

JDBC et Oracle



ORACLE®

Ing. Jamal BERRICH
ENSAO – Département Génie Informatique
jberrich@gmail.com
jberrich@ensa.ump.ma

Introduction :

The Oracle logo is displayed in a bold, red, sans-serif font. The word "ORACLE" is followed by a registered trademark symbol (®). The logo is centered within a white rectangular box with rounded corners and a subtle drop shadow.

ORACLE®

Eclipse : DBViewer

The screenshot displays the Eclipse DBViewer application window, titled "DBViewer - [Oracle] SCOTT.DEPT - Eclipse Platform". The interface includes a menu bar (File, Edit, Navigate, Search, Project, Run, Window, Help), a toolbar, and several panes:

- DB Tree View:** Located on the left, it shows the database structure. The "SCOTT" schema is expanded, revealing objects like SYNONYM, TABLE, BONUS, DEPT, EMP, SALGRADE, VIEW, and SEQUENCE. A red rectangle highlights the "SCOTT" schema and its sub-objects.
- SQL Execute View:** Located at the bottom right, it shows the SQL statement being executed: "scott : Oracle".
- Table View:** The main pane displays the data from the "SCOTT.DEPT" table. It includes a "Where:" clause field and a table with 4 rows and 4 columns (DEPTNO, DNAME, LOC, and an unnamed column).

The table data is as follows:

	DEPTNO	DNAME	LOC	
1	10	ACCOUNTING	NEW YORK	
2	20	RESEARCH	DALLAS	
3	30	SALES	CHICAGO	
4	40	OPERATIONS	BOSTON	

The status bar at the bottom indicates "[Oracle] SCOTT.DEPT" and "res:0.8sec".

JDBC : Les paquetages

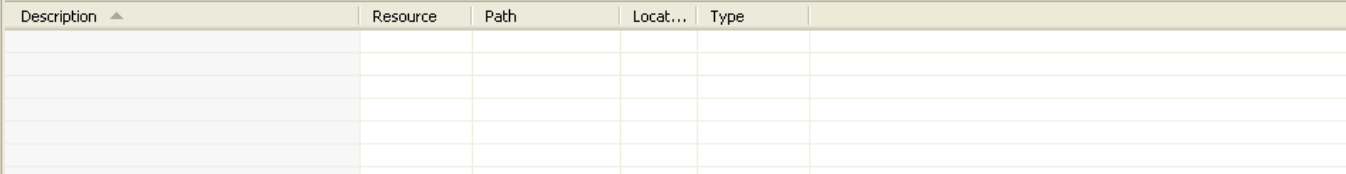
Classe / Interface	Description
java.sql.Driver java.sql.Connection	Pilotes JDBC pour les connexions aux sources de données SQL.
java.sql.Statement java.sql.PreparedStatement java.sql.CallableStatement	Construction d'ordre SQL.
java.sql.ResultSet	Gestion des résultats des requêtes SQL.
java.sql.DriverManager	Gestion des pilotes de connexion.
java.sql.SQLException	Gestion des erreurs SQL.
java.sql.DatabaseMetaData java.sql.ResultSetMetaData	Gestion des méta-informations (description de la base de données, des tables, ...).
java.sql.SavePoint	Gestion des transactions et sous-transactions.

JDBC : Les paquetages

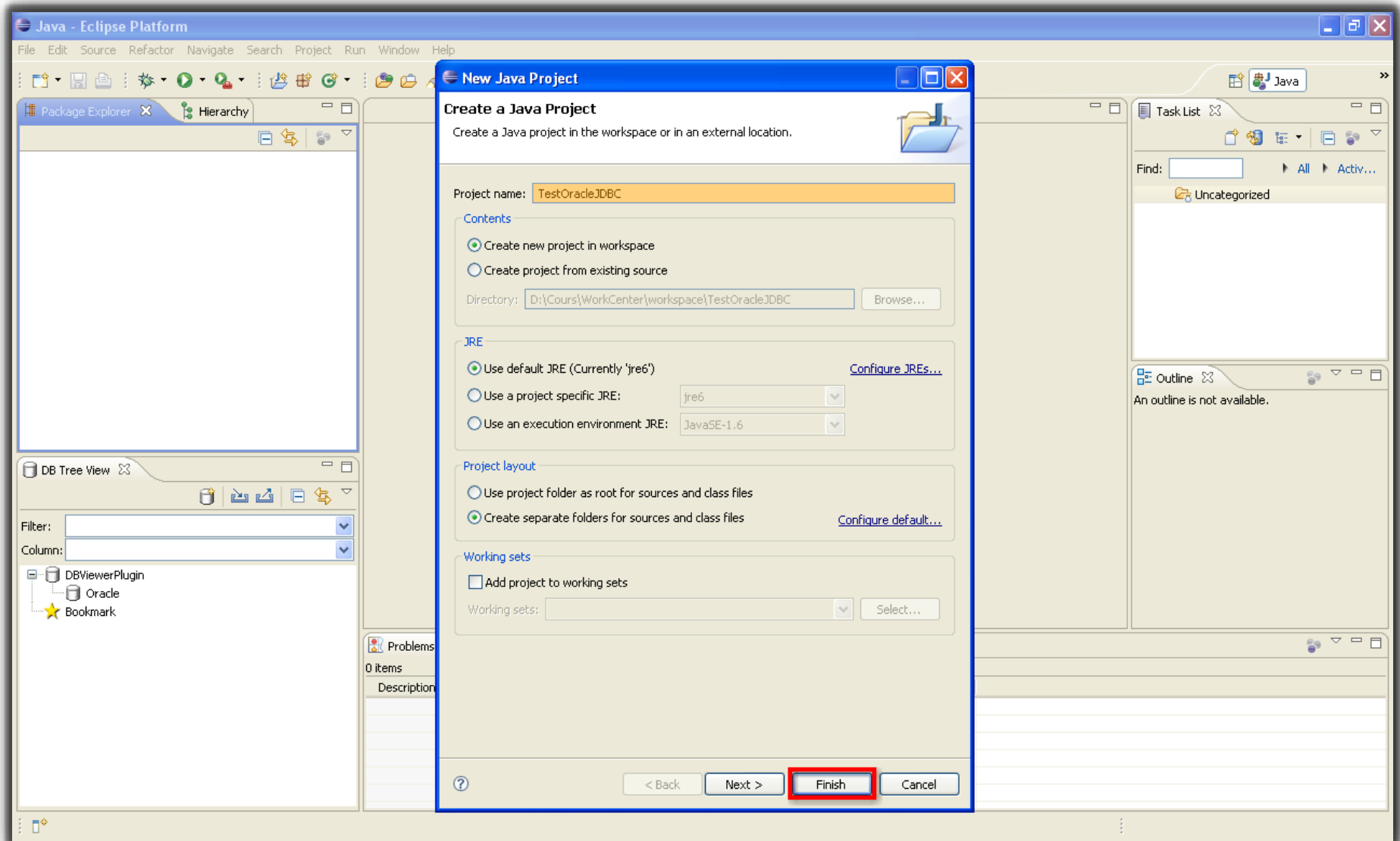
Classe / Interface	Description
<code>oracle.sql.OracleDriver</code> <code>oracle.sql.OracleConnection</code>	Connexions aux base de données (pilotes JDBC OCI et léger).
<code>oracle.sql.OracleStatement</code> <code>oracle.sql.OraclePreparedStatement</code> <code>oracle.sql.OracleCallableStatement</code>	Construction d'ordre SQL.
<code>oracle.sql.OracleResultSet</code>	Gestion des résultats des requêtes SQL.
<code>oracle.sql.OracleDriverManager</code>	Gestion des pilotes de connexion.
<code>oracle.sql.OracleSQLException</code>	Gestion des erreurs SQL.
<code>oracle.sql.OracleSavePoint</code>	Gestion des transactions et sous-transactions.

Structure d'un programme :

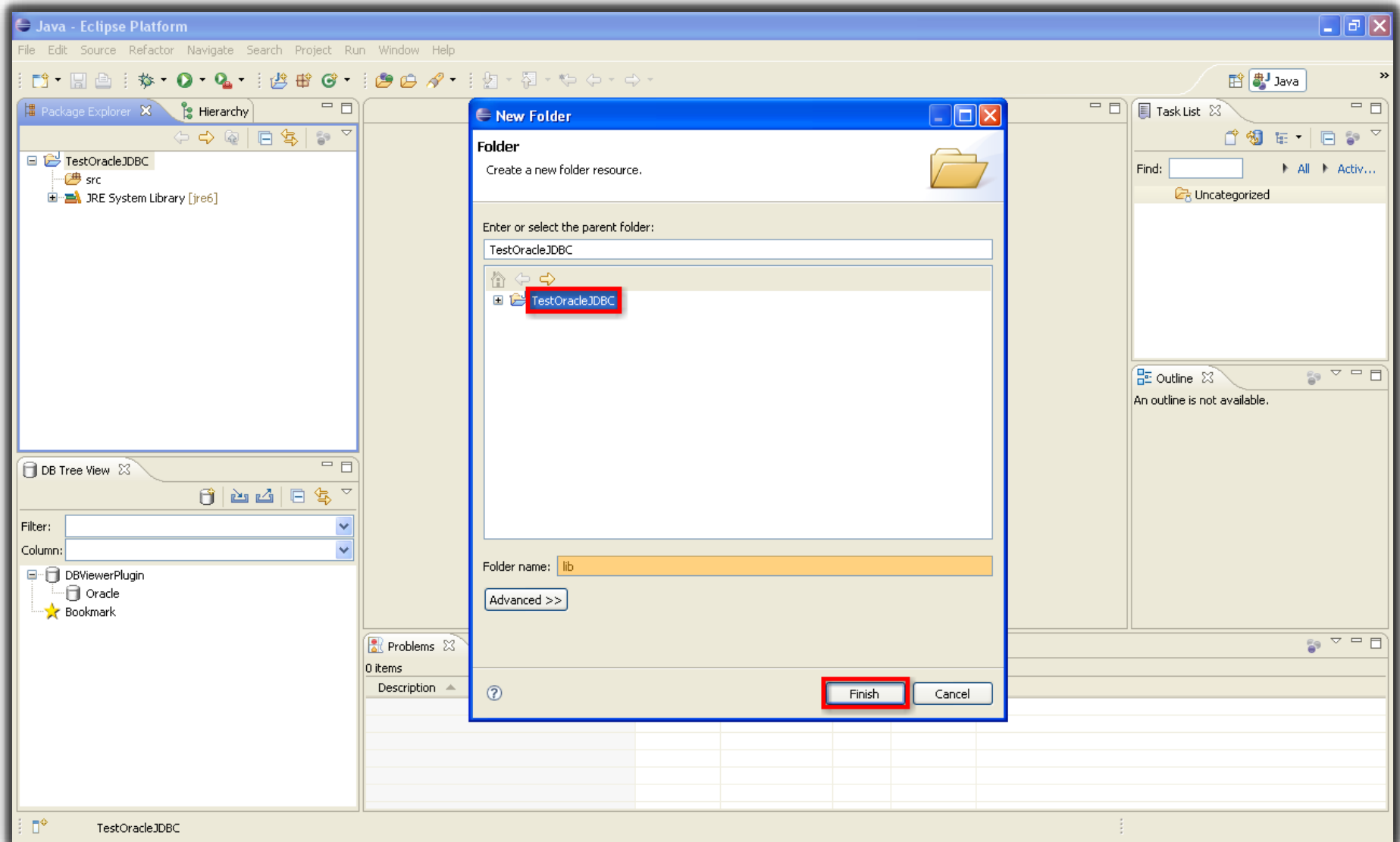
- ✓ Importation de paquetages.
- ✓ Chargement d'un pilote.
- ✓ Création d'une ou plusieurs connexions.
- ✓ Création d'un ou plusieurs états.
- ✓ Émission d'instructions SQL sur ces états.
- ✓ Fermeture des objets créés.



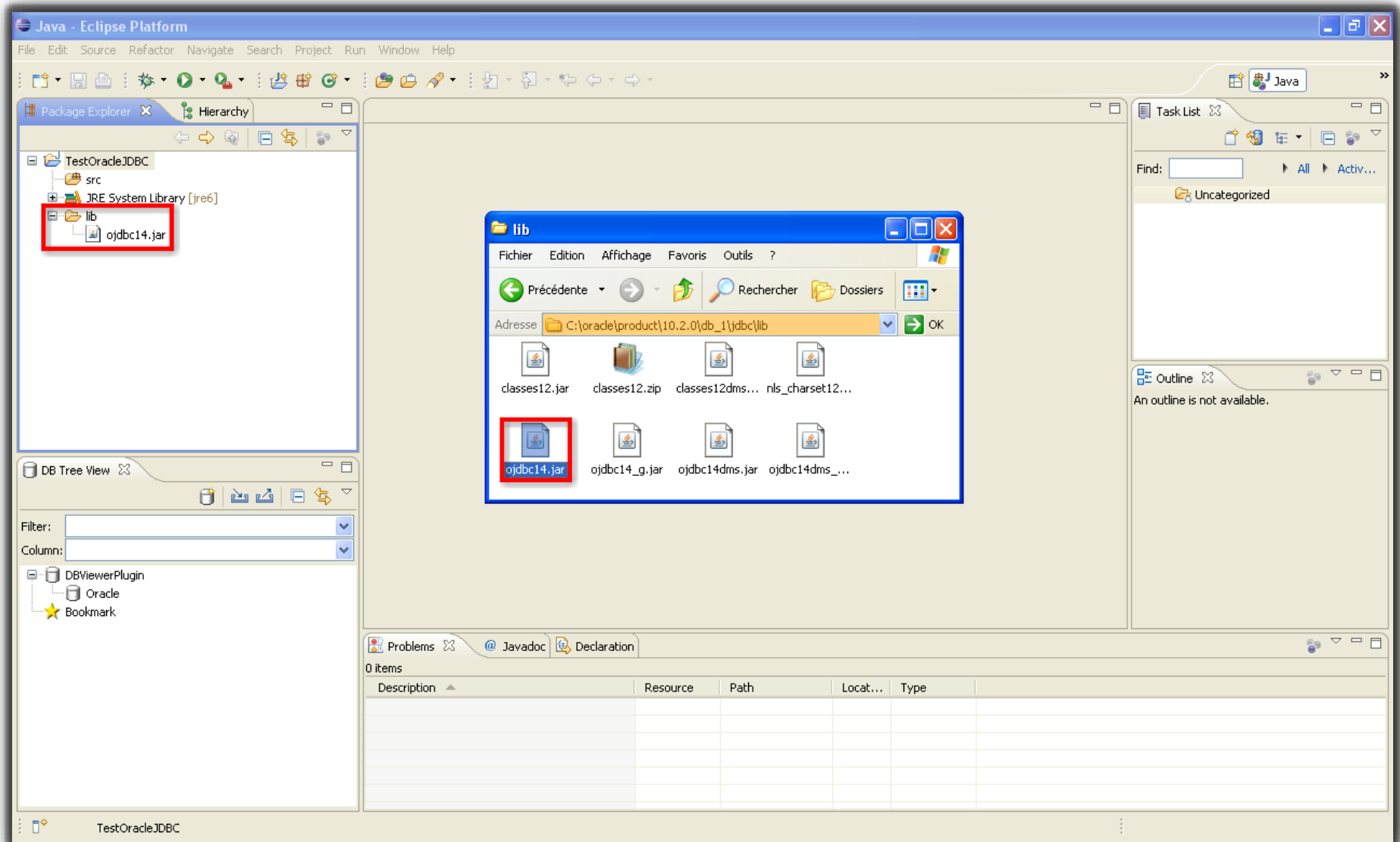
Structure d'un programme :



Structure d'un programme :

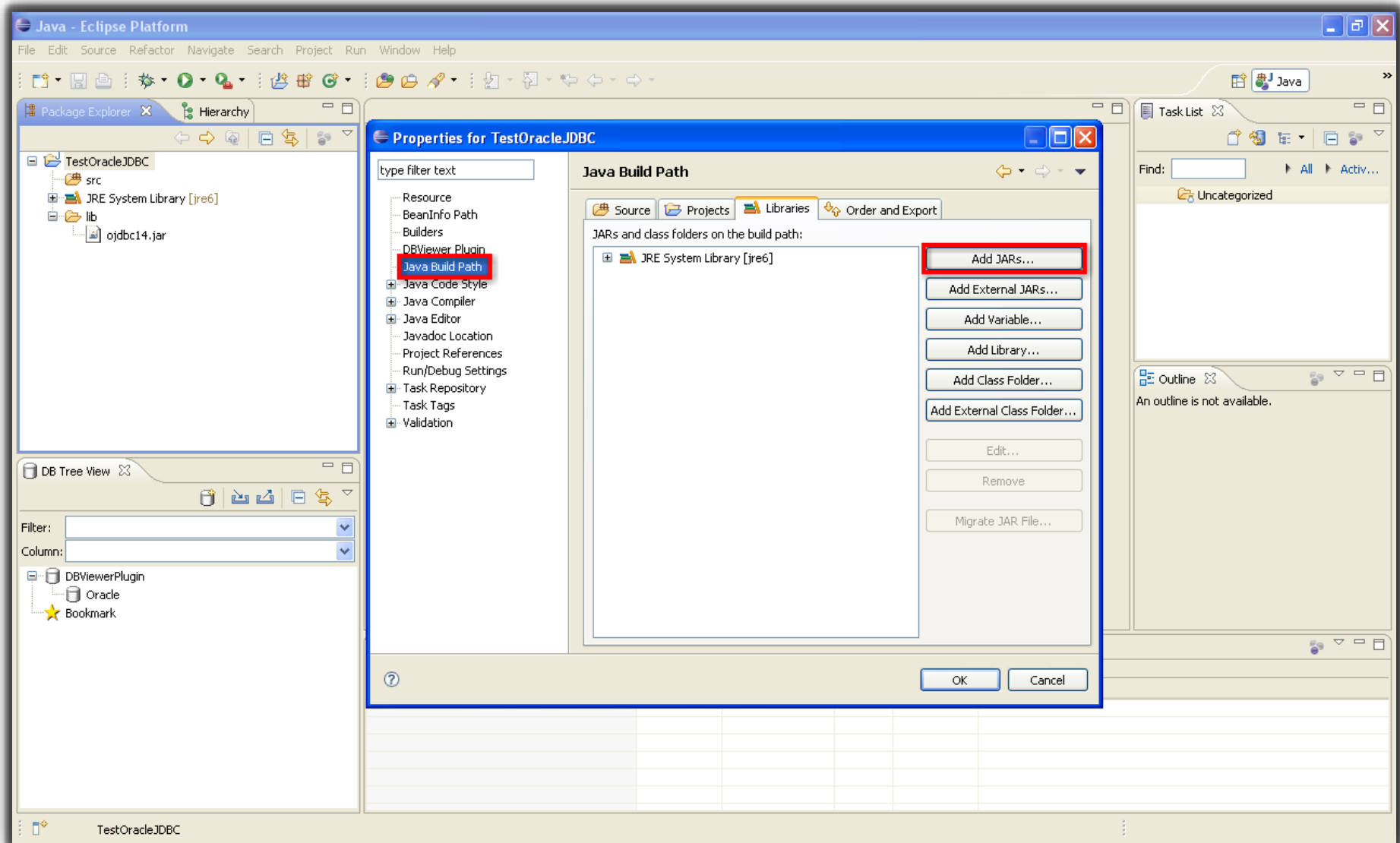


Structure d'un programme :

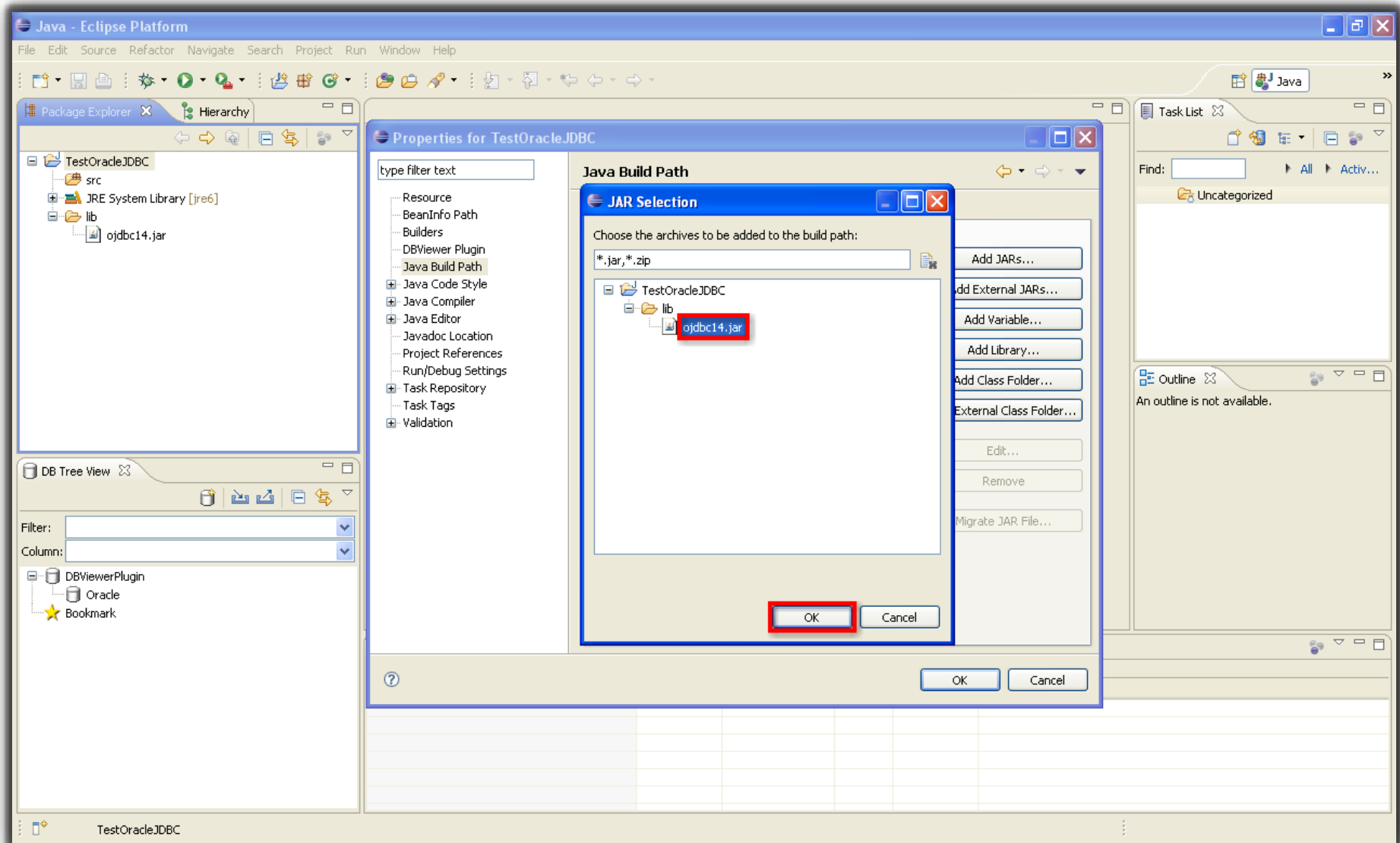




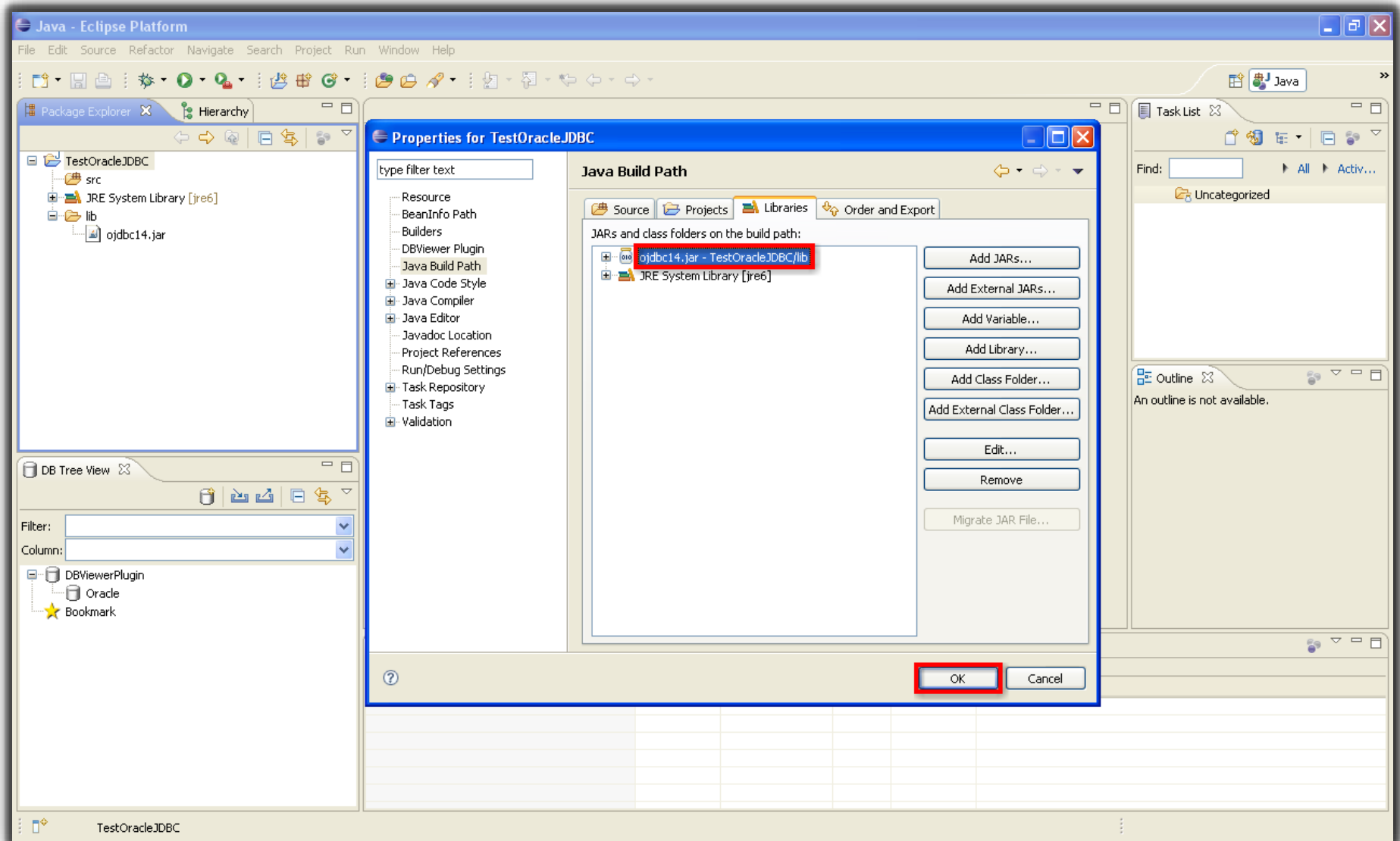
Structure d'un programme :



