mysql://localhost/maBase'
                        :prop) );

   JDBC_freeProp(prop);
```



Projet JDBCR4

par exemple la propriété *databaseName* pour SQL server ou *naming* pour jt400

(voir la liste des propriétés pour chaque driver, par exemple avec Squirrel)

Une fois connecté, vous devez distinguer deux types d'ordres :

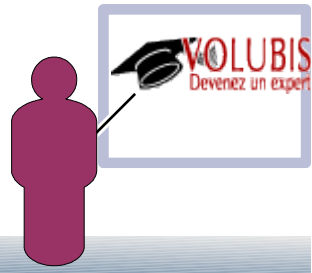
Les ordres immédiats

sont interprétés et exécutés dans la foulée

Les ordres préparés

sont interprétés une fois et exécutés, éventuellement, plusieurs fois.

lors de la préparation, ils peuvent contenir des marqueurs (?), qui seront remplacés par des valeurs à l'exécution.



Projet JDBCR4

→ JDBC_ExecUpd(connexion : 'ordre SQL ne retournant rien')

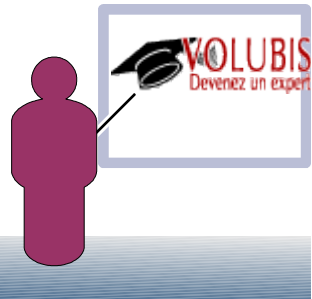
Exécute un ordre SQL comme CREATE..., UPDATE, DELETE, etc (pas de SELECT)

retourne :

```
 0 pour un ordre n'affectant aucune ligne
 n le nombre de lignes affectées
-1 pour un ordre SQL en erreur
```

exemple :

```
 /free

   rc = JDBC_ExecUpd( conn : 'delete from clients'
                             + ' where nocli = 999 ');
          if (rc < 0);
           // signaler une erreur
          endif;
```



Projet JDBCR4

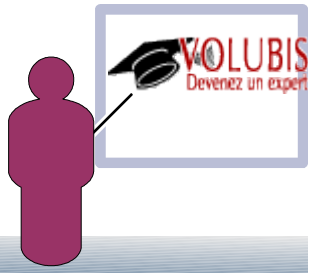
→ JDBC_ExecQRry(connexion : 'ordre SQL de type SELECT')

Exécute un ordre SQL retournant un jeu de résultats

retourne :

un objet ResultSet
*NULL pour un ordre SQL en erreur

```
exemple :
 Resset          s                    like(ResultSet)

/free

  resset= JDBC_ExecQry( conn : 'select nocli, raisoc, ville, tel'
                                + ' from clients where dep = 44');
          if (resset = *null);
           // signaler une erreur
          endif;
```



Projet JDBCR4

→ Pour lire un ResultSet utilisez les fonctions suivantes :

JDBC_nextRow(ResultSet)

se positionne sur la ligne suivante.
Retourne *OFF en cas de "fin de fichier", *ON dans le cas contraire

JDBC_getCol(ResultSet : NumCol)

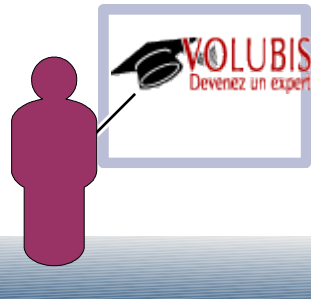
Retourne la valeur d'une colonne dont on fournit le n° dans le SELECT
(la première colonne porte le n° 1)

JDBC_getColByName(ResultSet : NomColonne)

Retourne la valeur d'une colonne dont on fournit le nom

JDBC_freeResult(ResultSet)

Ferme le ResultSet et libère la mémoire
(comme un CLOSE de curseur en SQLRPGLE)



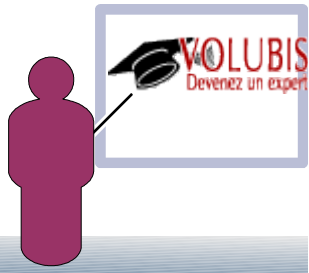
Projet JDBCR4

Exemple :

```
/free

   dow JDBC_nextRow(Resset);
        Dept = JDBC_getCol(Resset: 1);
        EmpNo = %int(JDBC_getCol(Resset: 2));
        Name = JDBC_getCol(Resset: 3);

    // traitement des données lues.

   enddo;

   JDBC_freeResult(Resset)
```

La fonction GetCol retourne toujours une chaîne, quelque soit le type d'origine dans la base de données, à vous de convertir.