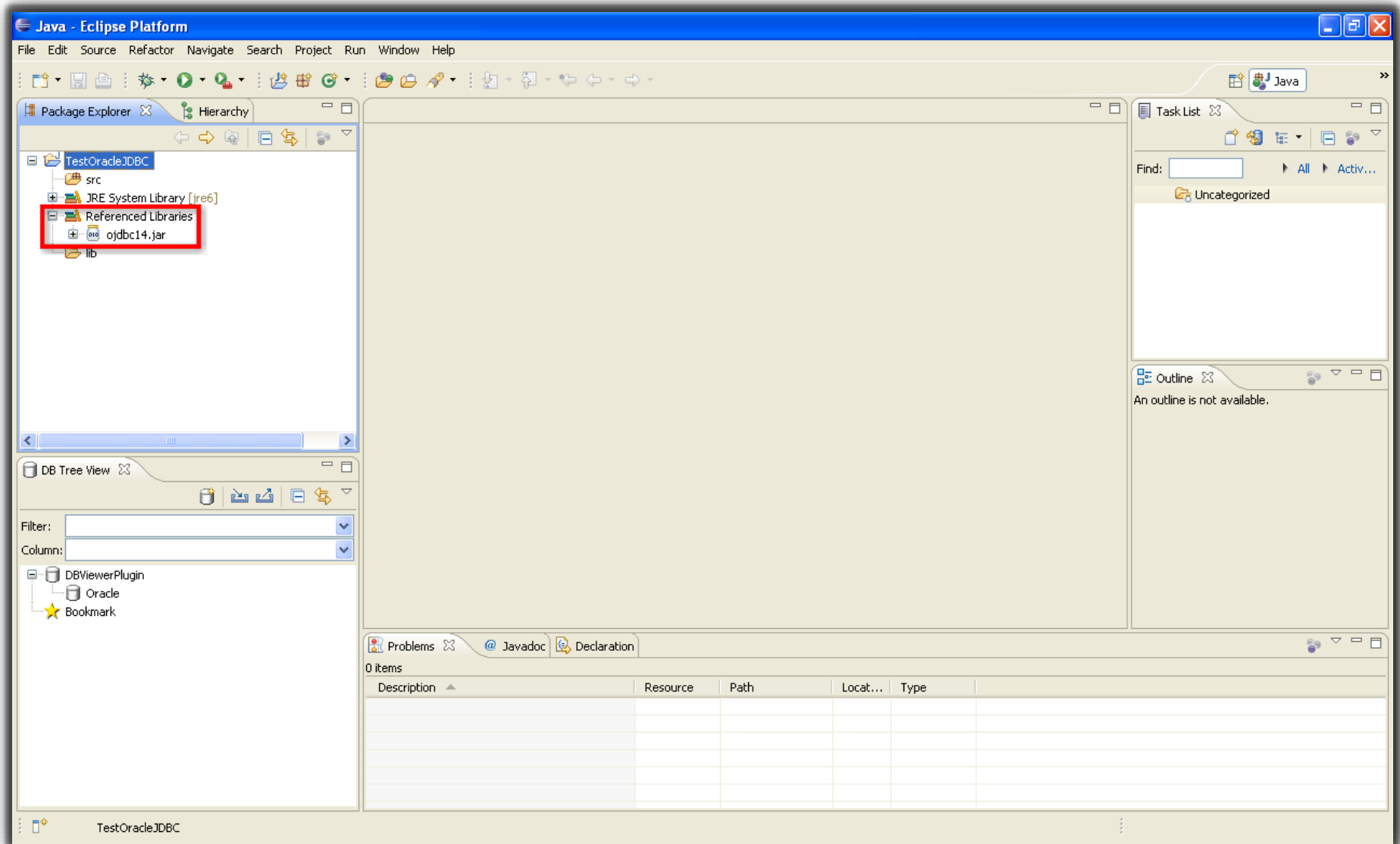
Structure d'un programme :

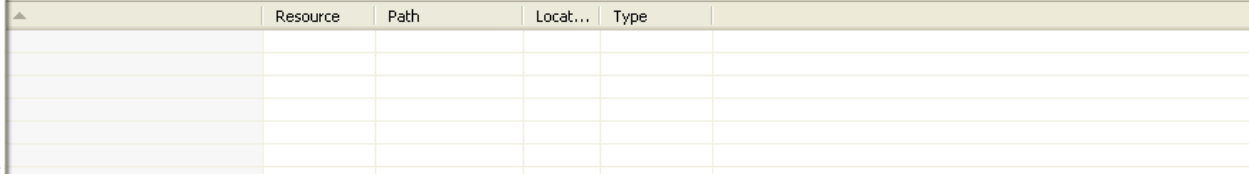


Structure d'un programme :

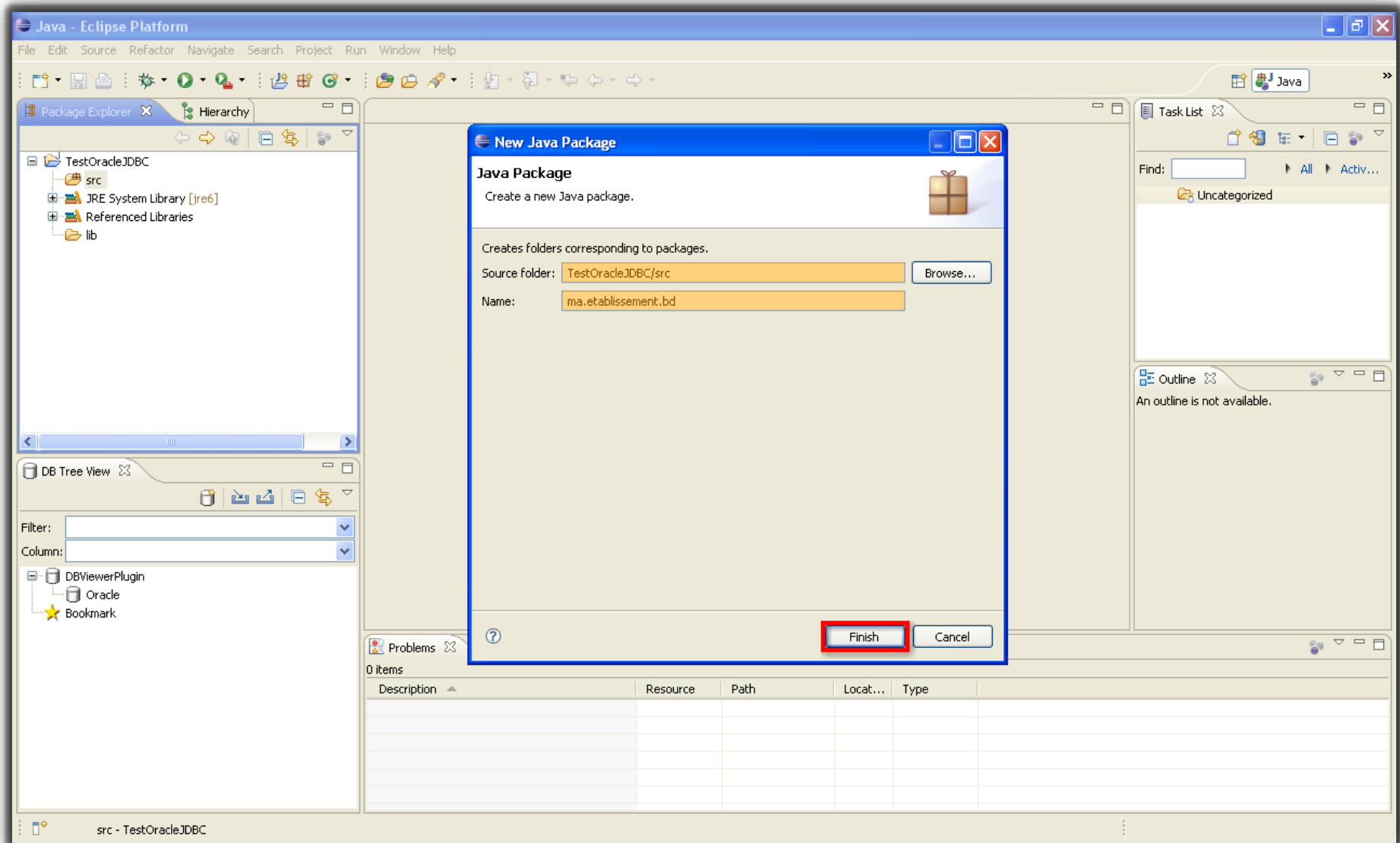


Structure d'un programme :

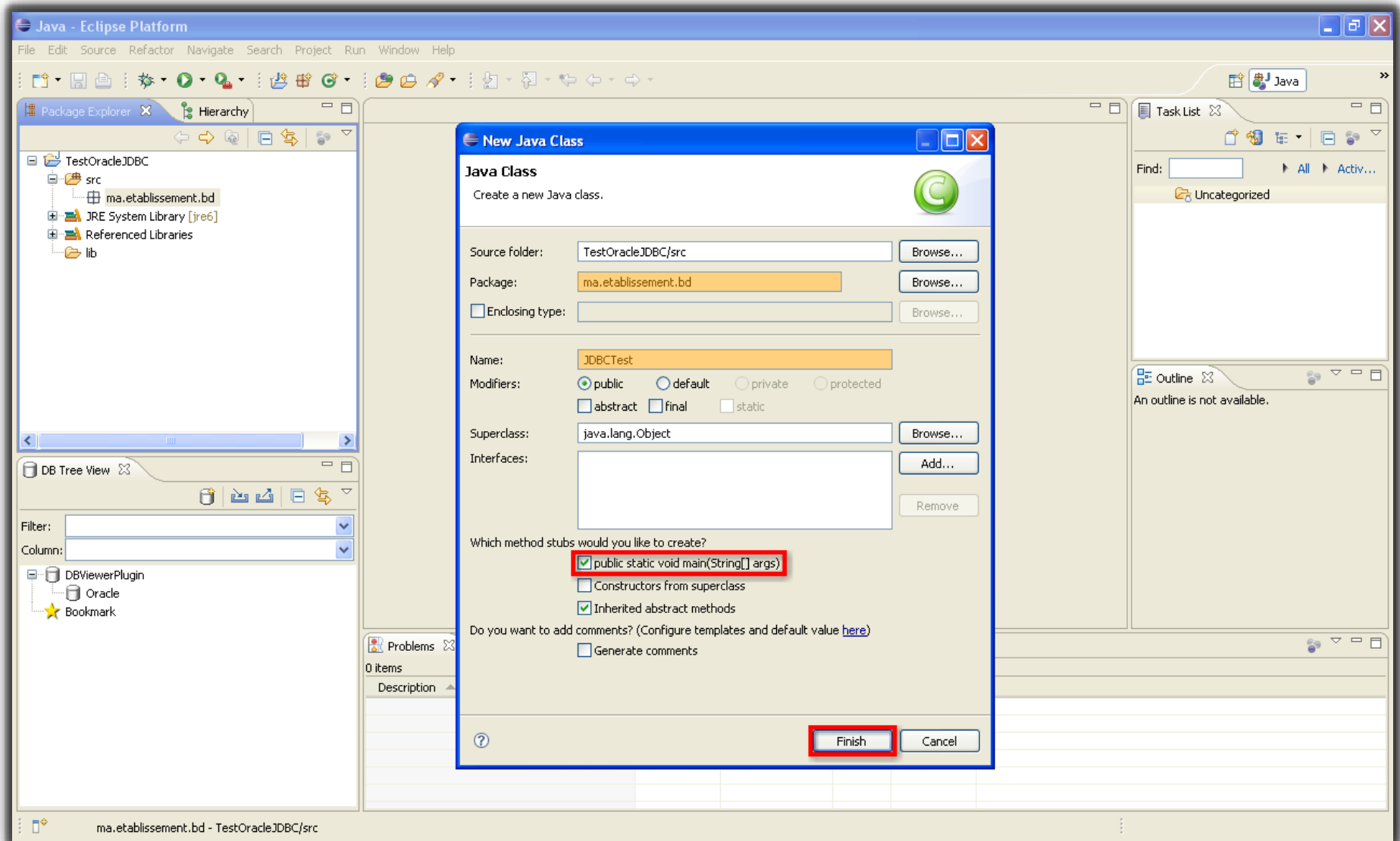




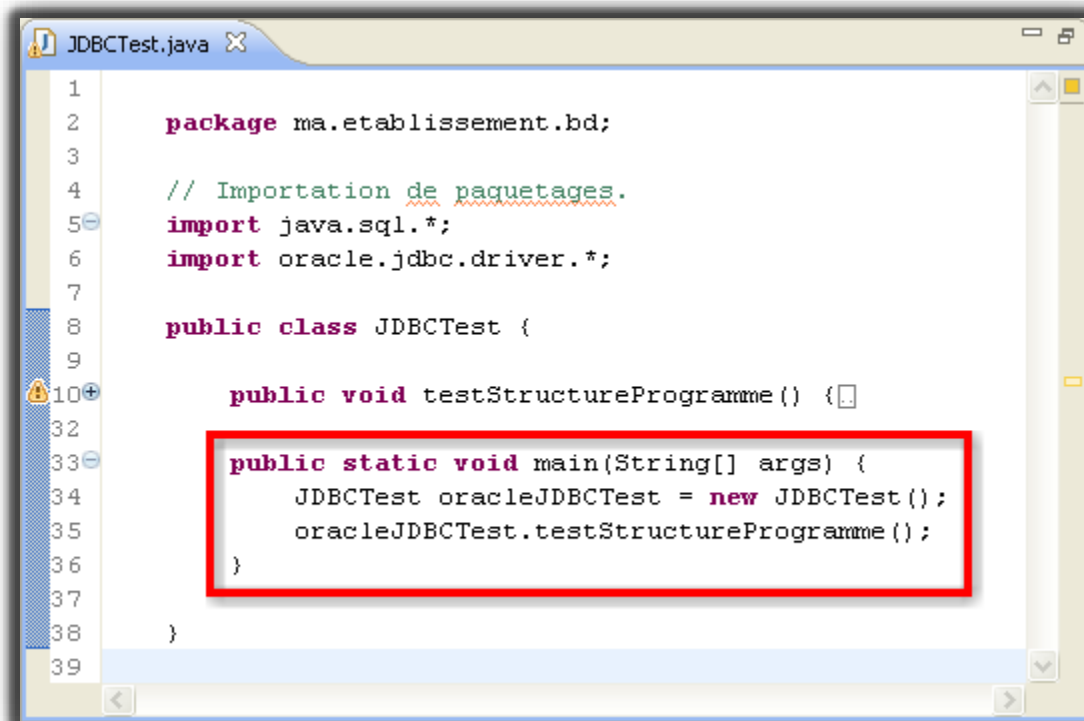
Structure d'un programme :



Structure d'un programme :

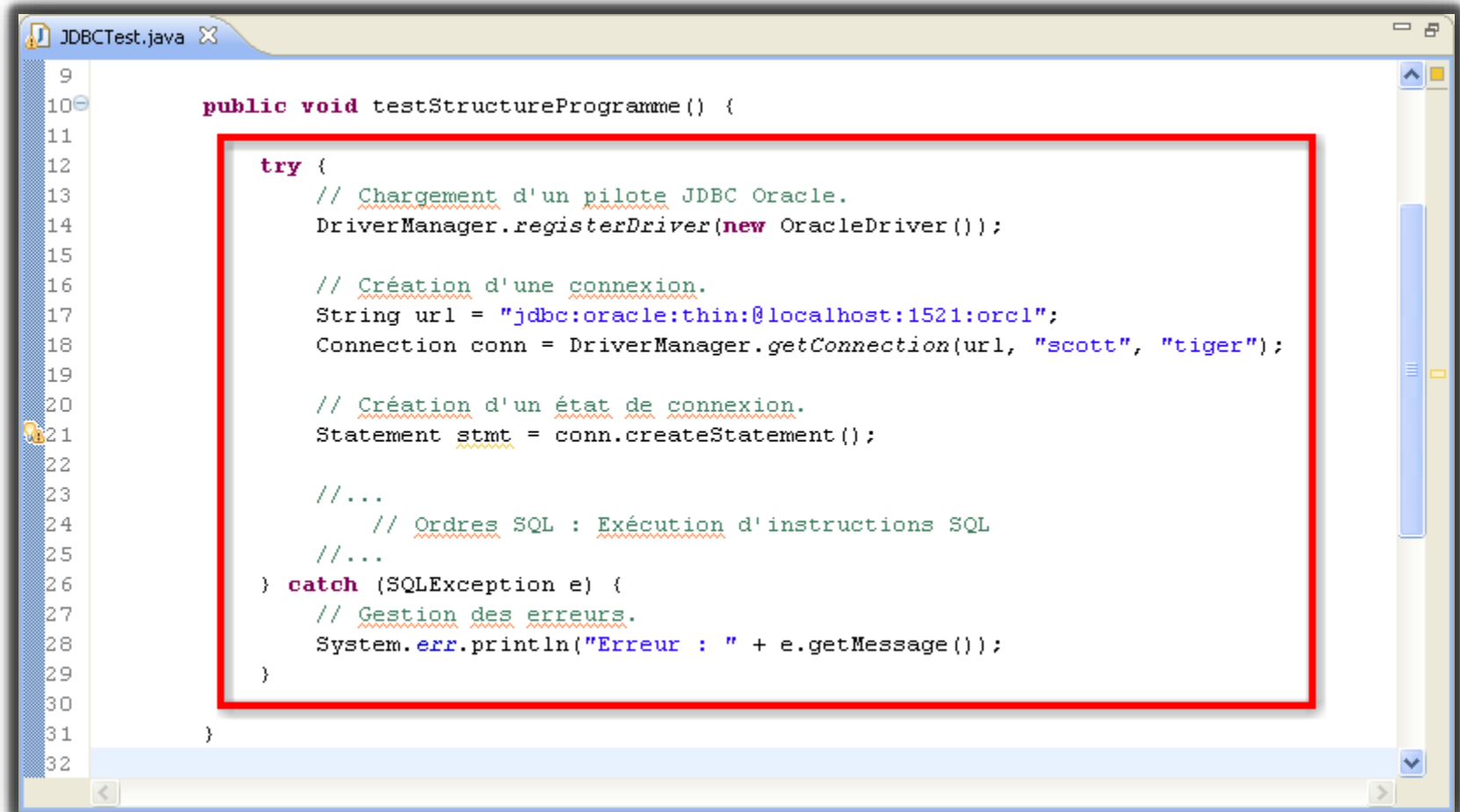


Structure d'un programme :



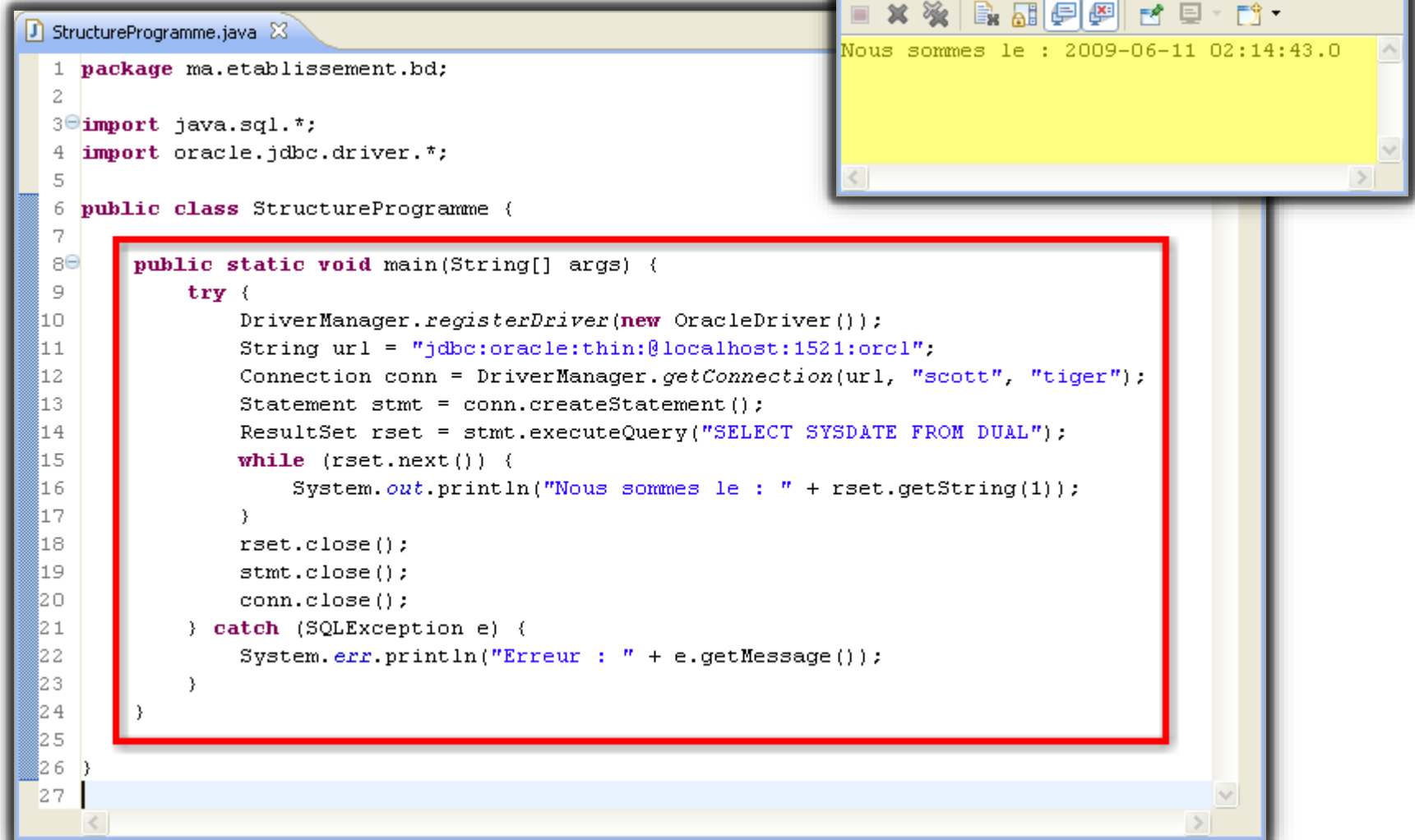
```
1
2  package ma.etablissement.bd;
3
4  // Importation de paquets.
5  import java.sql.*;
6  import oracle.jdbc.driver.*;
7
8  public class JDBCTest {
9
10     public void testStructureProgramme() {
11
12     }
13
14     public static void main(String[] args) {
15         JDBCTest oracleJDBCTest = new JDBCTest();
16         oracleJDBCTest.testStructureProgramme();
17     }
18 }
19
```

Structure d'un programme :



```
JDBCTest.java X
9
10 public void testStructureProgramme() {
11
12     try {
13         // Chargement d'un pilote JDBC Oracle.
14         DriverManager.registerDriver(new OracleDriver());
15
16         // Création d'une connexion.
17         String url = "jdbc:oracle:thin:@localhost:1521:orcl";
18         Connection conn = DriverManager.getConnection(url, "scott", "tiger");
19
20         // Création d'un état de connexion.
21         Statement stmt = conn.createStatement();
22
23         //...
24         // Ordres SQL : Exécution d'instructions SQL
25         //...
26     } catch (SQLException e) {
27         // Gestion des erreurs.
28         System.err.println("Erreur : " + e.getMessage());
29     }
30 }
31
32
```

Exemple :



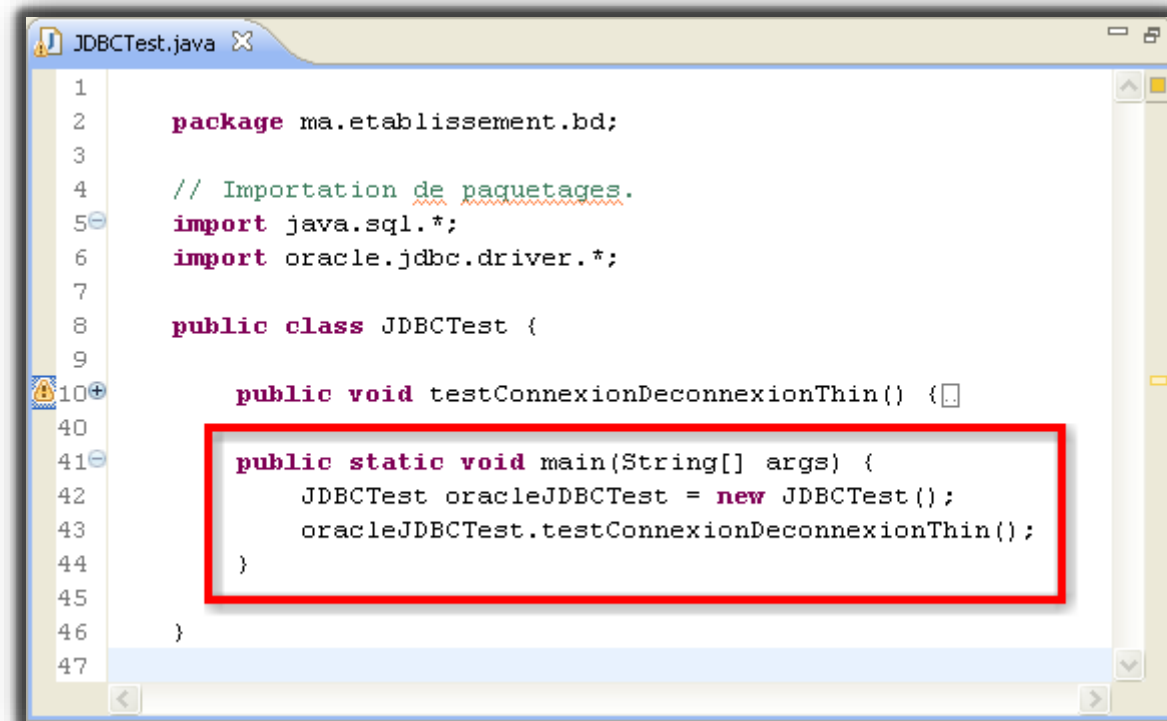
The image shows a Java IDE with two windows. The main window displays the source code for `StructureProgramme.java`. The code is as follows:

```
1 package ma.etablissement.bd;
2
3 import java.sql.*;
4 import oracle.jdbc.driver.*;
5
6 public class StructureProgramme {
7
8     public static void main(String[] args) {
9         try {
10             DriverManager.registerDriver(new OracleDriver());
11             String url = "jdbc:oracle:thin:@localhost:1521:orcl";
12             Connection conn = DriverManager.getConnection(url, "scott", "tiger");
13             Statement stmt = conn.createStatement();
14             ResultSet rset = stmt.executeQuery("SELECT SYSDATE FROM DUAL");
15             while (rset.next()) {
16                 System.out.println("Nous sommes le : " + rset.getString(1));
17             }
18             rset.close();
19             stmt.close();
20             conn.close();
21         } catch (SQLException e) {
22             System.err.println("Erreur : " + e.getMessage());
23         }
24     }
25 }
26
27
```

The code is enclosed in a red rectangular box. The second window, titled "Console", shows the output of the program:

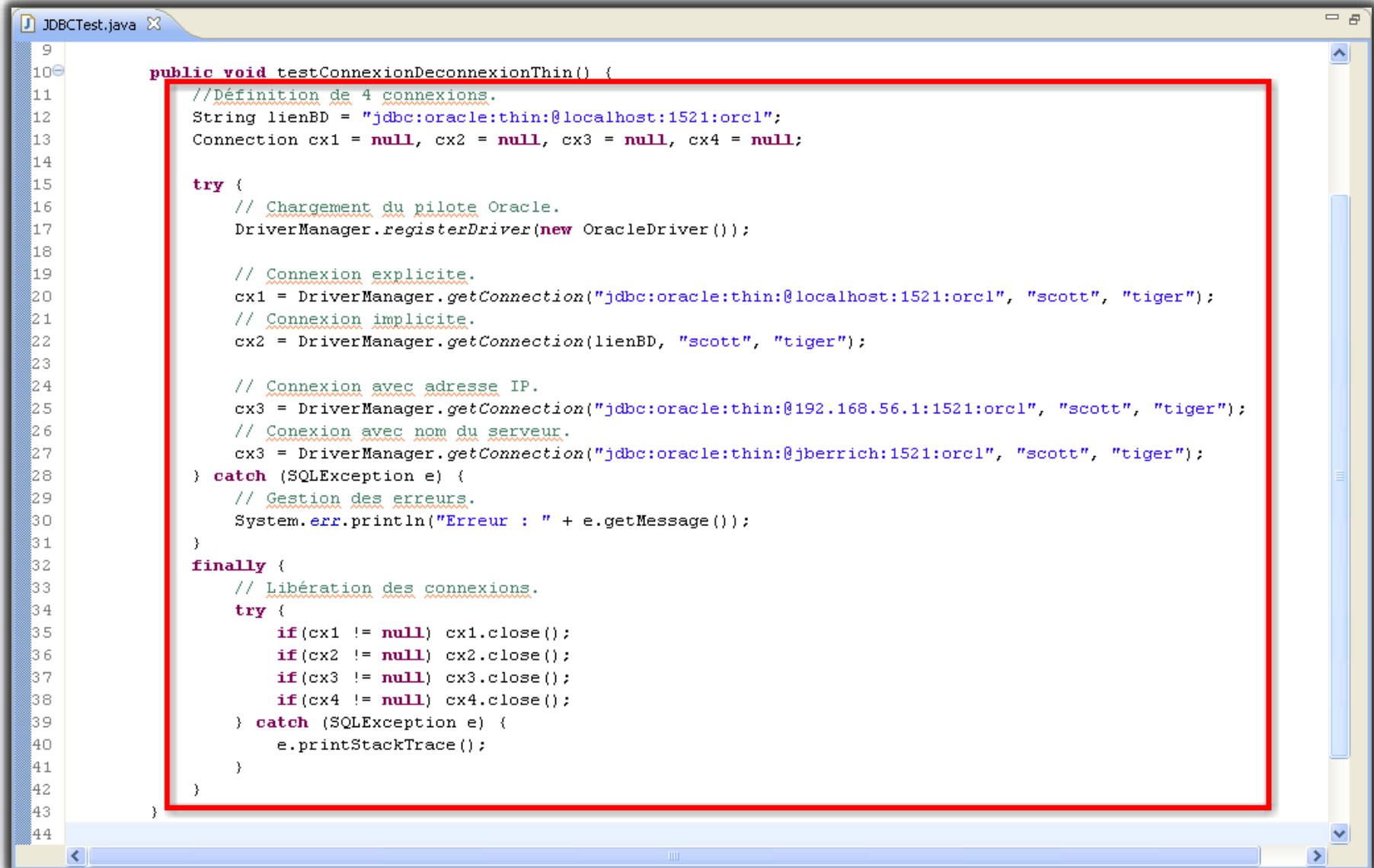
```
<terminated> StructureProgramme [Java Application] C:\Program Files\
Nous sommes le : 2009-06-11 02:14:43.0
```

Connexion / Déconnexion : thin



```
1
2  package ma.etablissement.bd;
3
4  // Importation de paquets.
5  import java.sql.*;
6  import oracle.jdbc.driver.*;
7
8  public class JDBCTest {
9
10     public void testConnexionDeconnexionThin() {
11
12         public static void main(String[] args) {
13             JDBCTest oracleJDBCTest = new JDBCTest();
14             oracleJDBCTest.testConnexionDeconnexionThin();
15         }
16     }
17 }
```


Connexion / Déconnexion : thin



```
JDBCTest.java
9
10 public void testConnexionDeconnexionThin() {
11     //Définition de 4 connexions.
12     String lienBD = "jdbc:oracle:thin:@localhost:1521:orcl";
13     Connection cx1 = null, cx2 = null, cx3 = null, cx4 = null;
14
15     try {
16         // Chargement du pilote Oracle.
17         DriverManager.registerDriver(new OracleDriver());
18
19         // Connexion explicite.
20         cx1 = DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:orcl", "scott", "tiger");
21         // Connexion implicite.
22         cx2 = DriverManager.getConnection(lienBD, "scott", "tiger");
23
24         // Connexion avec adresse IP.
25         cx3 = DriverManager.getConnection("jdbc:oracle:thin:@192.168.56.1:1521:orcl", "scott", "tiger");
26         // Connexion avec nom du serveur.
27         cx3 = DriverManager.getConnection("jdbc:oracle:thin:@jberrich:1521:orcl", "scott", "tiger");
28     } catch (SQLException e) {
29         // Gestion des erreurs.
30         System.err.println("Erreur : " + e.getMessage());
31     }
32     finally {
33         // Libération des connexions.
34         try {
35             if(cx1 != null) cx1.close();
36             if(cx2 != null) cx2.close();
37             if(cx3 != null) cx3.close();
38             if(cx4 != null) cx4.close();
39         } catch (SQLException e) {
40             e.printStackTrace();
41         }
42     }
43 }
44
```

Exemple :

```
ConnexionDeconnexionThin.java
1 package ma.etablissement.bd;
2
3 import java.sql.*;
4 import oracle.jdbc.driver.*;
5
6 public class ConnexionDeconnexionThin {
7
8     public ConnexionDeconnexionThin() {
9         try {
10             DriverManager.registerDriver(new OracleDriver());
11         } catch (SQLException e) {
12             e.printStackTrace();
13         }
14     }
15
16     public void oracleConnexionExplicite() {}
17
18     public void oracleConnexionImplicite() {}
19
20     public void oracleConnexionAvecAdresseIP() {}
21
22     public void oracleConnexionAvecNomDuServeur() {}
23
24     public static void main(String[] args) throws InterruptedException {
25         ConnexionDeconnexionThin oracleConnexionDeconnexionThin = new ConnexionDeconnexionThin();
26         oracleConnexionDeconnexionThin.oracleConnexionExplicite();
27         Thread.sleep(1000);
28         oracleConnexionDeconnexionThin.oracleConnexionImplicite();
29         Thread.sleep(1000);
30         oracleConnexionDeconnexionThin.oracleConnexionAvecAdresseIP();
31         Thread.sleep(1000);
32         oracleConnexionDeconnexionThin.oracleConnexionAvecNomDuServeur();
33     }
34 }
35
```

Exemple :



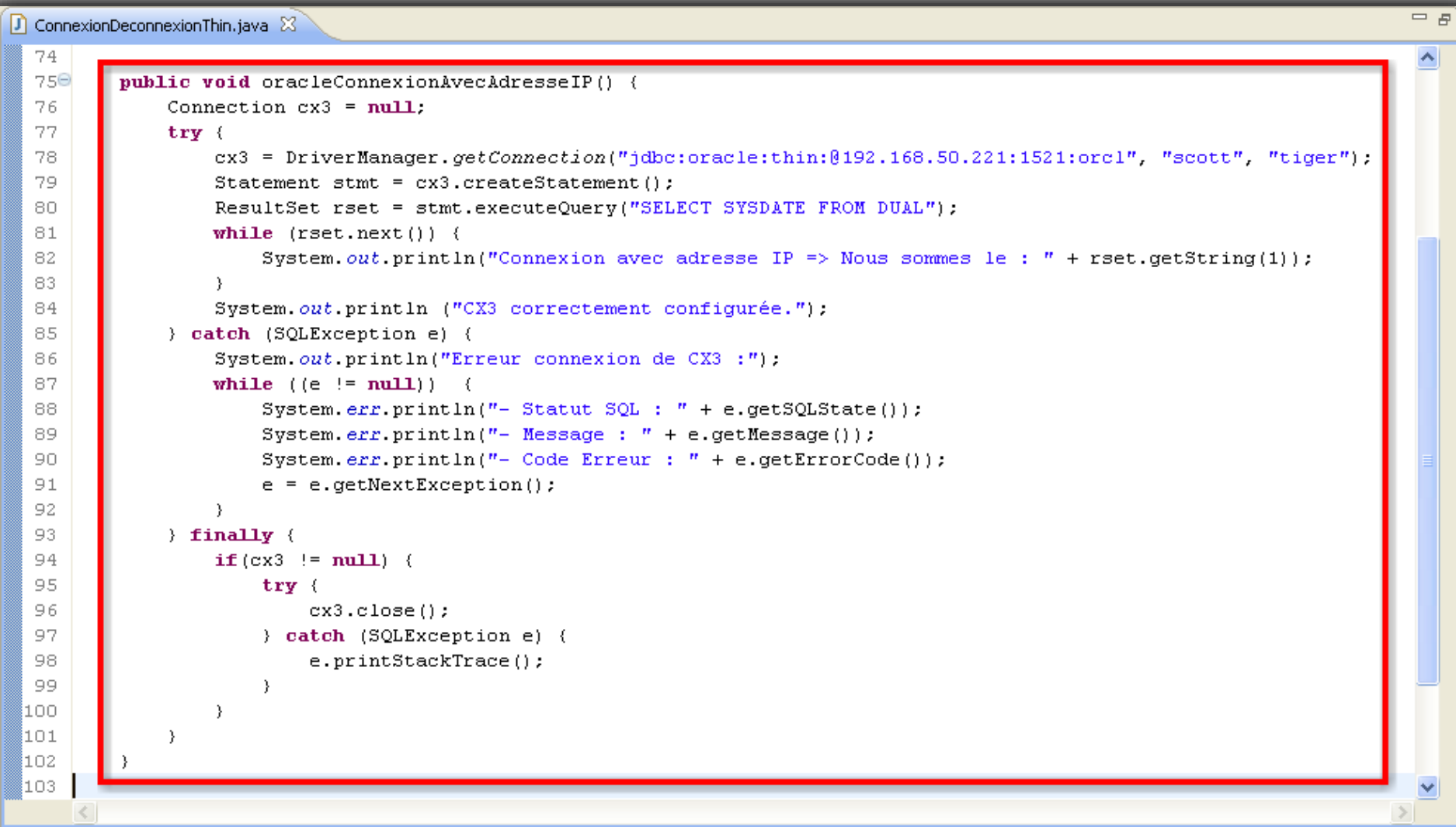
```
15
16 public void oracleConnexionExplicite() {
17     Connection cx1 = null;
18     try {
19         cx1 = DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:orcl","scott","tiger");
20         Statement stmt = cx1.createStatement();
21         ResultSet rset = stmt.executeQuery("SELECT SYSDATE FROM DUAL");
22         while (rset.next()) {
23             System.out.println("Connexion explicite => Nous sommes le : " + rset.getString(1));
24         }
25         System.out.println ("CX1 correctement configurée.");
26     } catch (SQLException e) {
27         System.out.println("Erreur connexion de CX1 :");
28         while ((e != null)) {
29             System.err.println("- Statut SQL : " + e.getSQLState());
30             System.err.println("- Message : " + e.getMessage());
31             System.err.println("- Code Erreur : " + e.getErrorCode());
32             e = e.getNextException();
33         }
34     } finally {
35         if(cx1 != null) {
36             try {
37                 cx1.close();
38             } catch (SQLException e) {
39                 e.printStackTrace();
40             }
41         }
42     }
43 }
44 }
```

Exemple :

ConnexionDeconnexionThin.java

```
44
45 public void oracleConnexionImplicite() {
46     String lienBD = "jdbc:oracle:thin:@localhost:1521:orcl";
47     Connection cx2 = null;
48     try {
49         cx2 = DriverManager.getConnection(lienBD, "scott", "tiger");
50         Statement stmt = cx2.createStatement();
51         ResultSet rset = stmt.executeQuery("SELECT SYSDATE FROM DUAL");
52         while (rset.next()) {
53             System.out.println("Connexion implicite => Nous sommes le : " + rset.getString(1));
54         }
55         System.out.println("CX2 correctement configurée.");
56     } catch (SQLException e) {
57         System.out.println("Erreur connexion de CX2 :");
58         while ((e != null)) {
59             System.err.println("- Statut SQL : " + e.getSQLState());
60             System.err.println("- Message : " + e.getMessage());
61             System.err.println("- Code Erreur : " + e.getErrorCode());
62             e = e.getNextException();
63         }
64     } finally {
65         if (cx2 != null) {
66             try {
67                 cx2.close();
68             } catch (SQLException e) {
69                 e.printStackTrace();
70             }
71         }
72     }
73 }
74
```

Exemple :



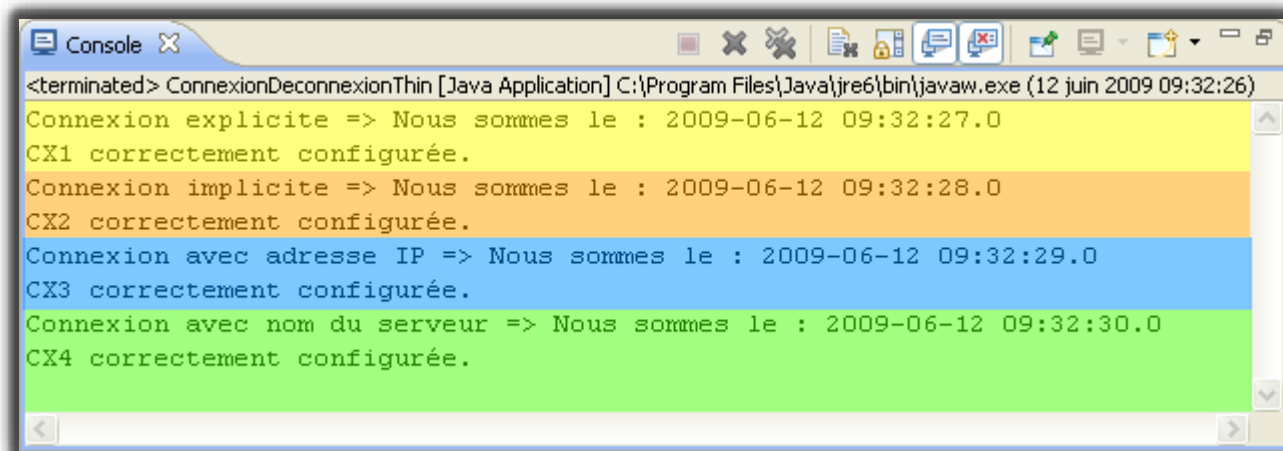
```
74
75 public void oracleConnexionAvecAdresseIP() {
76     Connection cx3 = null;
77     try {
78         cx3 = DriverManager.getConnection("jdbc:oracle:thin:@192.168.50.221:1521:orcl", "scott", "tiger");
79         Statement stmt = cx3.createStatement();
80         ResultSet rset = stmt.executeQuery("SELECT SYSDATE FROM DUAL");
81         while (rset.next()) {
82             System.out.println("Connexion avec adresse IP => Nous sommes le : " + rset.getString(1));
83         }
84         System.out.println("CX3 correctement configurée.");
85     } catch (SQLException e) {
86         System.out.println("Erreur connexion de CX3 :");
87         while ((e != null)) {
88             System.err.println("- Statut SQL : " + e.getSQLState());
89             System.err.println("- Message : " + e.getMessage());
90             System.err.println("- Code Erreur : " + e.getErrorCode());
91             e = e.getNextException();
92         }
93     } finally {
94         if(cx3 != null) {
95             try {
96                 cx3.close();
97             } catch (SQLException e) {
98                 e.printStackTrace();
99             }
100         }
101     }
102 }
103 }
```

Exemple :