```
    pour convertir en numérique : %int(), %uns(), %dec()
    pour les dates/heures       : %date(), %time(), %timestamp()
    pour les variables Unicode  : %ucs2()
```



Projet JDBCR4

une fois le resultSet créé, vous pouvez obtenir les métaData, soit la définition de ce que vous allez recevoir (sorte de DSPFFD)

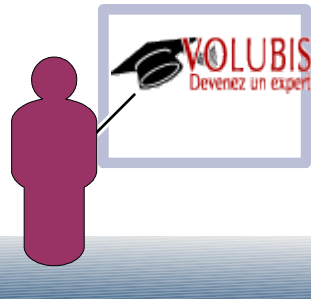
JDBC_getMetaData(ResultSet)
Retourne un objet MetaData décrivant un ResultSet

JDBC_getColCount(MetaData)
Retourne le nombre de colonnes

JDBC_getColName(MetaData : NumCol))
retoune le nom d'une colonne dont on fournit le n°

JDBC_getColDspSize(MetaData: NumCol)
retourne la taille (à l'affichage) d'une colonne

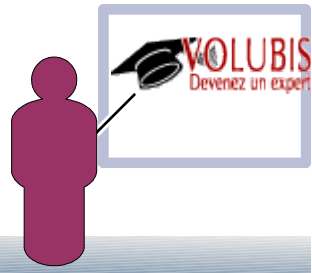
JDBC_getColTypName(MetaData: NumCol)
retourne le type d'une colonne



Projet JDBCR4

- Instructions préparés

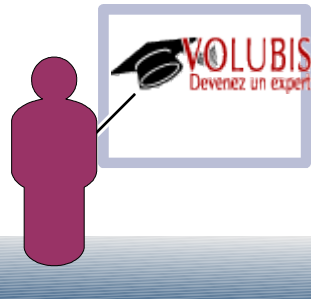
```
JDBC_PrepStmt( connexion : 'ordre SQL valide')

 Retourne un objet PreparedStatement pour l'instruction SQL

  L'instruction est "compilée" et ses exécutions ultérieures seront
   plus rapides.

  peut contenir des marqueurs représentant des valeurs fournies plus tard


JDBC_ExecPrepUpd( PreparedStatement )
  Exécute une instruction préparée qui ne retourne pas d'enregistrements

JDBC_ExecPrepQry( PreparedStatement )
  Exécute une instruction préparée qui retourne un jeu d'enregistrements

JDBC_FreePrepStmt( PreparedStatement )
  libère la mémoire associée à l'objet PreparedStatement
```



Projet JDBCR4

Exemple d'instruction préparée :

```
D Stmt            s                   like(PreparedStatement)
D ResSet          s                   like(ResultSet)

 /free
  // suite à une connexion . .
        Stmt = JDBC_PrepStmt( conn : 'Select nocli, raisoc +
                                      dep, ville           +
                                      from clients order by raisoc');
        if ( stmt = *null );
         // signaler l'erreur
        endif
        ResSet = JDBC_ExecPrepQry( Stmt );
        if (ResSet = *null);
         // autre erreur
        endif;
         // lecture des enregistrement comme vu plus haut

        JDBC_freeResult( ResSet );
        JDBC_freePrepStmt( stmt );
```



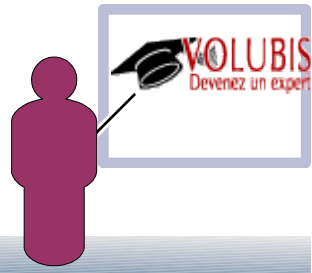
Projet JDBCR4

Si vous placez des marqueurs dans l'instruction préparée, vous devez fournir des valeurs avant l'exécution par :

```
JDBC_setString( stmt : numéro paramètre : 'chaîne');

JDBC_setInt( stmt : numéro paramètre : zone binaire);

JDBC_setDouble( stmt : numéro paramètre : zone virg. flottante );

JDBC_setDecimal( stmt : numéro paramètre : zone décimale);

JDBC_setDate( stmt : numéro paramètre : zone date );

JDBC_setTime( stmt : numéro paramètre : zone heure);;

JDBC_setTimestamp( stmt : numéro paramètre : zone horodatage);
```

Exemple :

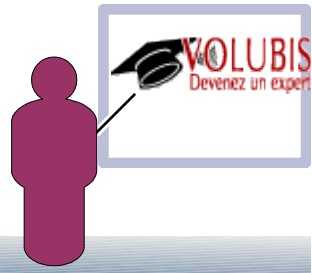
```
nocli = 1234;
JDBC_SetInt( stmt: 1: nocli );
```



Projet JDBCR4

Exemple complet avec un INSERT :