ConnexionDeconnexionThin.java

```
103
104 public void oracleConnexionAvecNomDuServeur() {
105     Connection cx4 = null;
106     try {
107         cx4 = DriverManager.getConnection("jdbc:oracle:thin:@jberrich:1521:orcl", "scott", "tiger");
108         Statement stmt = cx4.createStatement();
109         ResultSet rset = stmt.executeQuery("SELECT SYSDATE FROM DUAL");
110         while (rset.next()) {
111             System.out.println("Connexion avec nom du serveur => Nous sommes le : " + rset.getString(1));
112         }
113         System.out.println("CX4 correctement configurée.");
114     } catch (SQLException e) {
115         System.out.println("Erreur connexion de CX4 :");
116         while ((e != null)) {
117             System.err.println("- Statut SQL : " + e.getSQLState());
118             System.err.println("- Message : " + e.getMessage());
119             System.err.println("- Code Erreur : " + e.getErrorCode());
120             e = e.getNextException();
121         }
122     } finally {
123         if (cx4 != null) {
124             try {
125                 cx4.close();
126             } catch (SQLException e) {
127                 e.printStackTrace();
128             }
129         }
130     }
131 }
132
```

Exemple :



A screenshot of a Java console window titled "Console". The window displays the output of a Java application, showing four successful connection attempts with timestamps. The text is color-coded: yellow for the first two lines, orange for the third, blue for the fourth, and green for the fifth. The window has a standard Windows-style title bar and a toolbar with icons for file operations and window management.

```
<terminated> ConnexionDeconnexionThin [Java Application] C:\Program Files\Java\jre6\bin\javaw.exe (12 juin 2009 09:32:26)
Connexion explicite => Nous sommes le : 2009-06-12 09:32:27.0
CX1 correctement configurée.
Connexion implicite => Nous sommes le : 2009-06-12 09:32:28.0
CX2 correctement configurée.
Connexion avec adresse IP => Nous sommes le : 2009-06-12 09:32:29.0
CX3 correctement configurée.
Connexion avec nom du serveur => Nous sommes le : 2009-06-12 09:32:30.0
CX4 correctement configurée.
```

Example :

```
cx1 = DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:orcl","NULL","tiger");
```

```
ResultSet rset = stmt.executeQuery("DELECT FROM DUAL");
```

```
cx3 = DriverManager.getConnection("jdbc:oracle:thin:@192.168.50.222:1521:orcl", "scott", "tiger");
```

```
cx4 = DriverManager.getConnection("jdbc:oracle:thin:@jberrich:1521:NULL", "scott", "tiger");
```


Exemple :



The screenshot shows a Java console window titled "Console" with a standard Windows-style title bar. The window contains the output of a Java application, specifically a series of database connection attempts and their failures. The text is color-coded: yellow for the first error, orange for the second, blue for the third, and green for the fourth. The errors are as follows:

```
<terminated> ConnexionDeconnexionThin [Java Application] C:\Program Files\Java\jre6\bin\javaw.exe (12 juin 2009 15:39:53)
Erreur connexion de CX1 :
- Statut SQL : 72000
- Message : ORA-01017: invalid username/password; logon denied
- Code Erreur : 1017
Erreur connexion de CX2 :
- Statut SQL : 42000
- Message : ORA-00900: instruction SQL non valide
- Code Erreur : 900
Erreur connexion de CX3 :
- Statut SQL : null
- Message : Exception d'E/S: The Network Adapter could not establish the connection
- Code Erreur : 17002
Erreur connexion de CX4 :
- Statut SQL : null
- Message : Listener refused the connection with the following error:
ORA-12505, TNS:listener does not currently know of SID given in connect descriptor
The Connection descriptor used by the client was:
jberrich:1521:NULL
- Code Erreur : 0
```

Interface Connection :

Méthode	Description
<code>createStatement()</code>	Création d'un objet destiné à recevoir un ordre SQL statique non paramétré.
<code>prepareStatement(String)</code>	Précompile un ordre SQL acceptant des paramètres et pouvant être exécuté plusieurs fois.
<code>prepareCall(String)</code>	Appel d'une procédure cataloguée (certains pilotes attendent <i>execute</i> ou ne supportent pas <i>prepareCall</i>).
<code>void setAutoCommit(boolean)</code>	Positionne ou non le <i>commit</i> automatique.
<code>void commit()</code>	Valide la transaction.
<code>void rollback()</code>	Invalide la transaction.
<code>void close()</code>	Ferme la connexion.

État d'une connexion :

Classe / Interface	Description
Statement	Pour les ordres SQL statiques. Ces états sont construits par la méthode <i>createStatement</i> appliquée à la connexion.
PreparedStatement	Pour les SQL paramétrés. Ces états sont construits par la méthode <i>prepareStatement</i> appliquée à la connexion.
CallableStatement	Pour les procédures et fonctions cataloguées (Java, C, PL/SQL, etc.). Ces états sont construits par la méthode <i>callableStatement</i> appliquée à la connexion.

Interface Statement :

Méthode	Description
<code>ResultSet executeQuery(String)</code>	Exécute une requête et retourne un ensemble de lignes (<i>ResultSet</i>).
<code>int executeUpdate(String)</code>	Exécute une instruction SQL et retourne le nombre de lignes traitées (<i>INSERT</i> , <i>UPDATE</i> Ou <i>DELETE</i>) ou 0 pour les instructions ne retournant aucun résultat.
<code>boolean execute(String)</code>	Exécute une instruction SQL et renvoie <i>true</i> si c'est une instruction <i>SELECT</i> , <i>false</i> sinon.
<code>Connection getConnection()</code>	Retourne l'objet de connexion.
<code>void setMaxRows(int)</code>	Positionne la limite du nombre d'enregistrements à extraire par toute requête issue de cet état.
<code>int getUpdateCount()</code>	Nombre de lignes traitées par l'instruction SQL. (-1 si c'est une requête ou si l'instruction n'affecte aucune ligne).
<code>void close()</code>	Ferme l'état.

Exemple :

```
EtatsSimples.java X
1 package ma.etablissement.bd;
2
3 import java.sql.*;
4 import oracle.jdbc.driver.*;
5
6 public class EtatsSimples {
7
8     public static void main(String[] args) {
9         Connection cx = null;
10        try {
11            DriverManager.registerDriver(new OracleDriver());
12            cx = DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:orcl","scott","tiger");
13            String ordreLDD;
14            String ordreLMD;
15
16            // Création de l'état.
17            Statement etatSimple = cx.createStatement();
18
19            // Ordre LDD.
20            ordreLDD = "CREATE TABLE Compagnie(comp VARCHAR(4), " +
21                    "nomComp VARCHAR(30), " +
22                    "CONSTRAINT pk_Compagnie PRIMARY KEY(comp))";
23
24            etatSimple.execute(ordreLDD);
25            //Ordre LDD (autre écriture), j contient 0 (aucune ligne n'est concernée).
26            ordreLDD = "CREATE TABLE Avion (immat VARCHAR(6), " +
27                    "typeAvion VARCHAR(15), " +
28                    "cap NUMBER(3), " +
29                    "compa VARCHAR(4), " +
30                    "CONSTRAINT pk_Avion PRIMARY KEY(immat), " +
31                    "CONSTRAINT fk_Avion_comp_Compagnie FOREIGN KEY(compa) REFERENCES Compagnie(comp))";
32
33            int j = etatSimple.executeUpdate(ordreLDD);
34            System.out.println ("Nombre de lignes concernées = " + j);
35        }
36    }
37 }
```

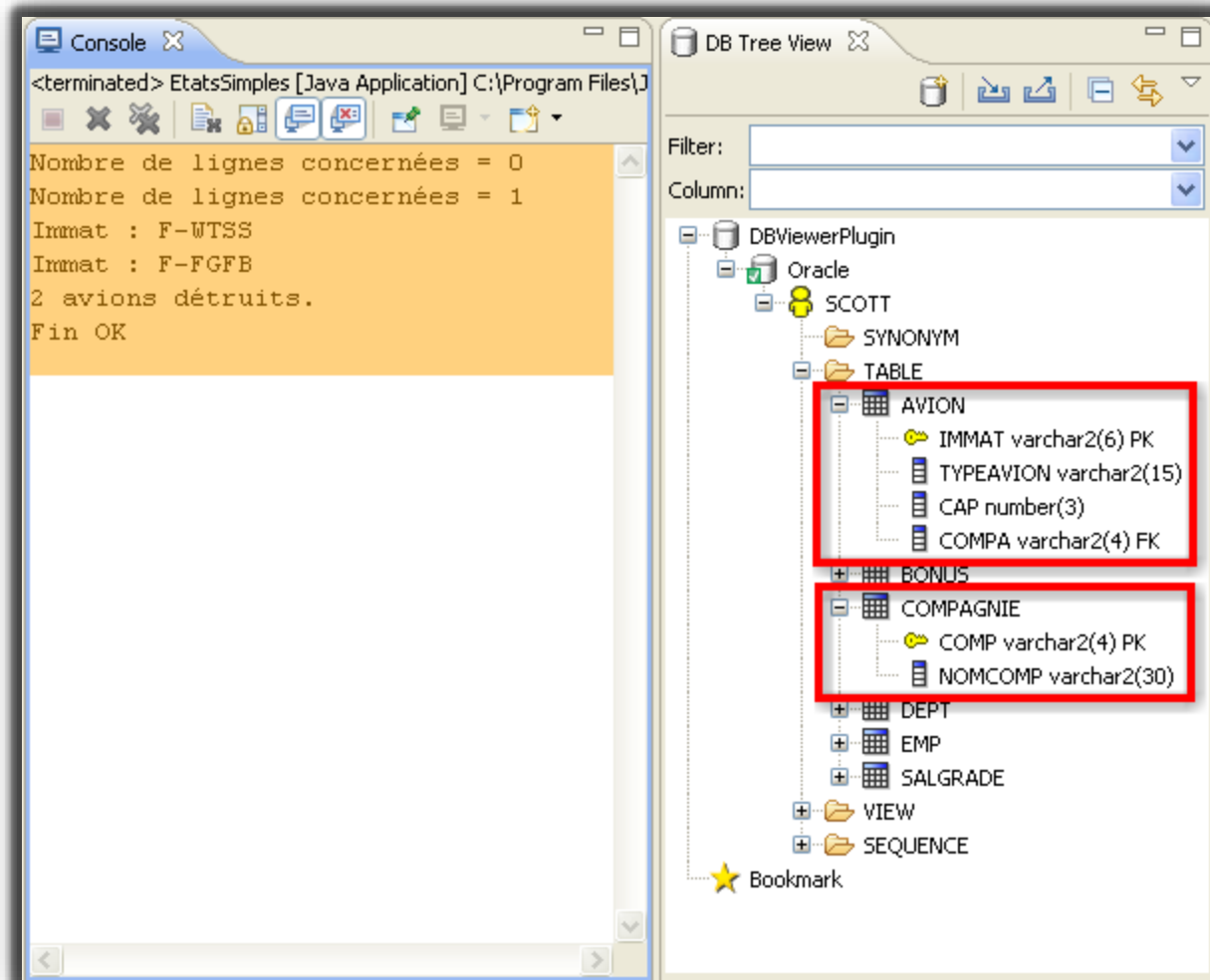
Exemple :

```
33
34 // Ordre LMD, k contient 1 (une ligne est concernée).
35 ordreLMD = "INSERT INTO Compagnie VALUES ('AF', 'Air France')";
36 int k = etatSimple.executeUpdate(ordreLMD);
37 System.out.println ("Nombre de lignes concernées = " + k);
38 // Ordre LMD (autres écritures).
39 ordreLMD = "INSERT INTO Avion VALUES ('F-WTSS', 'Concorde', 90, 'AF')";
40 etatSimple.execute(ordreLMD);
41 ordreLMD = "INSERT INTO Avion VALUES ('F-FGFB', 'A320', 148, 'AF')";
42 etatSimple.execute(ordreLMD);
43
44 // Pas plus de 10 lignes retournées.
45 etatSimple.setMaxRows(10);
46
47 // Chargement d'un curseur Java
48 ResultSet curseurJava = etatSimple.executeQuery("SELECT * FROM Avion");
49
50 while (curseurJava.next ())
51     System.out.println ("Immat : " + curseurJava.getString(1));
52
```

Exemple :

```
EtatsSimples.java
53 // Ordre LMD, contient 2 (avions supprimés).
54 etatSimple.execute("DELETE FROM Avion");
55 int l = etatSimple.getUpdateCount();
56 System.out.println(l + " avions détruits.");
57
58 System.out.println ("Fin OK");
59 } catch (SQLException e) {
60     System.out.println("Erreur connexion :");
61     while ((e != null)) {
62         System.err.println("- Statut SQL : " + e.getSQLState());
63         System.err.println("- Message : " + e.getMessage());
64         System.err.println("- Code Erreur : " + e.getErrorCode());
65         e = e.getNextException();
66     }
67 } finally {
68     if(cx != null) {
69         try {
70             cx.close();
71         } catch (SQLException e) {
72             e.printStackTrace();
73         }
74     }
75 }
76 }
77 }
78 }
```

Exemple :



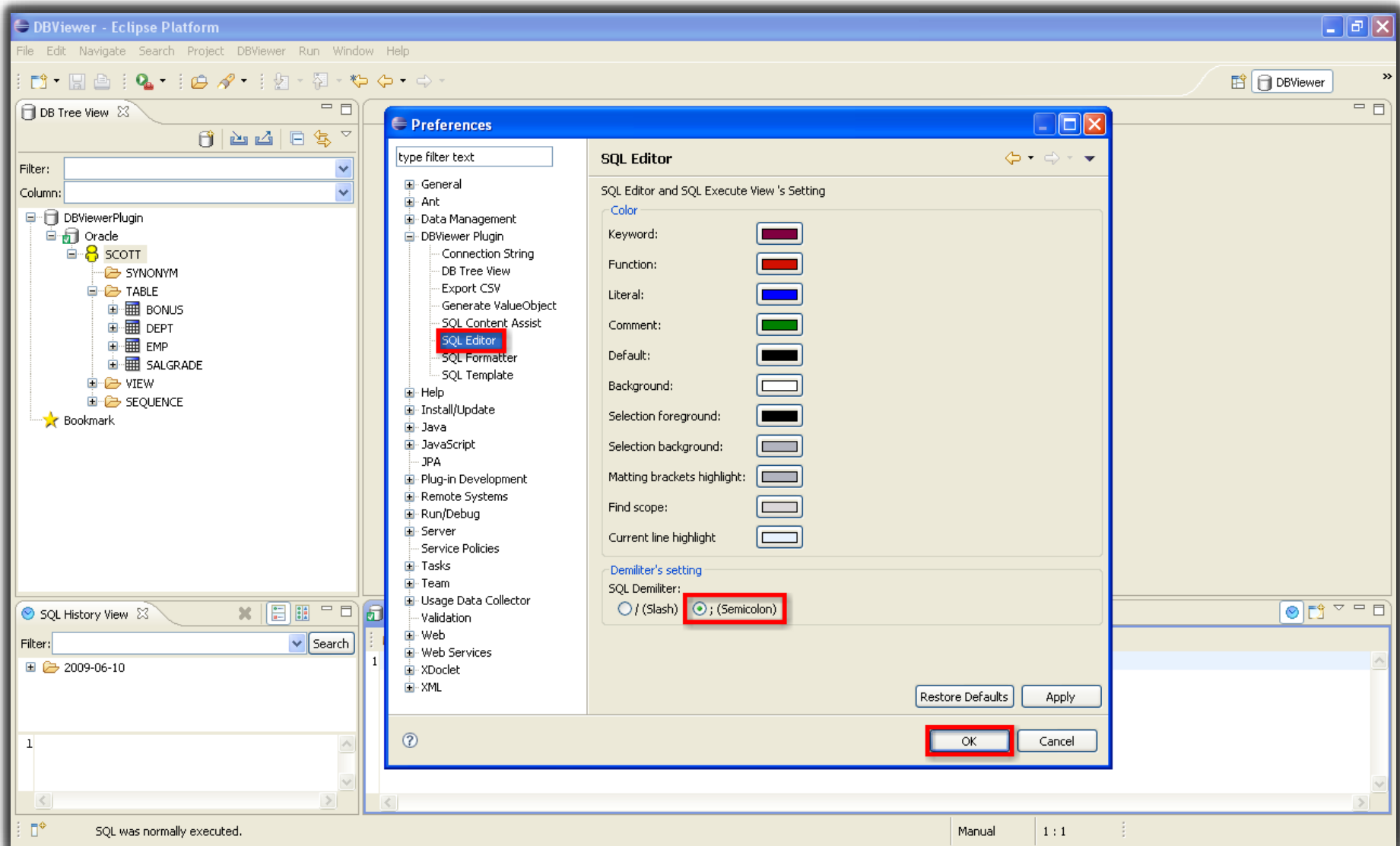
Méthode à utiliser :

Instruction SQL	Méthode	Type de retour
CREATE ALTER DROP	executeUpdate	Int
INSERT UPDATE DELETE	executeUpdate	Int
SELECT	executeQuery	ResultSet

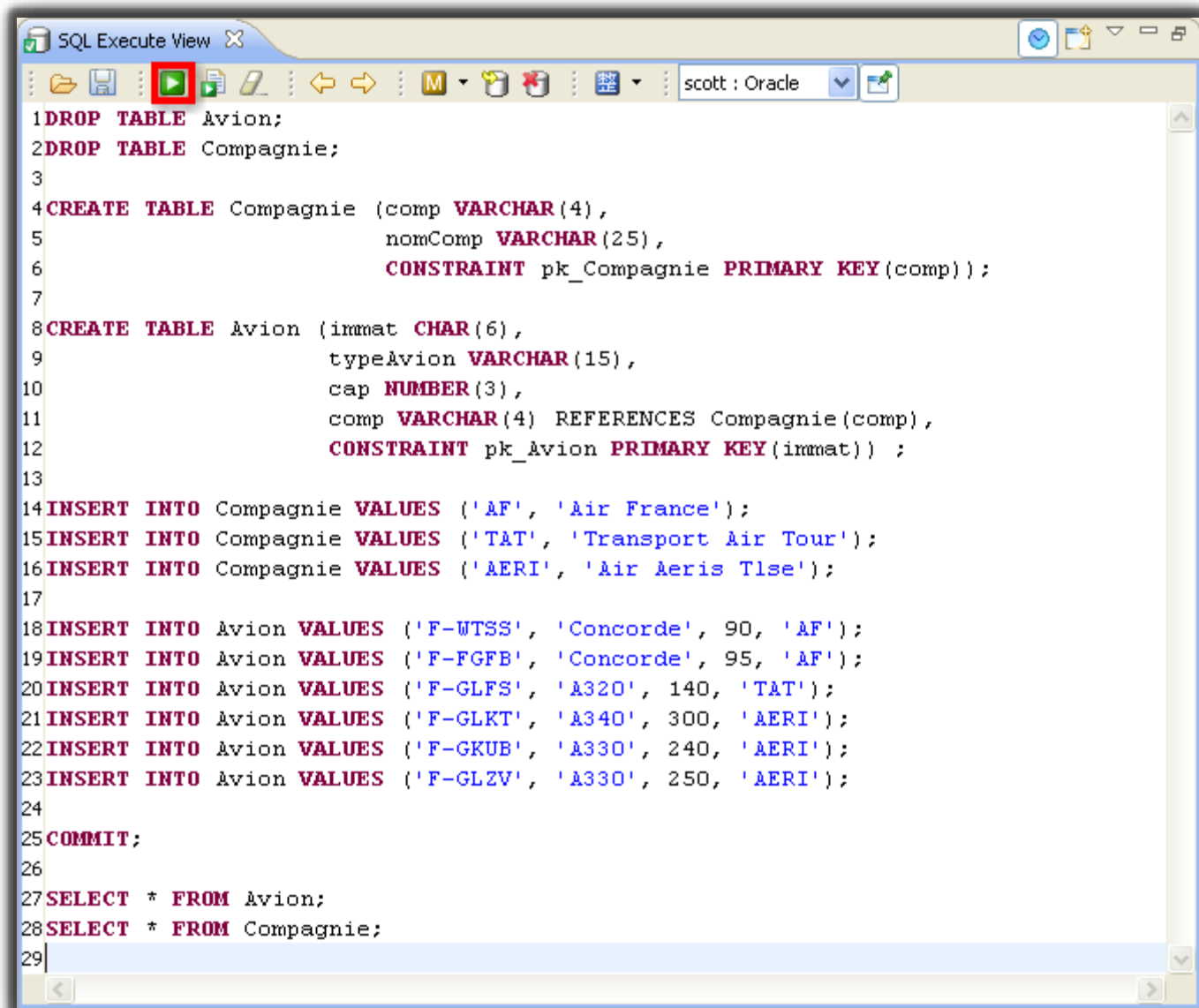
Interaction avec la base :

```
JDBCTest.java X
12
13 Connection cx = null;
14 try {
15     DriverManager.registerDriver(new OracleDriver());
16     cx = DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:orcl","scott","tiger");
17     Statement etat = cx.createStatement();
18     int j;
19
20     // Enregistrement présents dans la table :
21     // Fait la suppression.
22     etat.executeUpdate("DELECTE FROM Avion");
23     // Fait la suppression et passe à j le nombre d'enregistrements supprimés.
24     j = etat.executeUpdate("DELECTE FROM Avion");
25
26     // Aucun enregistrement dans la table :
27     // Aucun action sur la base.
28     etat.executeUpdate("DELECTE FROM Avion");
29     // Aucun action sur la base. Affecte à j la valeur 0.
30     j = etat.executeUpdate("DELECTE FROM Avion");
31
32     // Fait l'insertion.
33     etat.executeUpdate("INSERT INTO Compagnie VALUES('TAF', 'Toulouse Air Free')");
34     // Fait l'insertion, affecte à j la valeur 1.
35     j = etat.executeUpdate("INSERT INTO Compagnie VALUES('TAF', 'Toulouse Air Free')");
36
37     // Fait la modification. Si aucun enregistrement n'est concerné, aucune exception n'est levée.
38     etat.executeUpdate("UPDATE Compagnie SET nomComp = 'Air France Compagny' WHERE comp = 'AF'");
39     // Fait la (les) modification(s). Affect à j le nombre d'enregistrements modifiées
40     // (0 si aucun enregistrement n'est modifié).
41     j = etat.executeUpdate("UPDATE Avion SET capacite = capacite*1.2");
42
```

Extraction de données :



Extraction de données :



The screenshot shows a window titled "SQL Execute View" with a toolbar at the top. The toolbar includes icons for file operations (open, save, print), execution (a green play button icon is highlighted with a red square), and navigation (back, forward, home, end). A dropdown menu shows "scott : Oracle". The main area contains an SQL script with line numbers 1 through 29. The script defines two tables, 'Compagnie' and 'Avion', with their respective columns and constraints, and inserts data into them. The script ends with two SELECT statements to retrieve all data from both tables.

```
1 DROP TABLE Avion;
2 DROP TABLE Compagnie;
3
4 CREATE TABLE Compagnie (comp VARCHAR(4),
5                           nomComp VARCHAR(25),
6                           CONSTRAINT pk_Compagnie PRIMARY KEY(comp));
7
8 CREATE TABLE Avion (immat CHAR(6),
9                      typeAvion VARCHAR(15),
10                     cap NUMBER(3),
11                     comp VARCHAR(4) REFERENCES Compagnie(comp),
12                     CONSTRAINT pk_Avion PRIMARY KEY(immat));
13
14 INSERT INTO Compagnie VALUES ('AF', 'Air France');
15 INSERT INTO Compagnie VALUES ('TAT', 'Transport Air Tour');
16 INSERT INTO Compagnie VALUES ('AERI', 'Air Aeris Tlse');
17
18 INSERT INTO Avion VALUES ('F-WTSS', 'Concorde', 90, 'AF');
19 INSERT INTO Avion VALUES ('F-FGFB', 'Concorde', 95, 'AF');
20 INSERT INTO Avion VALUES ('F-GLFS', 'A320', 140, 'TAT');
21 INSERT INTO Avion VALUES ('F-GLKT', 'A340', 300, 'AERI');
22 INSERT INTO Avion VALUES ('F-GKUB', 'A330', 240, 'AERI');
23 INSERT INTO Avion VALUES ('F-GLZV', 'A330', 250, 'AERI');
24
25 COMMIT;
26
27 SELECT * FROM Avion;
28 SELECT * FROM Compagnie;
29
```

Extraction de données :

DBViewer - [Oracle] SCOTT.COMPAGNIE - Eclipse Platform

File Edit Navigate Search Project Run Window Help

DB Tree View

Filter:

Column:

DBViewerPlugin

Oracle

SCOTT

SYNONYM

TABLE

AVION

BONUS

COMPAGNIE

DEPT

EMP

SALGRADE

VIEW

SEQUENCE

Bookmark

SCOTT.AVION

Where:

	IMMAT	TYPEAVION	CAP	COMP
1	F-WTSS	Concorde	90	AF
2	F-FGFB	Concorde	95	AF
3	F-GLFS	A320	140	TAT
4	F-GLKT	A340	300	AERI
5	F-GKUB	A330	240	AERI
6	F-GLZV	A330	250	AERI

[Oracle] 6 / 6 rows [res:0.7sec] DDL Define

SCOTT.COMPAGNIE

Where:

	COMP	NOMCOMP
1	AF	Air France
2	TAT	Transport Air Tour
3	AERI	Air Aëris Tlse

[Oracle] 3 / 3 rows [res:0.1sec] DDL Define

SQL Execute View

scott : Oracle

```

1 DROP TABLE Avion;
2 DROP TABLE Compagnie;
3
4 CREATE TABLE Compagnie (comp VARCHAR(4),
5                          nomComp VARCHAR(25),
6                          CONSTRAINT pk_Compagnie PRIMARY KEY(comp));
7
8 CREATE TABLE Avion (immat CHAR(6),
9                      typeAvion VARCHAR(15),
10                     cap NUMBER(3),
11                     comp VARCHAR(4) REFERENCES Compagnie(comp),
12                     CONSTRAINT pk_Avion PRIMARY KEY(immat));
13
14 INSERT INTO Compagnie VALUES ('AF', 'Air France');
15 INSERT INTO Compagnie VALUES ('TAT', 'Transport Air Tour');
16 INSERT INTO Compagnie VALUES ('AERI', 'Air Aëris Tlse');
17
18 INSERT INTO Avion VALUES ('F-WTSS', 'Concorde', 90, 'AF');
19 INSERT INTO Avion VALUES ('F-FGFB', 'Concorde', 95, 'AF');

```

SQL History View

Filter: Search

2009-06-10

2009-06-27

01:01:45 DROP TABLE Avion; DROP TABLE Compagnie;

1 DROP TABLE Avion;

2 DROP TABLE Compagnie;

3

4 CREATE TABLE Compagnie (comp VARCHAR(4),

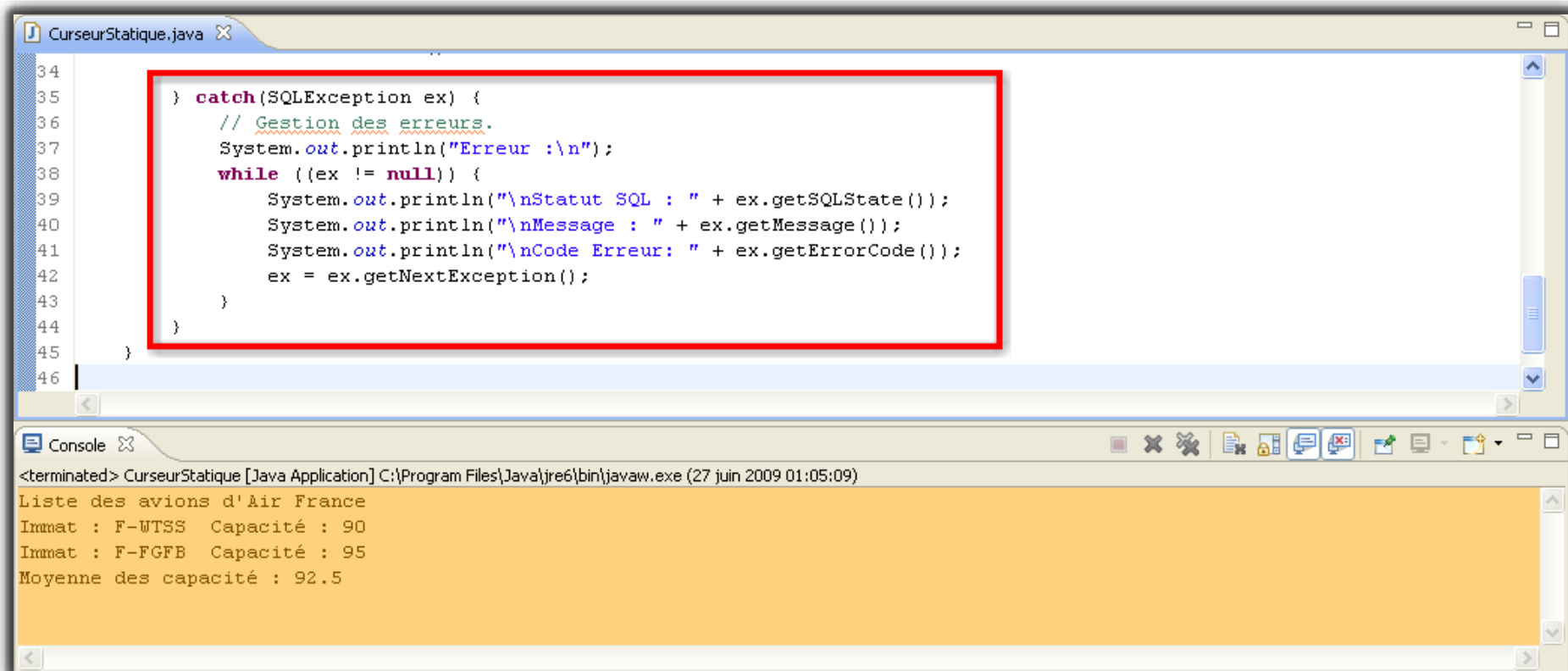
Interface ResultSet :

Méthode	Description
<code>boolean next()</code>	Charge l'enregistrement suivant en retournant <i>true</i> , retourne <i>false</i> lorsqu'il n'y a plus d'enregistrement suivant.
<code>void close()</code>	Ferme le curseur.
<code>getxxx(...)</code>	Récupère, au niveau de l'enregistrement, la valeur de la colonne numérotée de type <i>xxx</i> . Exemple : <i>getInt(1)</i> , <i>getString(1)</i> , <i>getDate(1)</i> , etc, pour récupérer la valeur de la première colonne.
<code>updatexxx(...)</code>	Modifié, au niveau de l'enregistrement, la valeur de la colonne numérotée de type <i>xxx</i> . Exemple : <i>updateInt(1,i)</i> <i>updateString(1,nom)</i> , etc.
<code>ResultSetMetaData getMetaData()</code>	Retourne un objet <i>ResultSetMetaData</i> correspondant au curseur.

Exemple :

```
CurseurStatique.java
1 package ma.etablissement.bd;
2
3 import java.sql.*;
4 import oracle.jdbc.driver.*;
5
6 public class CurseurStatique {
7
8     public static void main(String[] args) {
9         try {
10             DriverManager.registerDriver(new OracleDriver());
11             Connection cx = DriverManager.getConnection("jdbc:oracle:thin:@jberrich:1521:orcl","scott","tiger");
12             // Création de l'état.
13             Statement etatSimple = cx.createStatement();
14             // Création et chargement du curseur.
15             ResultSet curseurJava = etatSimple.executeQuery("SELECT immat,cap " +
16                                                         "FROM Avion " +
17                                                         "WHERE comp = (SELECT comp " +
18                                                         "FROM Compagnie " +
19                                                         "WHERE nomComp='Air France')");
20
21             System.out.println("Liste des avions d'Air France");
22             // Parcours du curseur : Extraction de colonnes.
23             float moyenneCapacité = 0;
24             int nbAvions = 0;
25             while (curseurJava.next ()) {
26                 System.out.print ("Immat : " + curseurJava.getString(1));
27                 System.out.println ("\tCapacité : " + curseurJava.getInt(2));
28                 moyenneCapacité += curseurJava.getInt(2);
29                 nbAvions++;
30             }
31             moyenneCapacité /= nbAvions;
32             System.out.println("Moyenne des capacité : " + moyenneCapacité);
33             // Fermeture du curseur.
34             curseurJava.close();
35         }
36     }
37 }
```

Exemple :



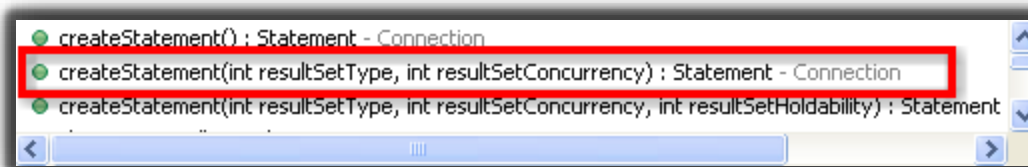
```
34  
35  
36 } catch(SQLException ex) {  
37     // Gestion des erreurs.  
38     System.out.println("Erreur :\n");  
39     while ((ex != null)) {  
40         System.out.println("\nStatut SQL : " + ex.getSQLState());  
41         System.out.println("\nMessage : " + ex.getMessage());  
42         System.out.println("\nCode Erreur: " + ex.getErrorCode());  
43         ex = ex.getNextException();  
44     }  
45 }  
46
```

Console

```
<terminated> CurseurStatique [Java Application] C:\Program Files\Java\jre6\bin\javaw.exe (27 juin 2009 01:05:09)  
Liste des avions d'Air France  
Immat : F-WTSS Capacité : 90  
Immat : F-FGFB Capacité : 95  
Moyenne des capacité : 92.5
```


Curseur navigables :

- ✓ Un curseur statique ResultSet déclaré sans option n'est ni navigable ni modifiable :
 - Seul un déplacement du début vers la fin est permis.
- ✓ Il est possible de rendre un curseur :
 - Navigable en permettant de le parcourir en avant ou en arrière.
 - Modifiable .
- ✓ Rendre l'accès direct à un enregistrement est possible selon deux manières :
 - Absolue : En partant du début ou de la fin.
 - Relative : En partant de la position courante du curseur.



Curseur navigables :

Constante	Explication
<code>ResultSet.TYPE_FORWARD_ONLY</code>	Le parcours du curseur s'opère invariablement du début à la fin (non navigable).
<code>ResultSet.TYPE_SCROLL_INSENSITIVE</code>	Le curseur est navigable mais pas sensible aux modifications.
<code>ResultSet.TYPE_SCROLL_SENSITIVE</code>	Le curseur est navigable et sensible aux modifications.

Curseur navigables :

Méthode	Fonction
<code>boolean isBeforeFirst()</code>	Indique si le curseur est positionné avant le premier enregistrement (<i>false</i> si aucun enregistrement n'existe).
<code>void beforeFirst()</code>	Positionne le curseur avant le premier enregistrement (aucun effet si le curseur est vide).
<code>boolean isFirst()</code>	Indique si le curseur est positionné sur le premier enregistrement (<i>false</i> si aucun enregistrement n'existe).
<code>boolean isLast()</code>	Indique si le curseur est positionné sur le dernier enregistrement (<i>false</i> si aucun enregistrement n'existe).
<code>boolean isAfterLast()</code>	Indique si le curseur est positionné après le dernier enregistrement (<i>false</i> si aucun enregistrement n'existe).

Curseur navigables :

Méthode	Fonction
<code>void afterLast()</code>	Positionne le curseur après le dernier enregistrement (aucun effet si le curseur est vide).
<code>boolean first()</code>	Positionne le curseur sur le premier enregistrement (<i>false</i> si aucun enregistrement n'existe).
<code>boolean previous()</code>	Positionne le curseur sur l'enregistrement précédent (<i>false</i> si aucun enregistrement n'existe).
<code>boolean last()</code>	Positionne le curseur sur le dernier enregistrement (<i>false</i> si aucun enregistrement n'existe).

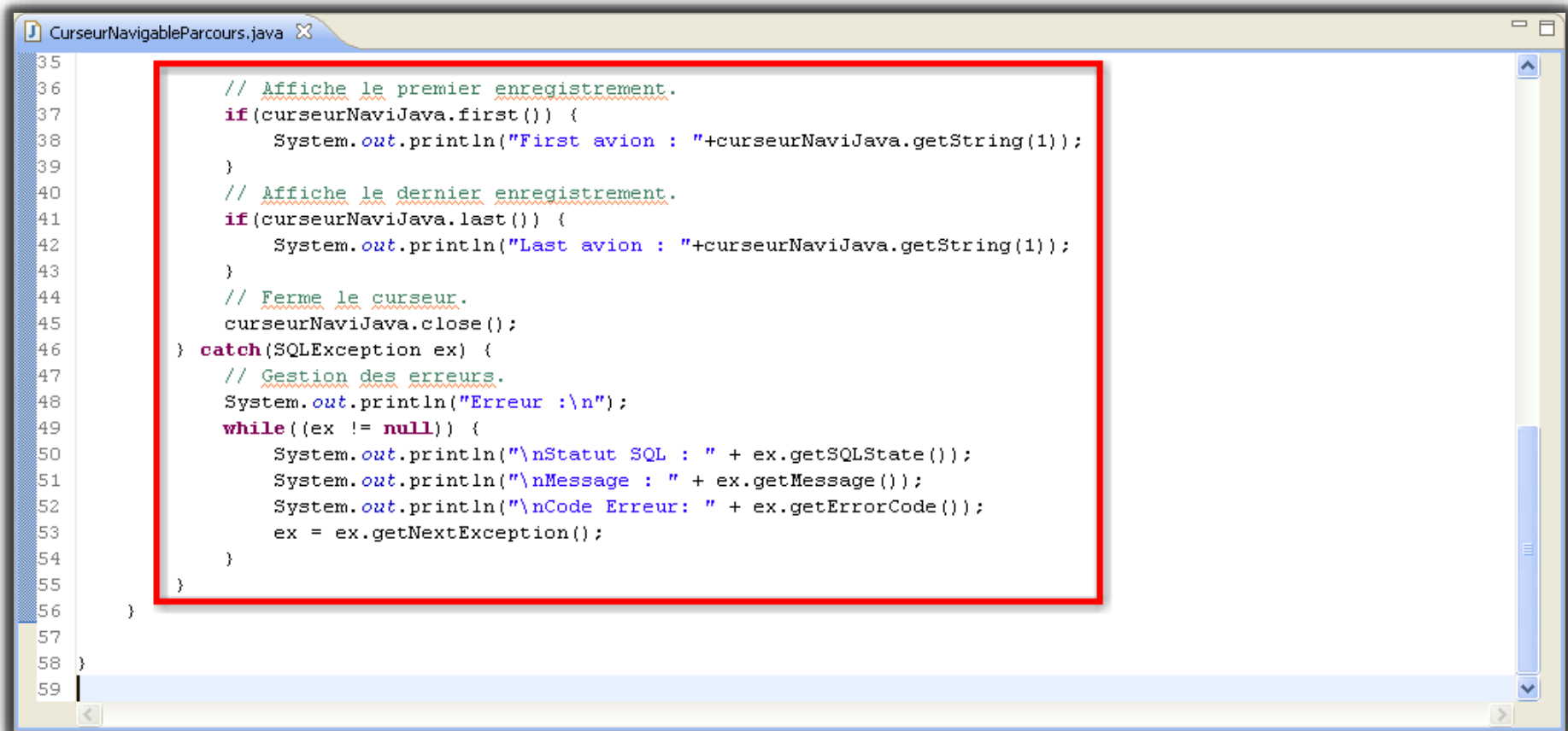
Curseur navigables :

Méthode	Fonction
<code>boolean absolute(int)</code>	Positionne le curseur sur le n-ième enregistrement (en partant du début si <i>n</i> positif, ou de la fin si <i>n</i> négatif, <i>false</i> si aucun enregistrement n'existe à cet indice).
<code>boolean relative(int)</code>	Positionne le curseur sur le n-ième enregistrement en partant de la position courante (en avant si <i>n</i> positif, ou en arrière si <i>n</i> négatif, <i>false</i> si aucun enregistrement n'existe à cet indice).

Exemple :

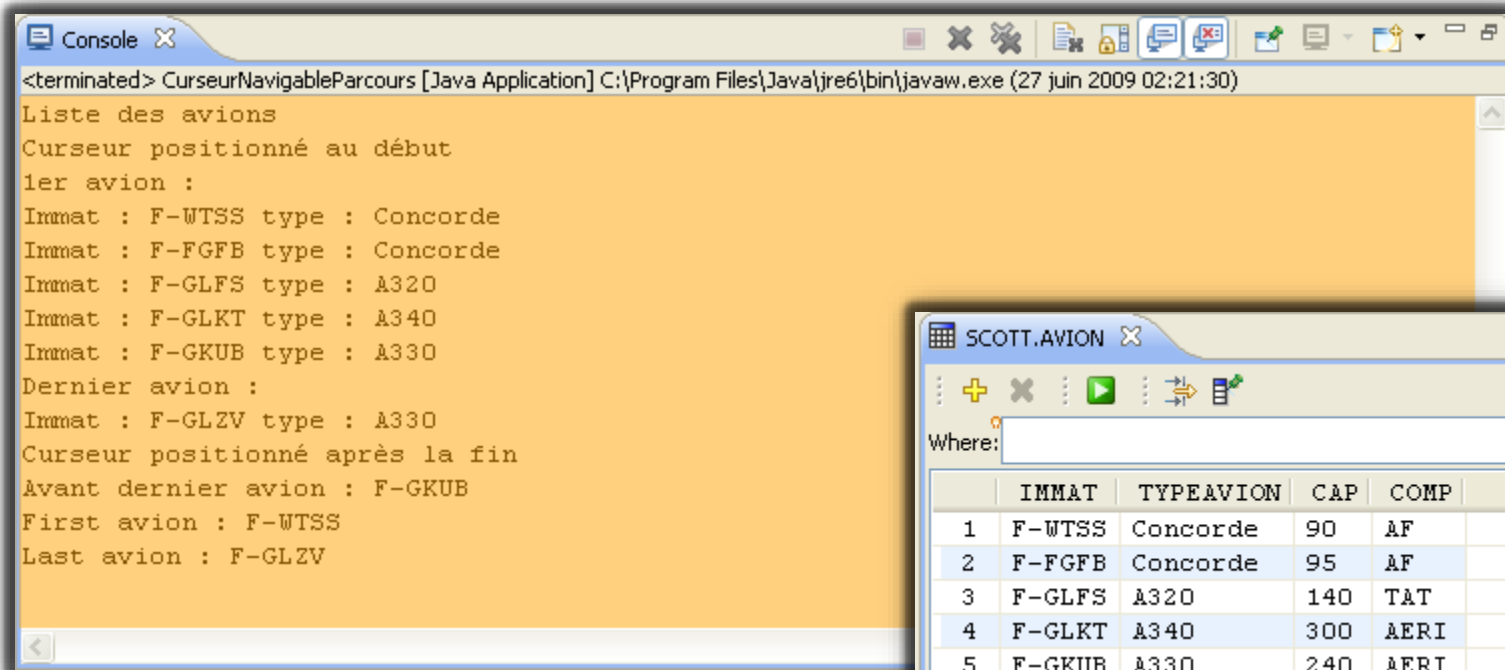
```
CurseurNavigableParcours.java X
1 package ma.etablissement.bd;
2
3 import java.sql.*;
4 import oracle.jdbc.driver.*;
5
6 public class CurseurNavigableParcours {
7
8     public static void main(String[] args) {
9         try {
10             DriverManager.registerDriver(new OracleDriver());
11             Connection cx = DriverManager.getConnection("jdbc:oracle:thin:@jberrich:1521:orcl","scott","tiger");
12             // Création de l'état.
13             Statement etatSimple = cx.createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,ResultSet.CONCUR_READ_ONLY);
14             // Création et chargement du curseur.
15             ResultSet curseurNaviJava = etatSimple.executeQuery("SELECT immat,typeAvion,cap FROM Avion");
16             System.out.println("Liste des avions");
17             // Test renvoyant true.
18             if(curseurNaviJava.isBeforeFirst()) System.out.println("Curseur positionné au début");
19             // Test renvoyant false.
20             if(curseurNaviJava.isFirst()) System.out.println("Curseur positionné sur le 1er déjà");
21             // Parcours du curseur en affichant les premier et dernier enregistrements.
22             while(curseurNaviJava.next()) {
23                 if(curseurNaviJava.isFirst()) System.out.println("1er avion : ");
24                 if(curseurNaviJava.isLast()) System.out.println("Dernier avion : ");
25                 System.out.print("Immat : " + curseurNaviJava.getString(1));
26                 System.out.println(" type : " + curseurNaviJava.getString(2));
27             }
28             // Test renvoyant true.
29             if(curseurNaviJava.isAfterLast()) System.out.println("Curseur positionné après la fin");
30             // Affiche l'avant-dernier enregistrement.
31             if(curseurNaviJava.previous())
32                 if (curseurNaviJava.previous()) {
33                     System.out.println("Avant dernier avion : " + curseurNaviJava.getString(1));
34                 }
35         }
36     }
37 }
```

Exemple :

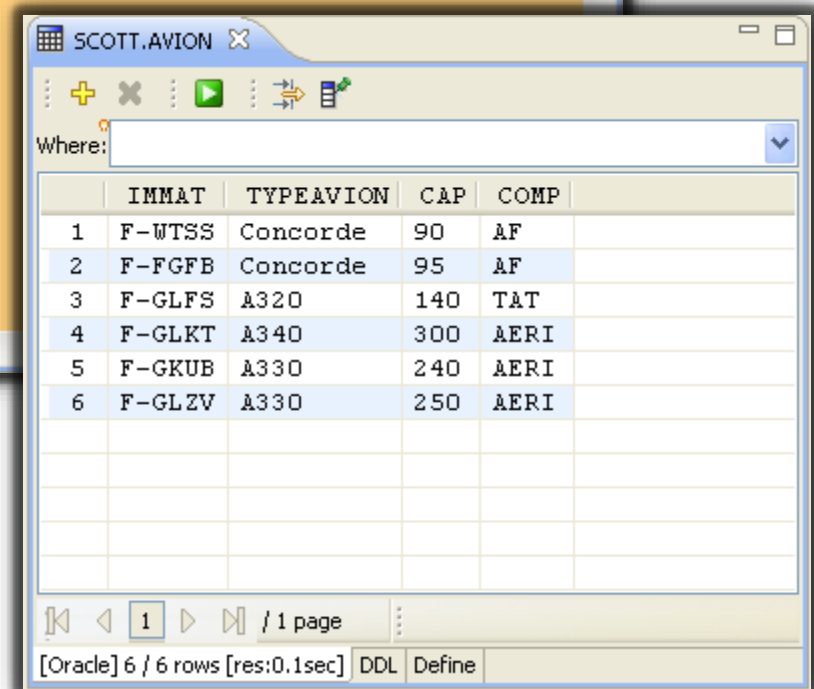


```
35
36 // Affiche le premier enregistrement.
37 if (curseurNaviJava.first()) {
38     System.out.println("First avion : "+curseurNaviJava.getString(1));
39 }
40 // Affiche le dernier enregistrement.
41 if (curseurNaviJava.last()) {
42     System.out.println("Last avion : "+curseurNaviJava.getString(1));
43 }
44 // Ferme le curseur.
45 curseurNaviJava.close();
46 } catch (SQLException ex) {
47     // Gestion des erreurs.
48     System.out.println("Erreur :\n");
49     while (ex != null) {
50         System.out.println("\nStatut SQL : " + ex.getSQLState());
51         System.out.println("\nMessage : " + ex.getMessage());
52         System.out.println("\nCode Erreur : " + ex.getErrorCode());
53         ex = ex.getNextException();
54     }
55 }
56 }
57
58 }
59 }
```