```
/free
  Stmt = JDBC_PrepStmt( conn : 'Insert Into clients  +
                               (nocli, raisoc, dep, ville)
                               values(? , ? , ? , ?) ');


  if ( stmt = *null );
   // erreur
  endif

  JDBC_setInt ( stmt: 1: 4321 );          // constantes OU variables
  JDBC_setString( stmt: 2: 'Volubis');
  JDBC_setDecimal( stmt: 3: 44);
  JDBC_setString( stmt: 4: 'Carquefou');

  if JDBC_execPrepUpd( stmt ) < 0;
   // exécution impossible
  endif;
```



Projet JDBCR4

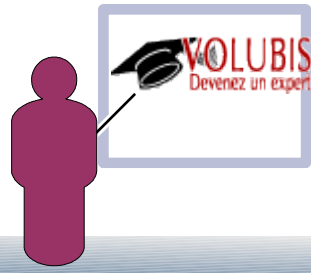
Valeurs nulles

Les fonctions JDBC_getCol et JDBC_Setxxx (int, date, etc...)
possèdent un paramètre supplémentaire, facultatif, de type indicateur.

sur JDBC_getCol il sera positionné à *ON si la colonne est nulle,
*OFF dans le cas contraire.

sur JDBC_SetInt et les autres fonctions d'assignation,

vous le positionnez à *ON pour signaler une valeur nulle,
*OFF dans le cas contraire.

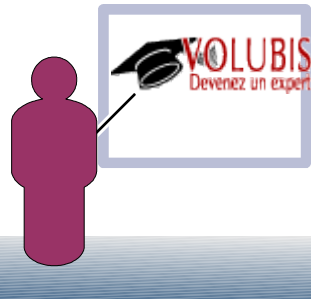


Projet JDBCR4

→ Procédures cataloguées

Vous pouvez aussi utiliser des instructions préparées pour les procédures cataloguées :

```
JDBC_PrepCall( Connection : 'instruction CALL') // préparation

JDBC_RegisterOutParameter( CallableStatement: ParmNum: DataType )
  // prévient JDBC que ce paramètre est en sortie (valeur retour)

JDBC_ExecCall( CallableStatement )              // Exécution

JDBC_FreeCallStmt( CallableStatement )          // libération mémoire
```



Projet JDBCR4

→ Procédures cataloguées

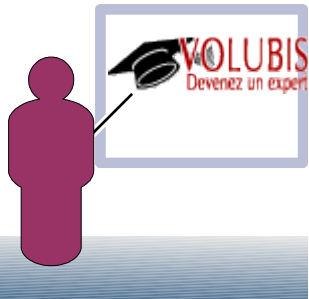
La fonction JDBC_ExecCall peut retourner *ON si un jeu ou plusieurs jeux d'enregistrements sont retournées par la procédure.

JDBC_getUpdateCount(CallableStatement)
Quand une procédure ne retourne pas de jeu d'enregistrements, nombre de lignes modifiées par la procédure.

JDBC_getResultSet(CallableStatement)
retourne un jeu d'enregistrements comme ExecuteQuery

JDBC_getMoreResults(CallableStatement)
passe au jeu d'enregistrements suivant, retourne *OFF s'il n'y en a pas

JDBC_getString(), JDBC_getInt(), JDBC_getShort(), JDBC_getBoolean()
permettent de récupérer les valeurs des paramètres en sortie



Projet JDBCR4

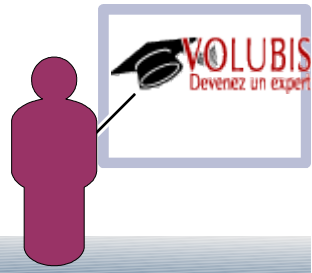
→ Transactions

Enfin, il existe deux fonctions pour gérer les transactions :

JDBC_Commit()

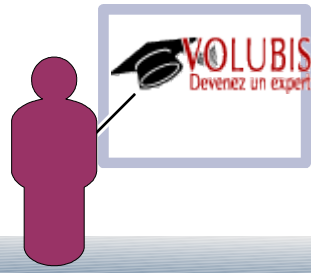
Et

JDBC_Rollback()



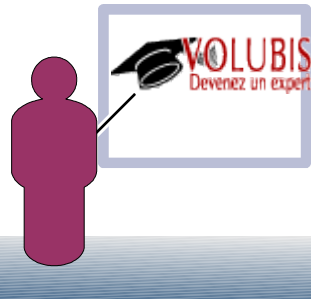
Exemple JDBCR4

```
 *
 *  création de la base de test (MYSQL)
 *  ===================================
 *  create database test;
 *
 *  CREATE TABLE clients (
 *      nocli int(10) unsigned NOT NULL AUTO_INCREMENT,
 *      nom varchar(45) NOT NULL,
 *      dep decimal(2,0) DEFAULT NULL,
 *      datcrt date DEFAULT NULL,
 *      PRIMARY KEY (`nocli`)
 *    );
 *
 *
 *  INSERT INTO clients VALUES (1, 'premier client', 44, '2009-11-02');
 *  INSERT INTO clients VALUES (2, 'deuxieme', 35, '2009-11-02');
 *  INSERT INTO clients VALUES (3, 'et de trois', 22, '2009-10-31');
 *
H DFTACTGRP(*NO) BNDDIR('JDBC')

FQSYSPRT   O    F  132        PRINTER
```



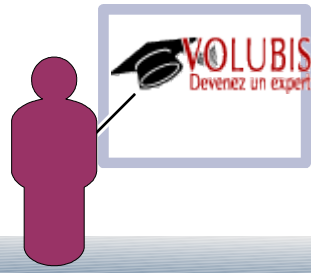
Exemple JDBCR4

```
 /copy jdbc_h

D userid          s              15A   inz('formation')
D passwrd         s              15A   inz('abc1234')

D conn            s                    like(Connection)
D ErrMsg          s              50A
D wait            s               1A
D count           s              10I 0
D rs              s                    like(ResultSet)
D nocli           s               5P 0
D nom             s              25A
D dep             s               2S 0
D datcrt          s                D
D rc              s              10I 0
D stmt            s                    like(PreparedStatement)
Drequete          S             256


 /free
```



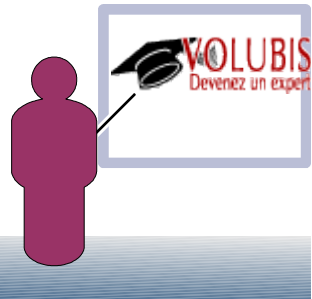
Exemple JDBCR4

```
 conn = JDBC_connect('com.mysql.jdbc.Driver'
                     :'jdbc:mysql://localhost/test'
                     : %trim(userid)
                     : %trim(passwrd) );

if conn = *null;
  dsply 'erreur de connexion';
  return;
ENDIF;

// insert de ligne
 exsr insert;

// liste des lignes
 exsr liste;

// destruction de la ligne
 exsr dlt;

jdbc_close(conn);
*inlr = *on;
```



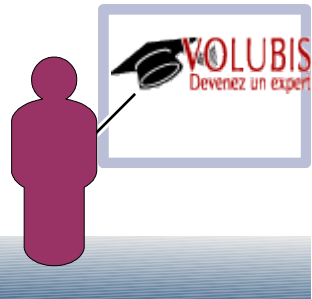
Exemple JDBCR4

```
BEGSR insert;
   stmt = JDBC_PrepStmt(conn : 'Insert Into  clients '
                             + '(nocli, nom, dep, datcrt)'
                             + ' values (?,?,?,?)' );
   if (stmt = *NULL);
     DSPLY  'prb sur PREPARE' ;
   else;

    JDBC_setInt      (stmt: 1: 999);
    JDBC_setString   (stmt: 2: 'le client 4');
    JDBC_setDecimal  (stmt: 3: 14 );
    JDBC_setDate     (stmt: 4: %date());

    rc = JDBC_ExecPrepUpd( stmt );
    if (rc < 0);
       DSPLY 'prb sur INSERT' ;
    endif;

    JDBC_FreePrepStmt( stmt );
   endif;

ENDSR;
```



Exemple JDBCR4

```
BEGSR liste;
  rs = jdbc_ExecQry( conn : 'Select nocli, nom, dep , datcrt'
                          + '  from clients                  '
                          + '  where nocli >  0               '
                          );
  if rs= *NULL;
    dsply 'erreur SQL' ;
  ELSE;
    dow (jdbc_nextRow(rs));
        nocli  = %int(jdbc_getCol(rs: 1));
        nom    = jdbc_getCol(rs: 2);
        dep    =  %dec(jdbc_getCol(rs: 3):2:0);
        datcrt = %date(jdbc_getcol(rs :4));
        except;
    enddo;
    jdbc_freeResult(rs);
  ENDIF;
ENDSR;
```



Exemple JDBCR4

```
  BEGSR dlt;
     rc = JDBC_ExecUpd( conn : 'delete from clients'
                               + ' where nocli = 999 ');
    if (rc < 0);
      DSPLY 'prb sur Delete' ;
    endif;

  ENDSR;

 /end-free


OQSYSPRT   E
O                        nocli          Z      5
O                        nom                  32
O                        dep                  70
O                        datcrt               85
```