Exemple :



```
<terminated> CurseurNavigableParcours [Java Application] C:\Program Files\Java\jre6\bin\javaw.exe (27 juin 2009 02:21:30)
Liste des avions
Curseur positionné au début
1er avion :
Immat : F-WTSS type : Concorde
Immat : F-FGFB type : Concorde
Immat : F-GLFS type : A320
Immat : F-GLKT type : A340
Immat : F-GKUB type : A330
Dernier avion :
Immat : F-GLZV type : A330
Curseur positionné après la fin
Avant dernier avion : F-GKUB
First avion : F-WTSS
Last avion : F-GLZV
```



	IMMAT	TYPEAVION	CAP	COMP
1	F-WTSS	Concorde	90	AF
2	F-FGFB	Concorde	95	AF
3	F-GLFS	A320	140	TAT
4	F-GLKT	A340	300	AERI
5	F-GKUB	A330	240	AERI
6	F-GLZV	A330	250	AERI

[Oracle] 6 / 6 rows [res:0.1sec] DDL Define

Exemple :

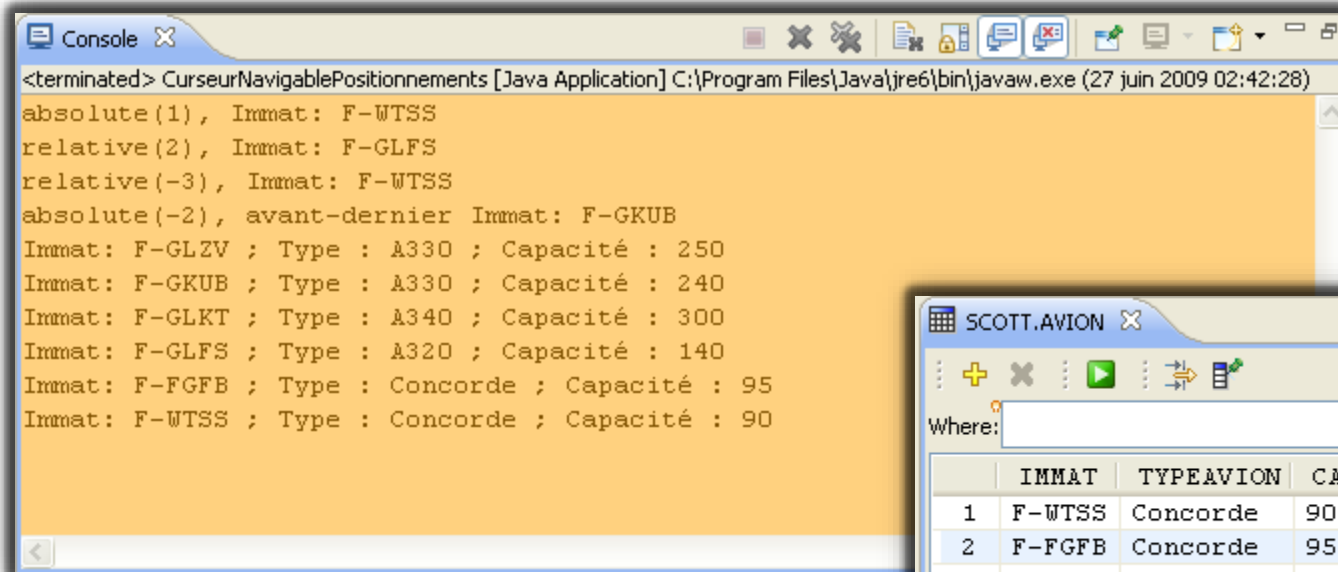
```
CurseurNavigablePositionnements.java X
1 package ma.etablissement.bd;
2
3 import java.sql.*;
4 import oracle.jdbc.driver.*;
5
6 public class CurseurNavigablePositionnements {
7
8     public static void main(String[] args) {
9         try {
10             DriverManager.registerDriver(new OracleDriver());
11             Connection cx = DriverManager.getConnection("jdbc:oracle:thin:@jberrich:1521:orcl","scott","tiger");
12             // Création de l'état avec curseurs insensibles et non modifiables.
13             Statement etatSimple = cx.createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,ResultSet.CONCUR_READ_ONLY);
14             // Création et chargement du curseur.
15             ResultSet curseurPosJava = etatSimple.executeQuery("SELECT immat,typeAvion,cap FROM Avion");
16             // Curseur sur le premier avion.
17             curseurPosJava.absolute(1);
18             System.out.println("absolute(1), Immat: " + curseurPosJava.getString(1));
19             // Accès au troisième avion.
20             if (curseurPosJava.relative(2))
21                 System.out.println("relative(2), Immat: " + curseurPosJava.getString(1));
22             else
23                 System.out.println("Pas de 3ème avion!");
24             // Retour au premier avion.
25             if (curseurPosJava.relative(-2))
26                 System.out.println("relative(-3), Immat: " + curseurPosJava.getString(1));
27             else
28                 System.out.println("Pas de 3ème avion donc pas de retour -2!");
29             // Accès à l'avant-dernier enregistrement.
30             if (curseurPosJava.absolute(-2))
31                 System.out.println("absolute(-2), avant-dernier Immat: " + curseurPosJava.getString(1));
32             else
33                 System.out.println("Pas d'avant dernier avion");
34         }
35     }
36 }
```

Exemple :

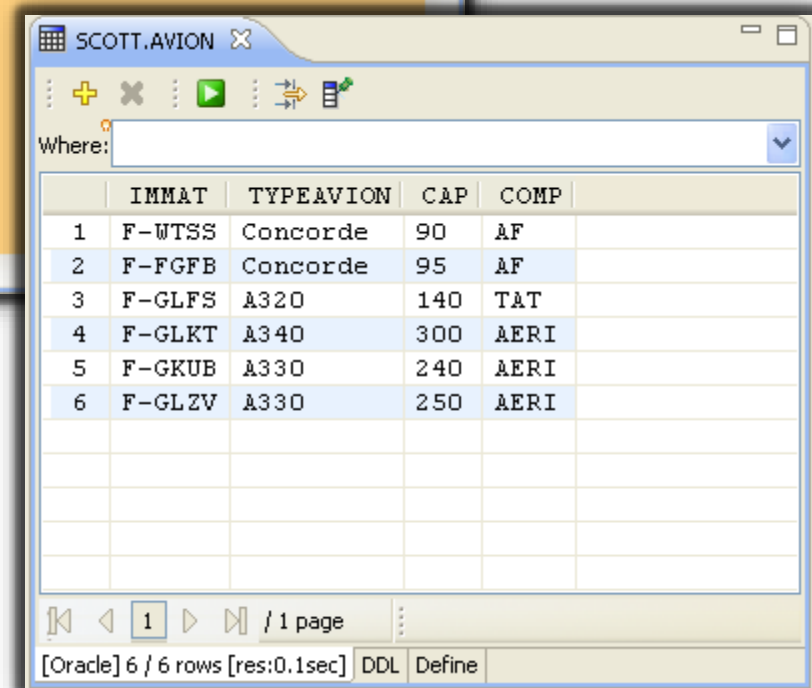


```
34
35 // Parcours du curseur en sens inverse.
36 curseurPosJava.afterLast();
37 while (curseurPosJava.previous()) {
38     System.out.print("Immat: " + curseurPosJava.getString(1));
39     System.out.print(" ; Type : " + curseurPosJava.getString(2));
40     System.out.print(" ; Capacité : " + curseurPosJava.getInt(3) + "\n");
41 }
42 // Ferme le curseur.
43 curseurPosJava.close();
44
45 } catch (SQLException ex) {
46     System.out.println("Erreur :\n");
47     while ((ex != null)) {
48         System.out.println("\nStatut SQL : " + ex.getSQLState());
49         System.out.println("\nMessage : " + ex.getMessage());
50         System.out.println("\nCode Erreur: " + ex.getErrorCode());
51         ex = ex.getNextException();
52     }
53 }
54 }
55
```

Exemple :



```
<terminated> CurseurNavigablePositionnements [Java Application] C:\Program Files\Java\jre6\bin\javaw.exe (27 juin 2009 02:42:28)
absolute(1), Immat: F-WTSS
relative(2), Immat: F-GLFS
relative(-3), Immat: F-WTSS
absolute(-2), avant-dernier Immat: F-GKUB
Immat: F-GLZV ; Type : A330 ; Capacité : 250
Immat: F-GKUB ; Type : A330 ; Capacité : 240
Immat: F-GLKT ; Type : A340 ; Capacité : 300
Immat: F-GLFS ; Type : A320 ; Capacité : 140
Immat: F-FGFB ; Type : Concorde ; Capacité : 95
Immat: F-WTSS ; Type : Concorde ; Capacité : 90
```



Where:

	IMMAT	TYPEAVION	CAP	COMP
1	F-WTSS	Concorde	90	AF
2	F-FGFB	Concorde	95	AF
3	F-GLFS	A320	140	TAT
4	F-GLKT	A340	300	AERI
5	F-GKUB	A330	240	AERI
6	F-GLZV	A330	250	AERI

[Oracle] 6 / 6 rows [res:0.1sec] DDL Define

Curseur modifiable:

- ✓ Un curseur modifiable permet de mettre à jour la base de données :
 - Modification de colonnes.
 - Suppression et insertion d'enregistrements.

Constante	Explication
<code>ResultSet.CONCUR_READ_ONLY</code>	Le curseur ne peut être modifié.
<code>ResultSet.CONCUR_UPDATABLE</code>	Le curseur peut être modifié.

Curseur modifiables :

Méthode	Fonction
<code>int getResultSetType()</code>	Renvoie le caractère navigable des curseurs d'un état donné.
<code>int getResultSetConcurrency()</code>	Renvoie le caractère modifiable des curseurs d'un état donné.
<code>int getType()</code>	Renvoie le caractère navigable d'un curseur donné.
<code>int getConcurrency()</code>	Renvoie le caractère modifiable d'un curseur donné.

Curseur modifiables :

Méthode	Fonction
<code>void deleteRow()</code>	Supprime l'enregistrement courant.
<code>void updateRow()</code>	Modifie la table avec l'enregistrement courant.
<code>void cancelRowUpdate</code>	Annule les modifications faites sur l'enregistrement courant.
<code>void moveToInsertRow()</code>	Déplace le curseur vers un nouvel enregistrement.
<code>void insertRow()</code>	Insère dans la table l'enregistrement courant.
<code>void moveToCurrentRow()</code>	Retour vers l'enregistrement courant (à utiliser éventuellement après <i>moveToInsertRow</i>).

Exemple :

```
CurseurModifiableSuppression.java
1 package ma.etablissement.bd;
2
3 import java.sql.*;
4 import oracle.jdbc.driver.*;
5
6 public class CurseurModifiableSuppression {
7
8     public static void main(String[] args) {
9         try{
10             DriverManager.registerDriver(new OracleDriver());
11             // Création de l'état et désactivation de la validation automatique.
12             Connection cx = DriverManager.getConnection("jdbc:oracle:thin:@jberrich:1521:orcl","scott","tiger");
13             Statement etatSimple = cx.createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,ResultSet.CONCUR_UPDATABLE);
14             cx.setAutoCommit(false);
15             // Création du curseur.
16             ResultSet curseurModifJava = etatSimple.executeQuery("SELECT immat,typeAvion,cap FROM Avion");
17             // Accès direct au troisième avion, suppression de l'enregistrement.
18             if (curseurModifJava.absolute(3)) {
19                 System.out.println("Immat à supprimer : " + curseurModifJava.getString(1));
20                 curseurModifJava.deleteRow();
21                 cx.commit();
22                 System.out.println("Supprimé au niveau de la base.");
23             } else
24                 System.out.println("Pas de 3ème avion!");
25             curseurModifJava.beforeFirst();
26             while(curseurModifJava.next()) {
27                 System.out.print("Immat : " + curseurModifJava.getString(1));
28                 System.out.println(" type : " + curseurModifJava.getString(2));
29             }
30             // Ferme le curseur.
31             curseurModifJava.close();
32         }
```

Exemple :

```
CurseurModifiableSuppression.java X
32
33 } catch(SQLException ex) {
34     // Gestion des erreurs.
35     System.out.println("Erreur :\n");
36     while ((ex != null)) {
37         System.out.println("\nStatut SQL : " + ex.getSQLState());
38         System.out.println("\nMessage : " + ex.getMessage());
39         System.out.println("\nCode Erreur : " + ex.getErrorCode());
40         ex = ex.getNextException();
41     }
42 }
43
44 }
45
46
```

SCOTT.AVION X

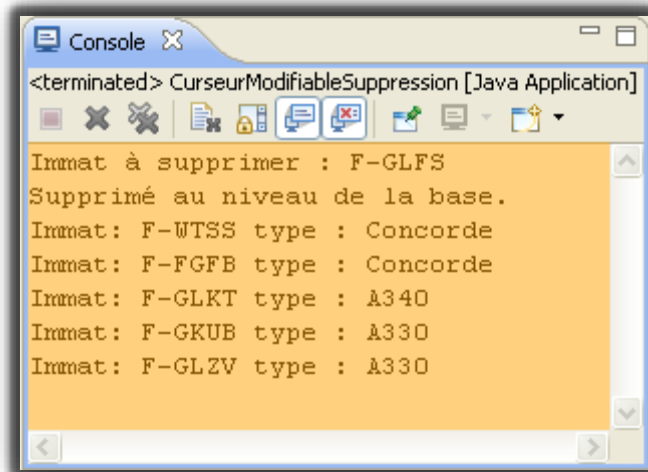
Where:

	IMMAT	TYPEAVION	CAP	COMP
1	F-WTSS	Concorde	90	AF
2	F-FGFB	Concorde	95	AF
3	F-GLFS	A320	140	TAT
4	F-GLKT	A340	300	AERI
5	F-GKUB	A330	240	AERI
6	F-GLZV	A330	250	AERI

1 / 1 page

[Oracle] 6 / 6 rows [res:0.0sec] DDL Define

Exemple :



SCOTT.AVION

Where:

	IMMAT	TYPEAVION	CAP	COMP
1	F-WTSS	Concorde	90	AF
2	F-FGFB	Concorde	95	AF
3	F-GLKT	A340	300	AERI
4	F-GKUB	A330	240	AERI
5	F-GLZV	A330	250	AERI

1 / 1 page

[Oracle] 5 / 5 rows [res:0.1sec] DDL Define

Exemple :

```
CurseurModifiableSuppression2.java
1 package ma.etablissement.bd;
2
3 import java.sql.*;
4 import oracle.jdbc.driver.*;
5
6 public class CurseurModifiableSuppression2 {
7
8     public static void main(String[] args) {
9         try {
10             DriverManager.registerDriver(new OracleDriver());
11             Connection cx = DriverManager.getConnection("jdbc:oracle:thin:@jberrich:1521:orcl","scott","tiger");
12             // Création de l'état et désactivation de la validation automatique.
13             Statement etatSimple = cx.createStatement(ResultSet.TYPE_FORWARD_ONLY,ResultSet.CONCUR_UPDATABLE);
14             cx.setAutoCommit(false);
15             // Création du curseur.
16             ResultSet curseurModifJava = etatSimple.executeQuery("SELECT immat,typeAvion,cap FROM Avion");
17             // Accès à l'enregistrement et suppression.
18             String pImmat = "F-GLFS";
19             while(curseurModifJava.next()) {
20                 System.out.println("Longueur immat : " + curseurModifJava.getString(1).length());
21                 if(curseurModifJava.getString(1).equals(pImmat)) {
22                     System.out.println("Immat à supprimer : " + curseurModifJava.getString(1));
23                     curseurModifJava.deleteRow();
24                     cx.commit();
25                     System.out.println("Supprimé au niveau de la base.");
26                 } else {
27                     System.out.print("Immat: [" + curseurModifJava.getString(1) + "]");
28                     System.out.println(" type : " + curseurModifJava.getString(2));
29                 }
30             }
31             // Ferme le curseur.
32             curseurModifJava.close();
33         }
34     }
35 }
```

Exemple :

The image shows two overlapping windows. The background window is a Java IDE titled 'CurseurModifiableSuppression2.java'. It contains a code snippet for handling SQL exceptions, highlighted with a red rectangle. The code is as follows:

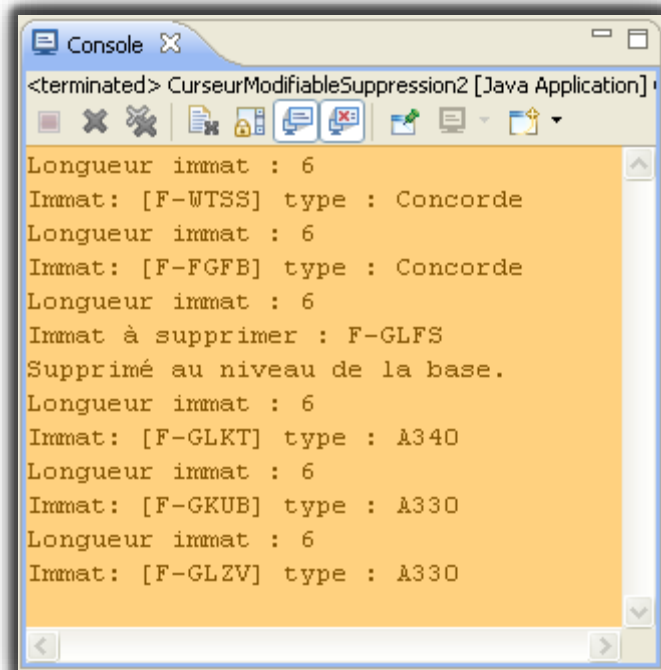
```
33  
34 } catch(SQLException ex) {  
35     // Gestion des erreurs.  
36     System.out.println("Erreur :\n");  
37     while ((ex != null)) {  
38         System.out.println("\nStatut SQL : " + ex.getSQLState());  
39         System.out.println("\nMessage : " + ex.getMessage());  
40         System.out.println("\nCode Erreur : " + ex.getErrorCode());  
41         ex = ex.getNextException();  
42     }  
43 }  
44  
45 }  
46  
47
```

The foreground window is a database viewer titled 'SCOTT.AVION'. It displays a table with the following data:

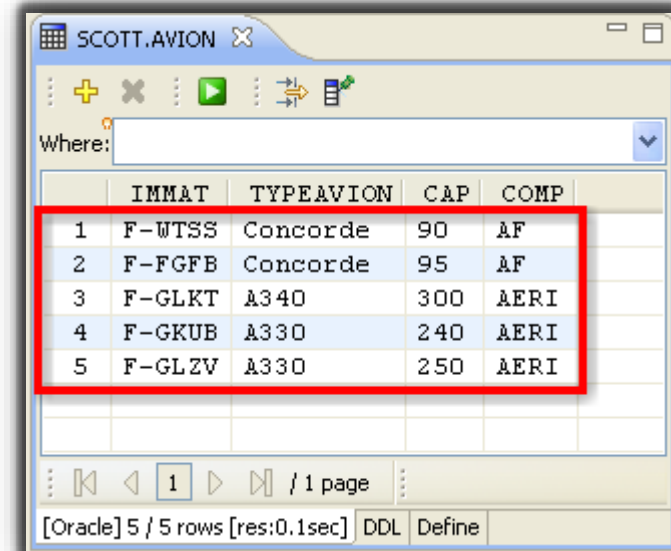
	IMMAT	TYPEAVION	CAP	COMP
1	F-WTSS	Concorde	90	AF
2	F-FGFB	Concorde	95	AF
3	F-GLFS	A320	140	TAT
4	F-GLKT	A340	300	AERI
5	F-GKUB	A330	240	AERI
6	F-GLZV	A330	250	AERI

The row with IMMAT 'F-GLFS' (row 3) is highlighted with a red rectangle. The database viewer also shows a 'Where:' field and navigation controls at the bottom.

Exemple :



```
<terminated> CurseurModifiableSuppression2 [Java Application]
Longueur immat : 6
Immat: [F-WTSS] type : Concorde
Longueur immat : 6
Immat: [F-FGFB] type : Concorde
Longueur immat : 6
Immat à supprimer : F-GLFS
Supprimé au niveau de la base.
Longueur immat : 6
Immat: [F-GLKT] type : A340
Longueur immat : 6
Immat: [F-GKUB] type : A330
Longueur immat : 6
Immat: [F-GLZV] type : A330
```



	IMMAT	TYPEAVION	CAP	COMP
1	F-WTSS	Concorde	90	AF
2	F-FGFB	Concorde	95	AF
3	F-GLKT	A340	300	AERI
4	F-GKUB	A330	240	AERI
5	F-GLZV	A330	250	AERI

[Oracle] 5 / 5 rows [res:0.1sec] DDL Define

Exemple :

```
CurseurModifiableModification.java
1 package ma.etablissement.bd;
2
3 import java.sql.*;
4 import oracle.jdbc.driver.*;
5
6 public class CurseurModifiableModification {
7
8     public static void main(String[] args) {
9         try {
10             DriverManager.registerDriver(new OracleDriver());
11             // Création de l'état et désactivation de la validation automatique.
12             Connection cx = DriverManager.getConnection("jdbc:oracle:thin:@jberrich:1521:orcl","scott","tiger");
13             Statement etatSimple = cx.createStatement(ResultSet.TYPE_SCROLL_SENSITIVE,ResultSet.CONCUR_UPDATABLE);
14             cx.setAutoCommit(false);
15             // Création du curseur.
16             ResultSet curseurModifJava = etatSimple.executeQuery("SELECT immat,typeAvion,cap FROM Avion");
17             // Accès à l'enregistrement.
18             if(curseurModifJava.absolute(5)) {
19                 System.out.println("Avion à modifier : " + curseurModifJava.getString(1));
20                 curseurModifJava.updateString(2,"A380");
21                 curseurModifJava.updateInt("CAP",350);
22                 curseurModifJava.updateRow();
23                 // Validation.
24                 cx.commit();
25                 System.out.println("Modifié au niveau de la base.");
26             } else
27                 System.out.println("Pas de 5ème avion!");
28             // Ferme le curseur.
29             curseurModifJava.beforeFirst();
30             while(curseurModifJava.next()) {
31                 System.out.print("Immat: " + curseurModifJava.getString(1));
32                 System.out.println(" type : " + curseurModifJava.getString(2));
33                 System.out.println(" capacité : " + curseurModifJava.getInt(3));
34             }
35             curseurModifJava.close();
36         }
37     }
38 }
```

Exemple :

The image shows two overlapping windows from a software development environment. The background window is a Java editor titled 'CurseurModifiableModification.java'. It contains a code snippet for handling SQL exceptions, highlighted with a red rectangle. The code is as follows:

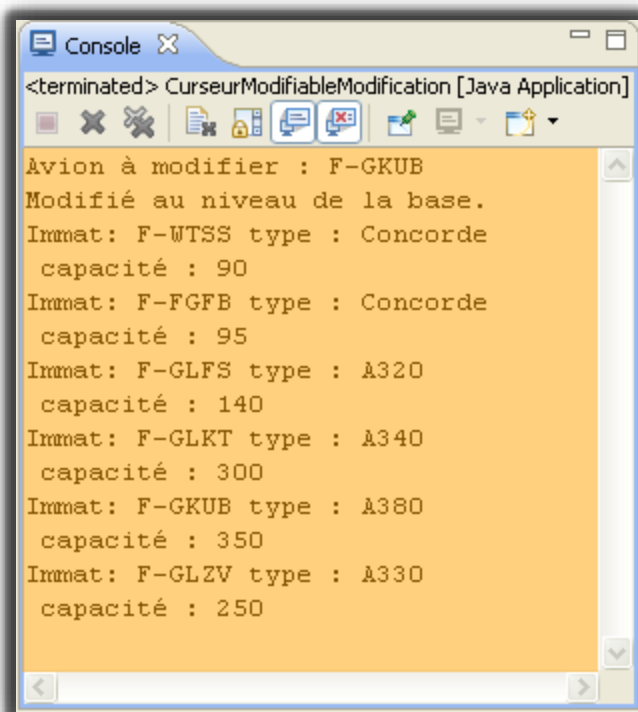
```
36
37 } catch(SQLException ex){
38     // Gestion des erreurs.
39     System.out.println("Erreur :\n");
40     while ((ex != null)) {
41         System.out.println("\nStatut SQL : " + ex.getSQLState());
42         System.out.println("\nMessage : " + ex.getMessage());
43         System.out.println("\nCode Erreur: " + ex.getErrorCode());
44         ex = ex.getNextException();
45     }
46 }
47
48
49 }
50
```

The foreground window is a database viewer titled 'SCOTT.AVION'. It displays a table with the following data:

	IMMAT	TYPEAVION	CAP	COMP
1	F-WTSS	Concorde	90	AF
2	F-FGFB	Concorde	95	AF
3	F-GLFS	A320	140	TAT
4	F-GLKT	A340	300	AERI
5	F-GKUB	A330	240	AERI
6	F-GLZV	A330	250	AERI

The row with index 5 (F-GKUB, A330, 240, AERI) is highlighted with a red rectangle. The database viewer also shows a 'Where:' field and navigation controls at the bottom.

Exemple :



SCOTT.AVION

Where:

	IMMAT	TYPEAVION	CAP	COMP
1	F-WTSS	Concorde	90	AF
2	F-FGFB	Concorde	95	AF
3	F-GLFS	A320	140	TAT
4	F-GLKT	A340	300	AERI
5	F-GKUB	A380	350	AERI
6	F-GLZV	A330	250	AERI

1 / 1 page

[Oracle] 6 / 6 rows [res:0.0sec] DDL Define

Exemple :

```
1 package ma.etablissement.bd;
2
3 import java.sql.*;
4 import oracle.jdbc.driver.*;
5
6 public class CurseurModifiableInsertion {
7
8     public static void main(String[] args) {
9         try {
10             DriverManager.registerDriver(new OracleDriver());
11             // Création de l'état et désactivation de la validation automatique.
12             Connection cx = DriverManager.getConnection("jdbc:oracle:thin:@jberrich:1521:orcl","scott","tiger");
13             Statement etatSimple = cx.createStatement(ResultSet.TYPE_SCROLL_SENSITIVE,ResultSet.CONCUR_UPDATABLE);
14             cx.setAutoCommit(false);
15             // Création du curseur.
16             ResultSet curseurModifJava = etatSimple.executeQuery("SELECT immat,typeAvion,cap FROM Avion");
17             // Première étape.
18             curseurModifJava.moveToInsertRow();
19             // Deuxième étape.
20             curseurModifJava.updateString(1,"F-LUTE");
21             curseurModifJava.updateString(2,"TB20");
22             curseurModifJava.updateInt(3,4);
23             // Troisième étape.
24             curseurModifJava.insertRow();
25             // Validation.
26             cx.commit();
27             System.out.println("Inséré au niveau de la base.");
28             curseurModifJava.beforeFirst();
29             while(curseurModifJava.next()) {
30                 System.out.print("Immat: " + curseurModifJava.getString(1));
31                 System.out.print(" ;type : " + curseurModifJava.getString(2));
32                 System.out.print(" ;capacité : " + curseurModifJava.getInt(3) + "\n");
33             }
34             // Ferme le curseur.
35             curseurModifJava.close();
36         }
37     }
38 }
```


Exemple :

The image shows a Java IDE window titled 'CurseurModifiableInsertion.java' with a red box highlighting a catch block for `SQLException`. The code inside the catch block is as follows:

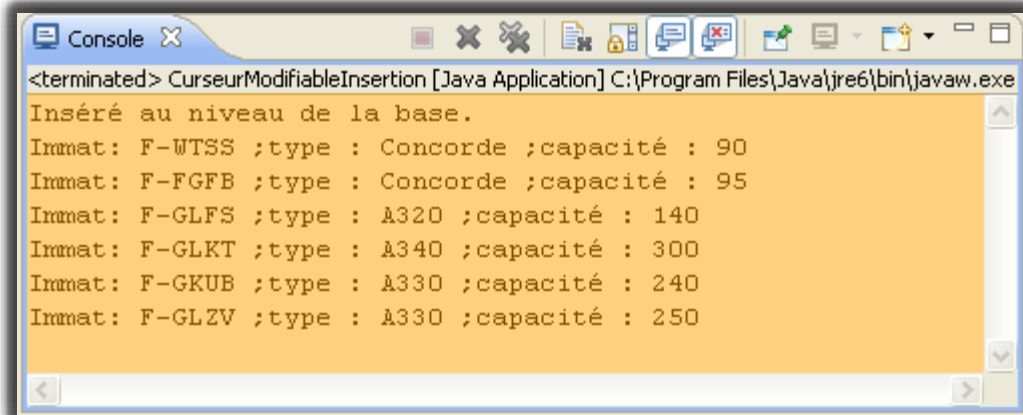
```
36  
37 } catch(SQLException ex) {  
38     // Gestion des erreurs.  
39     System.out.println("Erreur :\n");  
40     while ((ex != null)) {  
41         System.out.println("\nStatut SQL : " + ex.getSQLState());  
42         System.out.println("\nMessage : " + ex.getMessage());  
43         System.out.println("\nCode Erreur : " + ex.getErrorCode());  
44         ex = ex.getNextException();  
45     }  
46 }  
47  
48  
49 }  
50
```

Below the IDE window, there is a SQL query result window titled 'SCOTT.AVION'. It displays a table with 6 rows and 5 columns: IMMAT, TYPEAVION, CAP, COMP, and an empty column. The data is as follows:

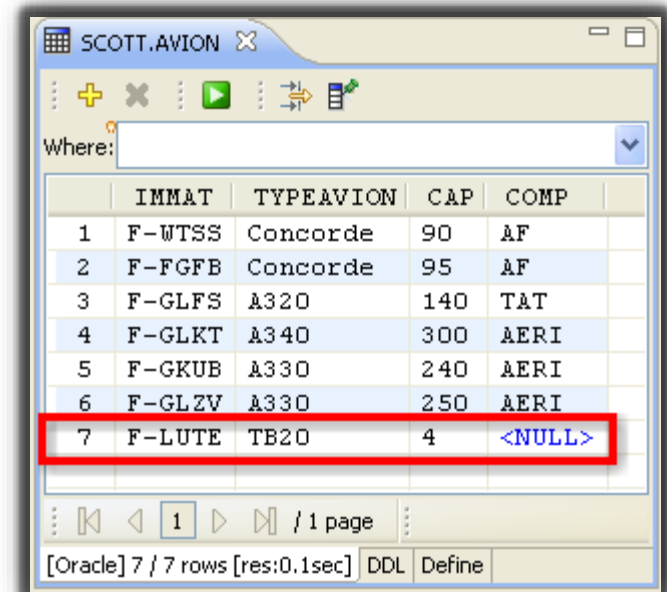
	IMMAT	TYPEAVION	CAP	COMP	
1	F-WTSS	Concorde	90	AF	
2	F-FGFB	Concorde	95	AF	
3	F-GLFS	A320	140	TAT	
4	F-GLKT	A340	300	AERI	
5	F-GKUB	A330	240	AERI	
6	F-GLZV	A330	250	AERI	

The SQL query result window also shows a 'Where:' field and a 'DDL Define' button. The status bar at the bottom indicates '[Oracle] 6 / 6 rows [res:0.6sec] DDL Define'.

Exemple :



```
<terminated> CurseurModifiableInsertion [Java Application] C:\Program Files\Java\jre6\bin\javaw.exe
Inséré au niveau de la base.
Immat: F-WTSS ;type : Concorde ;capacité : 90
Immat: F-FGFB ;type : Concorde ;capacité : 95
Immat: F-GLFS ;type : A320 ;capacité : 140
Immat: F-GLKT ;type : A340 ;capacité : 300
Immat: F-GKUB ;type : A330 ;capacité : 240
Immat: F-GLZV ;type : A330 ;capacité : 250
```



SCOTT.AVION

Where:

	IMMAT	TYPEAVION	CAP	COMP
1	F-WTSS	Concorde	90	AF
2	F-FGFB	Concorde	95	AF
3	F-GLFS	A320	140	TAT
4	F-GLKT	A340	300	AERI
5	F-GKUB	A330	240	AERI
6	F-GLZV	A330	250	AERI
7	F-LUTE	TB20	4	<NULL>

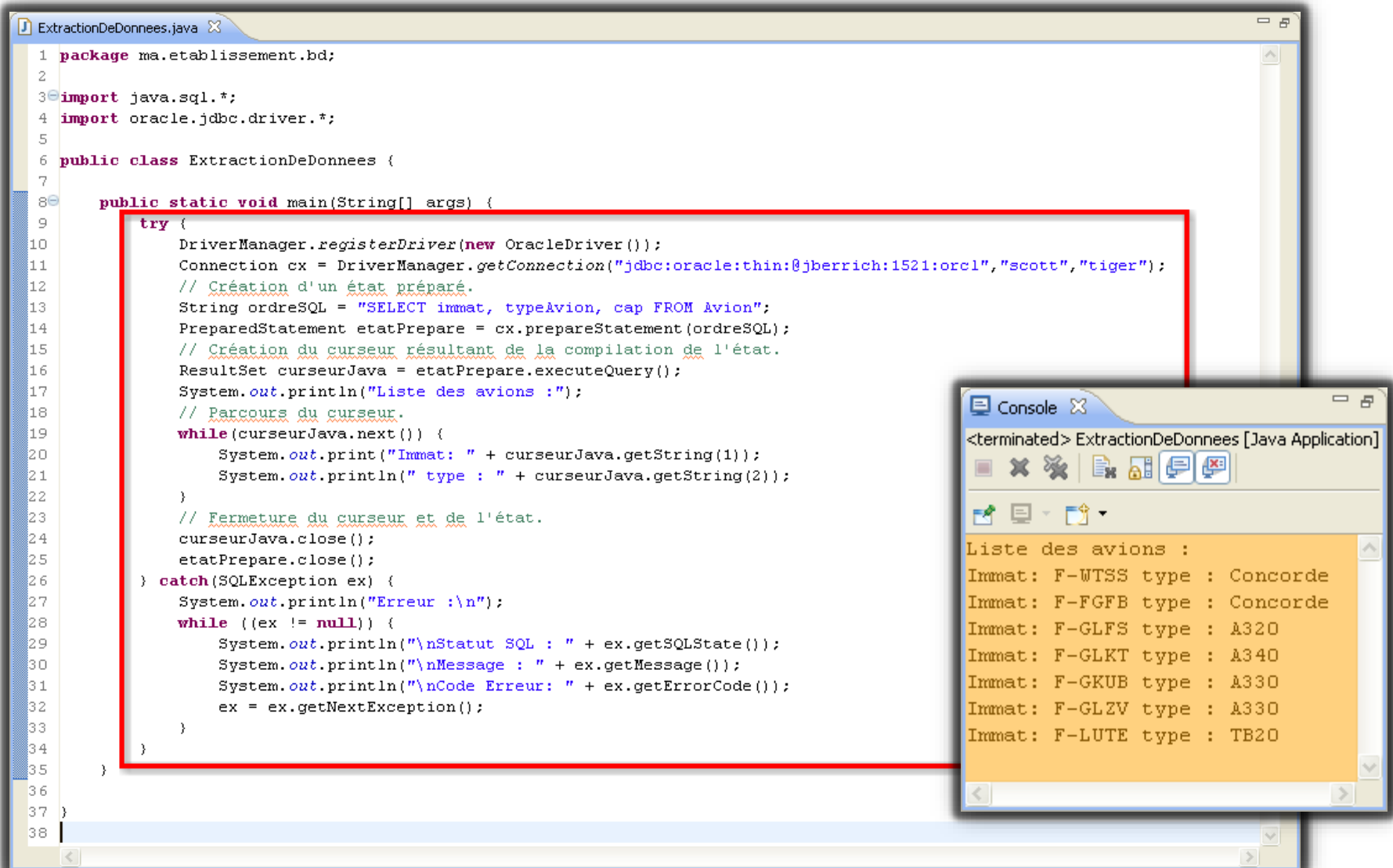
1 / 1 page

[Oracle] 7 / 7 rows [res:0.1sec] DDL Define

Interface PreparedStatement :

Méthode	Description
<code>ResultSet executeQuery()</code>	Exécute la requête et retourne un curseur ni navigable, ni modifiable par défaut.
<code>int executeUpdate()</code>	Exécute une instruction <i>LMD</i> (<i>INSERT</i> , <i>UPDATE</i> ou <i>DELETE</i>) et retourne le nombre de lignes traitées ou 0 pour les instructions SQL ne retourne aucun résultat (<i>LDD</i>).
<code>boolean execute()</code>	Exécute une instruction SQL et renvoie <i>true</i> , si c'est une instruction <i>SELECT</i> , <i>false</i> sinon.
<code>void setNull(int, int)</code>	Affecte la valeur <i>NULL</i> au paramètre de numéro et de type (classification java.sql.Type) spécifiés.
<code>void close()</code>	Ferme l'état.

Exemple :



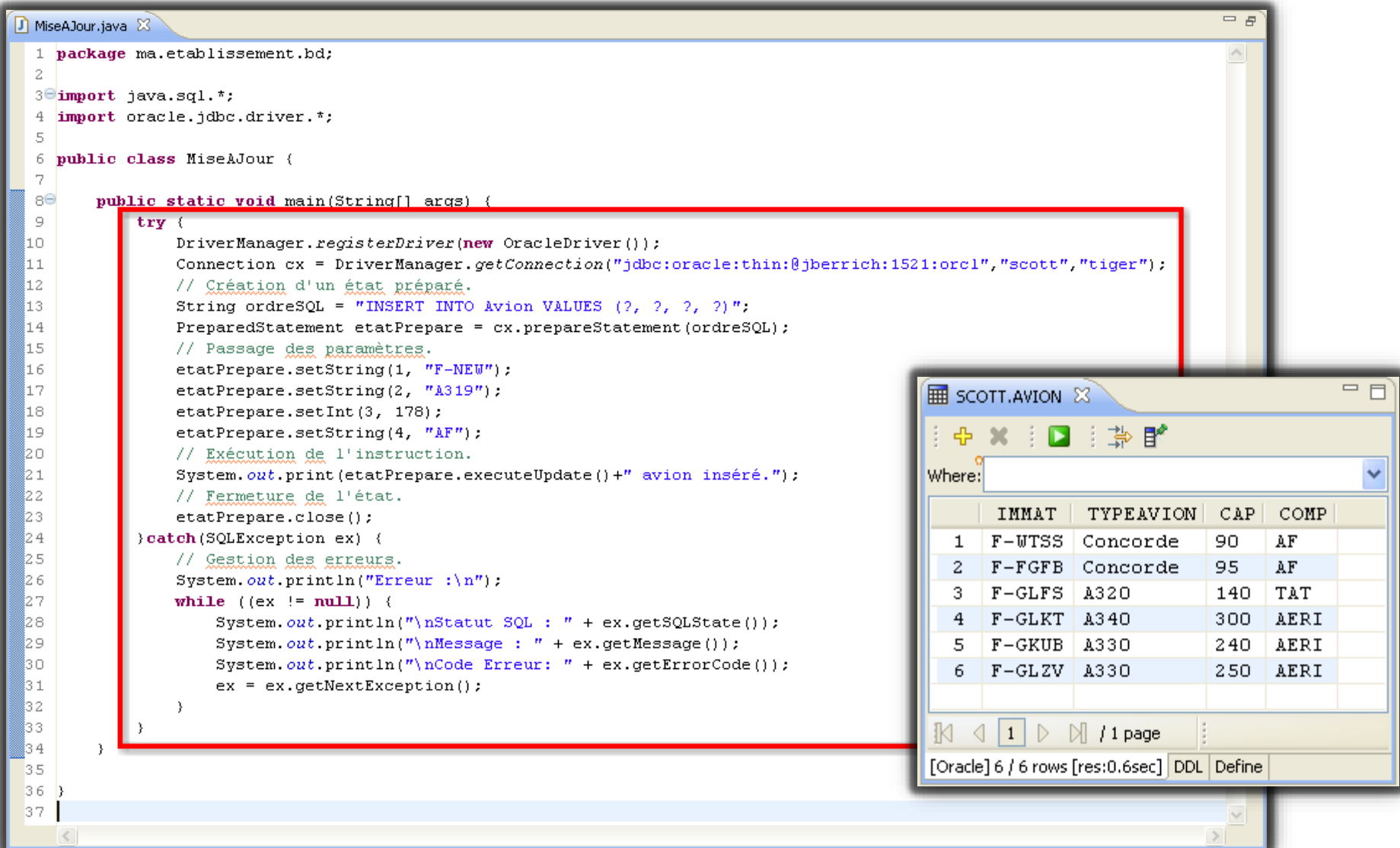
The image shows a Java IDE with two windows. The main window displays the source code for `ExtractionDeDonnees.java`. The code is a Java application that connects to an Oracle database, queries a table named `Avion`, and prints the results. A red rectangle highlights the `main` method. The console window shows the output of the program, which is a list of aircraft.

```
1 package ma.etablissement.bd;
2
3 import java.sql.*;
4 import oracle.jdbc.driver.*;
5
6 public class ExtractionDeDonnees {
7
8     public static void main(String[] args) {
9         try {
10             DriverManager.registerDriver(new OracleDriver());
11             Connection cx = DriverManager.getConnection("jdbc:oracle:thin:@jberrich:1521:orcl","scott","tiger");
12             // Cr ation d'un  tat pr par .
13             String ordreSQL = "SELECT immat, typeAvion, cap FROM Avion";
14             PreparedStatement etatPrepare = cx.prepareStatement(ordreSQL);
15             // Cr ation du curseur r sultant de la compilation de l' tat.
16             ResultSet curseurJava = etatPrepare.executeQuery();
17             System.out.println("Liste des avions :");
18             // Parcours du curseur.
19             while (curseurJava.next()) {
20                 System.out.print("Immat: " + curseurJava.getString(1));
21                 System.out.println(" type : " + curseurJava.getString(2));
22             }
23             // Fermeture du curseur et de l' tat.
24             curseurJava.close();
25             etatPrepare.close();
26         } catch (SQLException ex) {
27             System.out.println("Erreur :\n");
28             while ((ex != null)) {
29                 System.out.println("\nStatut SQL : " + ex.getSQLState());
30                 System.out.println("\nMessage : " + ex.getMessage());
31                 System.out.println("\nCode Erreur: " + ex.getErrorCode());
32                 ex = ex.getNextException();
33             }
34         }
35     }
36 }
37
38 }
```

Console Output:

```
<terminated> ExtractionDeDonnees [Java Application]
Liste des avions :
Immat: F-WTSS type : Concorde
Immat: F-FGFB type : Concorde
Immat: F-GLFS type : A320
Immat: F-GLKT type : A340
Immat: F-GKUB type : A330
Immat: F-GLZV type : A330
Immat: F-LUTE type : TB20
```

Exemple :



The image shows a Java IDE window titled 'MiseAJour.java' with the following code:

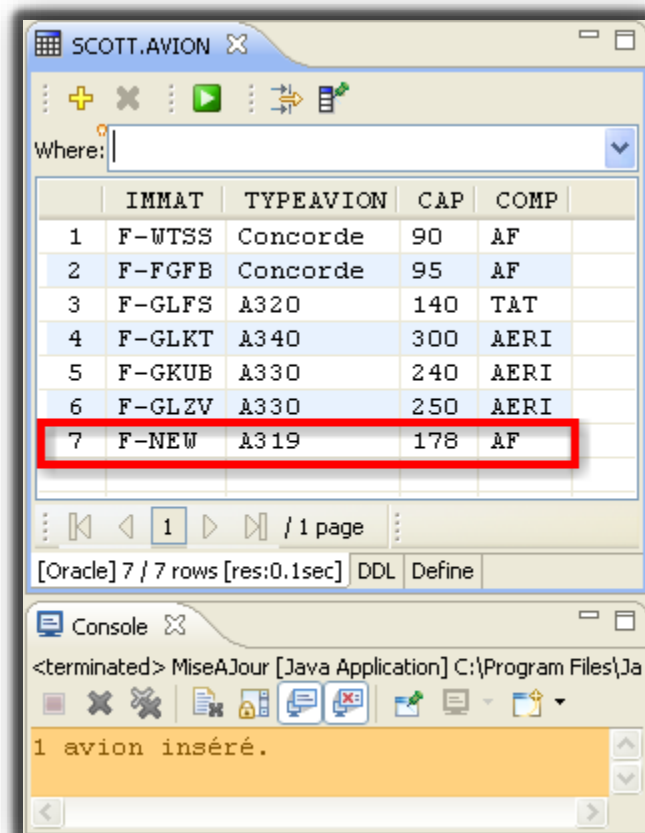
```
1 package ma.etablissement.bd;
2
3 import java.sql.*;
4 import oracle.jdbc.driver.*;
5
6 public class MiseAJour {
7
8     public static void main(String[] args) {
9         try {
10             DriverManager.registerDriver(new OracleDriver());
11             Connection cx = DriverManager.getConnection("jdbc:oracle:thin:@jberrich:1521:orcl","scott","tiger");
12             // Création d'un état préparé.
13             String ordreSQL = "INSERT INTO Avion VALUES (?, ?, ?, ?)";
14             PreparedStatement etatPrepare = cx.prepareStatement(ordreSQL);
15             // Passage des paramètres.
16             etatPrepare.setString(1, "F-NEW");
17             etatPrepare.setString(2, "A319");
18             etatPrepare.setInt(3, 178);
19             etatPrepare.setString(4, "AF");
20             // Exécution de l'instruction.
21             System.out.print(etatPrepare.executeUpdate()+" avion inséré.");
22             // Fermeture de l'état.
23             etatPrepare.close();
24         } catch (SQLException ex) {
25             // Gestion des erreurs.
26             System.out.println("Erreur :\n");
27             while ((ex != null)) {
28                 System.out.println("\nStatut SQL : " + ex.getSQLState());
29                 System.out.println("\nMessage : " + ex.getMessage());
30                 System.out.println("\nCode Erreur: " + ex.getErrorCode());
31                 ex = ex.getNextException();
32             }
33         }
34     }
35 }
36
37
```

A red rectangle highlights the try-catch block. In the background, another window titled 'SCOTT.AVION' displays a table of aircraft data:

	IMMAT	TYPEAVION	CAP	COMP
1	F-WTSS	Concorde	90	AF
2	F-FGFB	Concorde	95	AF
3	F-GLFS	A320	140	TAT
4	F-GLKT	A340	300	AERI
5	F-GKUB	A330	240	AERI
6	F-GLZV	A330	250	AERI

The window also shows a 'Where:' field, navigation buttons, and a status bar indicating '[Oracle] 6 / 6 rows [res:0.6sec] DDL Define'.

Exemple :



SCOTT.AVION

Where:

	IMMAT	TYPEAVION	CAP	COMP
1	F-WTSS	Concorde	90	AF
2	F-FGFB	Concorde	95	AF
3	F-GLFS	A320	140	TAT
4	F-GLKT	A340	300	AERI
5	F-GKUB	A330	240	AERI
6	F-GLZV	A330	250	AERI
7	F-NEW	A319	178	AF

[Oracle] 7 / 7 rows [res:0.1sec] DDL Define

Console

<terminated> MiseAJour [Java Application] C:\Program Files\Ja

1 avion inséré.

Exemple :

InstructionLDD.java

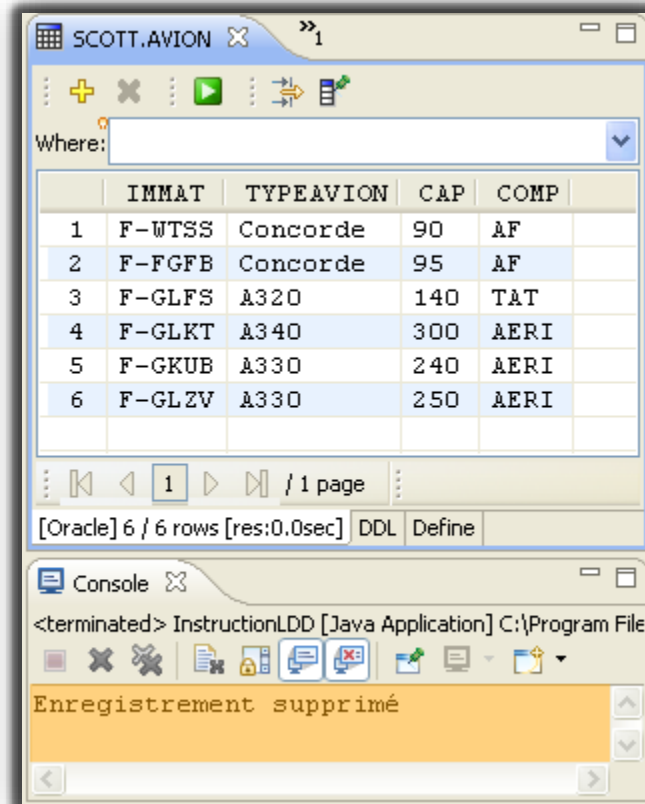
```
1 package ma.etablissement.bd;
2
3 import java.sql.*;
4 import oracle.jdbc.driver.*;
5
6 public class InstructionLDD {
7
8     public static void main(String[] args) {
9         try {
10             DriverManager.registerDriver(new OracleDriver());
11             Connection cx = DriverManager.getConnection("jdbc:oracle:thin:@jberrich:1521:orcl", "scott", "tiger");
12             // Création d'un état préparé.
13             String ordreSQL = "DELETE FROM Avion WHERE immat = ?";
14             PreparedStatement etatPrepare = cx.prepareStatement(ordreSQL);
15             // Passage des paramètres.
16             etatPrepare.setString(1, "F-NEW ");
17             // Exécution de l'instruction.
18             if(!etatPrepare.execute()) {
19                 System.out.println("Enregistrement supprimé");
20                 cx.commit();
21             }
22             // Fermeture de l'état.
23             etatPrepare.close();
24         } catch (SQLException ex) {
25             // Gestion des erreurs.
26             System.out.println("Erreur : \n");
27             while ((ex != null)) {
28                 System.out.println("\nStatut SQL : " + ex.getSQLState());
29                 System.out.println("\nMessage : " + ex.getMessage());
30                 System.out.println("\nCode Erreur : " + ex.getErrorCode());
31                 ex = ex.getNextException();
32             }
33         }
34     }
35 }
36
37
```

SCOTT.AVION

	IMMAT	TYPEAVION	CAP	COMP
1	F-WTSS	Concorde	90	AF
2	F-FGFB	Concorde	95	AF
3	F-GLFS	A320	140	TAT
4	F-GLKT	A340	300	AERI
5	F-GKUB	A330	240	AERI
6	F-GLZV	A330	250	AERI
7	F-NEW	A319	178	AF

[Oracle] 7 / 7 rows [res:0.0sec] DDL Define

Exemple :



The screenshot shows a database application window titled "SCOTT.AVION". It features a toolbar with icons for adding, deleting, and executing queries. Below the toolbar is a "Where:" filter field. The main area displays a table with 6 rows and 5 columns: IMMAT, TYPEAVION, CAP, and COMP. The table data is as follows:

	IMMAT	TYPEAVION	CAP	COMP
1	F-WTSS	Concorde	90	AF
2	F-FGFB	Concorde	95	AF
3	F-GLFS	A320	140	TAT
4	F-GLKT	A340	300	AERI
5	F-GKUB	A330	240	AERI
6	F-GLZV	A330	250	AERI

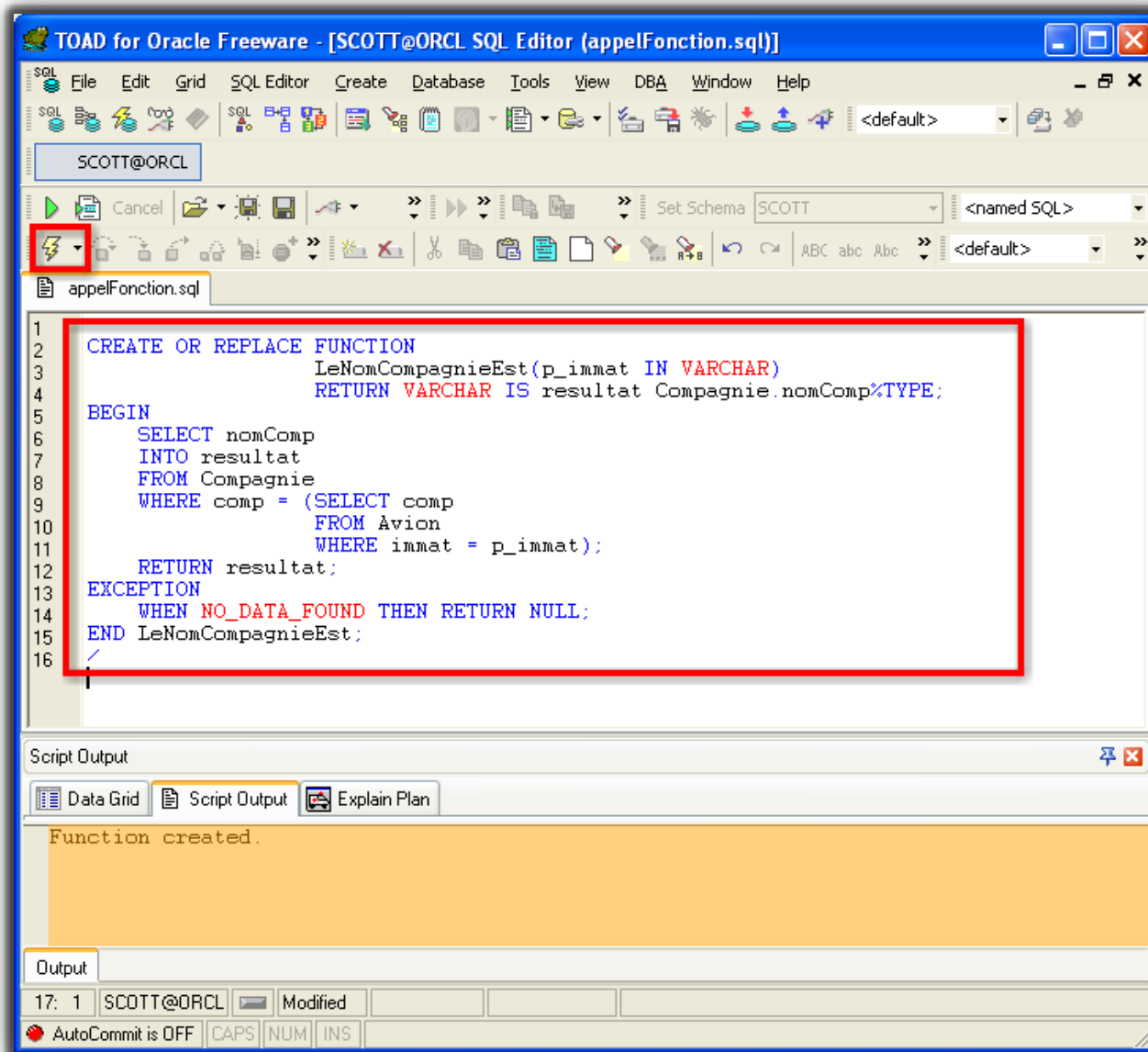
Below the table is a navigation bar with buttons for first, previous, next, and last records, and a page indicator showing "1 / 1 page". Below the navigation bar is a status bar showing "[Oracle] 6 / 6 rows [res:0.0sec]" and buttons for "DDL" and "Define". At the bottom of the window is a "Console" window showing the message "<terminated> InstructionLDD [Java Application] C:\Program File" and a yellow highlighted message "Enregistrement supprimé".

Interface CallableStatement :

Type du sous-programme	Paramètre
Fonction	{? = call nomFonction({=[?, ?, ...]})}
Procédure	{call nomProcédure({=[?, ?, ...]})}

Méthode	Description
ResultSet executeQuery()	Idem <i>PreparedStatement</i> .
int executeUpdate()	Idem <i>PreparedStatement</i> .
boolean execute()	Idem <i>PreparedStatement</i> .
void registerOutParameter(int, int)	Transfère un paramètre de sortie à un indice donné d'un type Java.
boolean wasNull()	Détermine si le dernier paramètre de sortie extrait est à <i>NULL</i> . Cette méthode doit être seulement invoquée après une méthode de type <i>getxxx</i> .

Exemple :



Exemple :

```
AppelFonction.java X
1 package ma.etablissement.bd;
2
3 import java.sql.*;
4
5
6 public class AppelFonction {
7
8     public static void main(String[] args) {
9         try{
10             DriverManager.registerDriver(new OracleDriver());
11             Connection cx = DriverManager.getConnection("jdbc:oracle:thin:@jberrich:1521:orcl","scott","tiger");
12             // Création d'un état callable.
13             String ordreSQL = "{? = call LeNomCompagnieEst(?)}";
14             CallableStatement etatAppelable = cx.prepareCall(ordreSQL);
15             // Déclaration du paramètre de sortie.
16             etatAppelable.registerOutParameter(1, java.sql.Types.VARCHAR);
17             // Passage du paramètre d'entrée.
18             etatAppelable.setString(2, "F-GLFS");
19             System.out.println("Avant appel :");
20             // Exécution de la fonction.
21             etatAppelable.execute();
22             // Extraction du résultat.
23             System.out.println("Nom de la compagnie de F-GLFS : " + etatAppelable.getString(1));
24             // Fermeture de la connexion.
25             cx.close();
26         } catch (SQLException ex) {
27             System.out.println("Erreur :\n");
28             while ((ex != null)) {
29                 System.out.println("\nStatut SQL : " + ex.getSQLState());
30                 System.out.println("\nMessage : " + ex.getMessage());
31                 System.out.println("\nCode Erreur : " + ex.getErrorCode());
32                 ex = ex.getNextException();
33             }
34         }
35     }
36 }
37
38
```

Exemple :

The screenshot displays the TOAD for Oracle Freeware interface. The main window is titled "TOAD for Oracle Freeware - [SCOTT@ORCL SQL Editor (<No name>)]". The menu bar includes File, Edit, Grid, SQL Editor, Create, Database, Tools, View, DBA, Window, and Help. The toolbar contains various icons for file operations, execution, and database management. The SQL Editor shows a PL/SQL script with the following code:

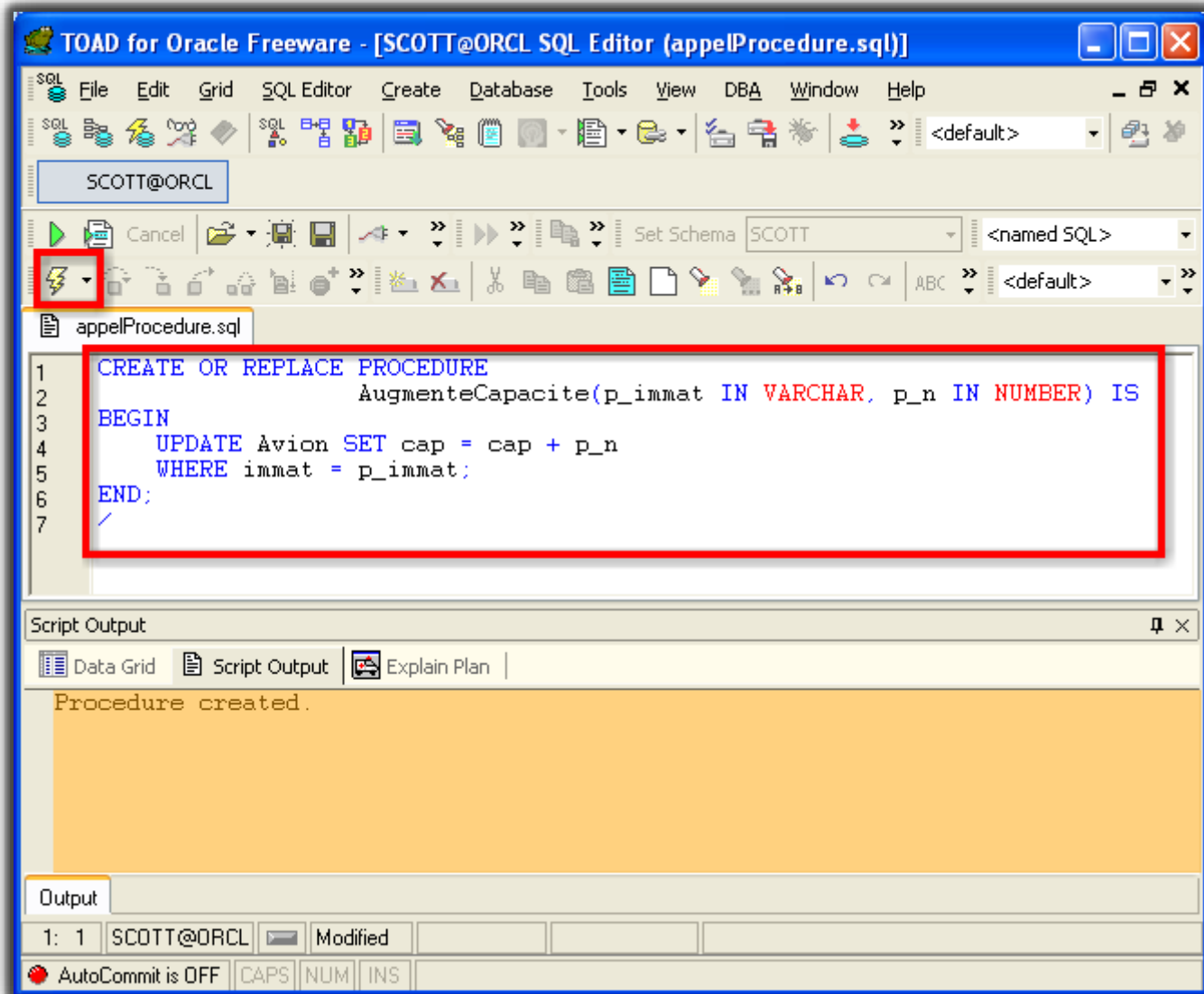
```
1 SET SERVEROUTPUT ON
2
3 DECLARE
4   n VARCHAR(25);
5 BEGIN
6   n := LeNomCompagnieEst('F-GLFS');
7   DBMS_OUTPUT.put_line('Nom de la compagnie de F-GLFS : ' || n);
8 END;
9 /
10
```

The script is executed, and the output is displayed in the "Script Output" pane. The output shows the company name "Nom de la compagnie de F-GLFS : Transport Air Tour" and the message "PL/SQL procedure successfully completed.".

A "Console" window is also open, showing the execution of the Java application "AppelFonction [Java Application] C:\Program Files\Java\jre6\bin\javaw.exe". The console output shows the text "Avant appel :" and "Nom de la compagnie de F-GLFS : Transport Air Tour".

The "Output" pane at the bottom shows the current session information: "1: 1 SCOTT@ORCL Modified". The status bar indicates "AutoCommit is OFF" and shows the current session name "CAPS" and the current user "NUM".

Example :



Exemple :

```
AppelProcedure.java
1 package ma.etablissement.bd;
2
3 import java.sql.*;
4
5
6 public class AppelProcedure {
7
8     public static void main(String[] args) {
9         try {
10             DriverManager.registerDriver(new OracleDriver());
11             Connection cx = DriverManager.getConnection("jdbc:oracle:thin:@jberrich:1521:orcl","scott","tiger");
12             // Création d'un état callable.
13             String ordreSQL = "{call AugmenteCapacite(?,?)}";
14             System.out.println("Avant prépare :");
15             CallableStatement etatAppelable = cx.prepareCall(ordreSQL);
16             // Passage du paramètre d'entrée.
17             etatAppelable.setString(1,"F-GLFS");
18             etatAppelable.setInt(2,50);
19             System.out.println("Avant appel :");
20             // Exécution de la fonction.
21             etatAppelable.execute();
22             etatAppelable.close();
23             // Fermeture de la connexion.
24             cx.close();
25         } catch (SQLException ex) {
26             System.out.println("Erreur :\n");
27             while ((ex != null)) {
28                 System.out.println("\nStatut SQL : " + ex.getSQLState());
29                 System.out.println("\nMessage : " + ex.getMessage());
30                 System.out.println("\nCode Erreur : " + ex.getErrorCode());
31                 ex = ex.getNextException();
32             }
33         }
34     }
35 }
36
37
```

Exemple :

SCOTT.AVION

Where:

	IMMAT	TYPEAVION	CAP	COMP
1	F-WTSS	Concorde	90	AF
2	F-FGFB	Concorde	95	AF
3	F-GLFS	A320	190	TAT
4	F-GLKT	A340	300	AERI
5	F-GKUB	A330	240	AERI
6	F-GLZV	A330	250	AERI

[Oracle] 6 / 6 rows [res:0.0sec] DDL Define

Console

<terminated> AppelProcedure [Java Application] C:\Program File

Avant prepare :
Avant appel :

SCOTT.AVION

Where:

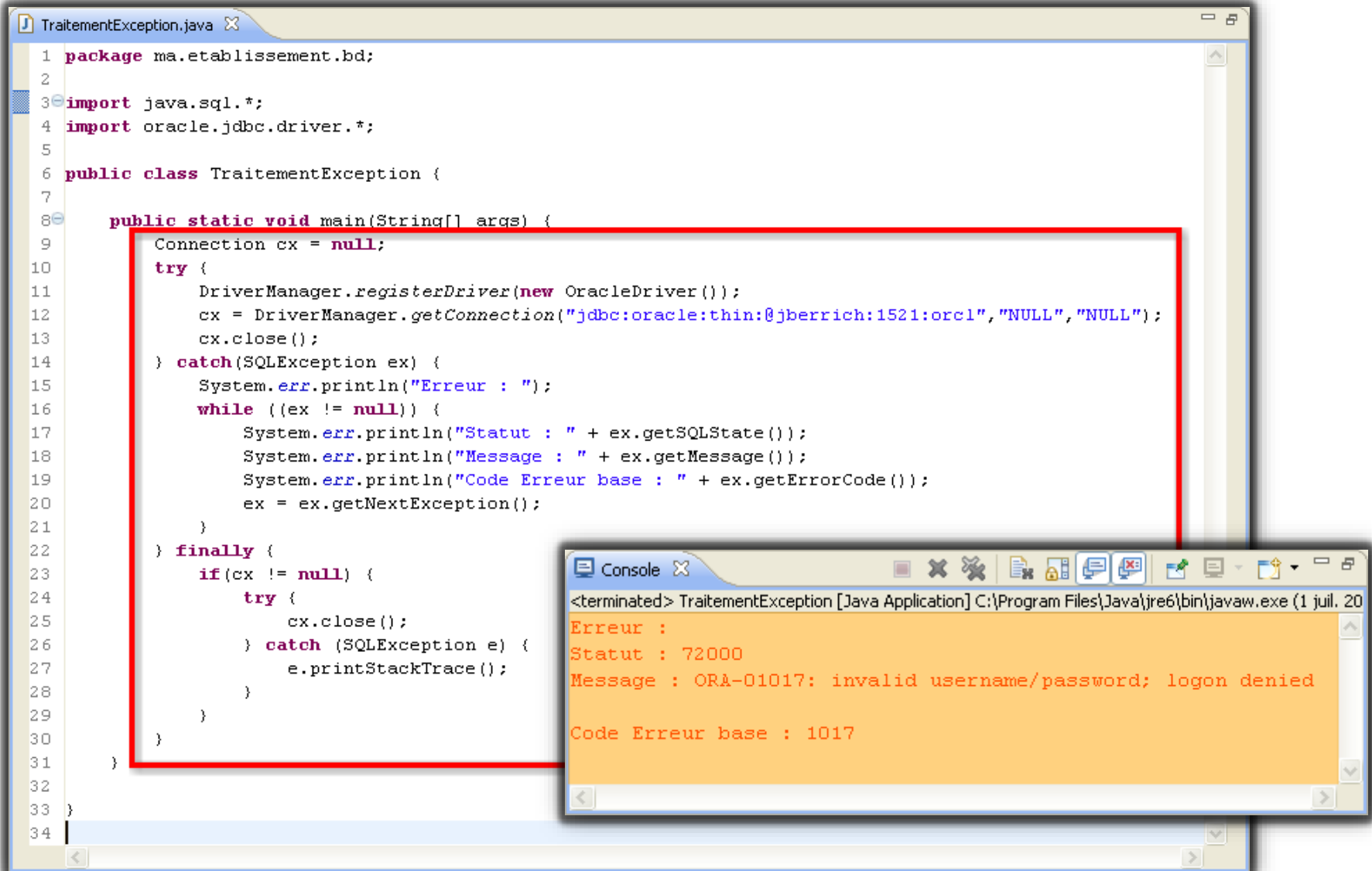
	IMMAT	TYPEAVION	CAP	COMP
1	F-WTSS	Concorde	90	AF
2	F-FGFB	Concorde	95	AF
3	F-GLFS	A320	140	TAT
4	F-GLKT	A340	300	AERI
5	F-GKUB	A330	240	AERI
6	F-GLZV	A330	250	AERI

[Oracle] 6 / 6 rows [res:0.7sec] DDL Define

Traitement des exceptions :

Méthode	Description
<code>String getMessage()</code>	Message décrivant l'erreur.
<code>String getSQLState()</code>	Code erreur SQL Standard.
<code>int getErrorCode()</code>	Code erreur SQL de la base.
<code>SQLException getNextException()</code>	Chaînage à l'exception suivante (si une erreur renvoie plusieurs messages).

Exemple :



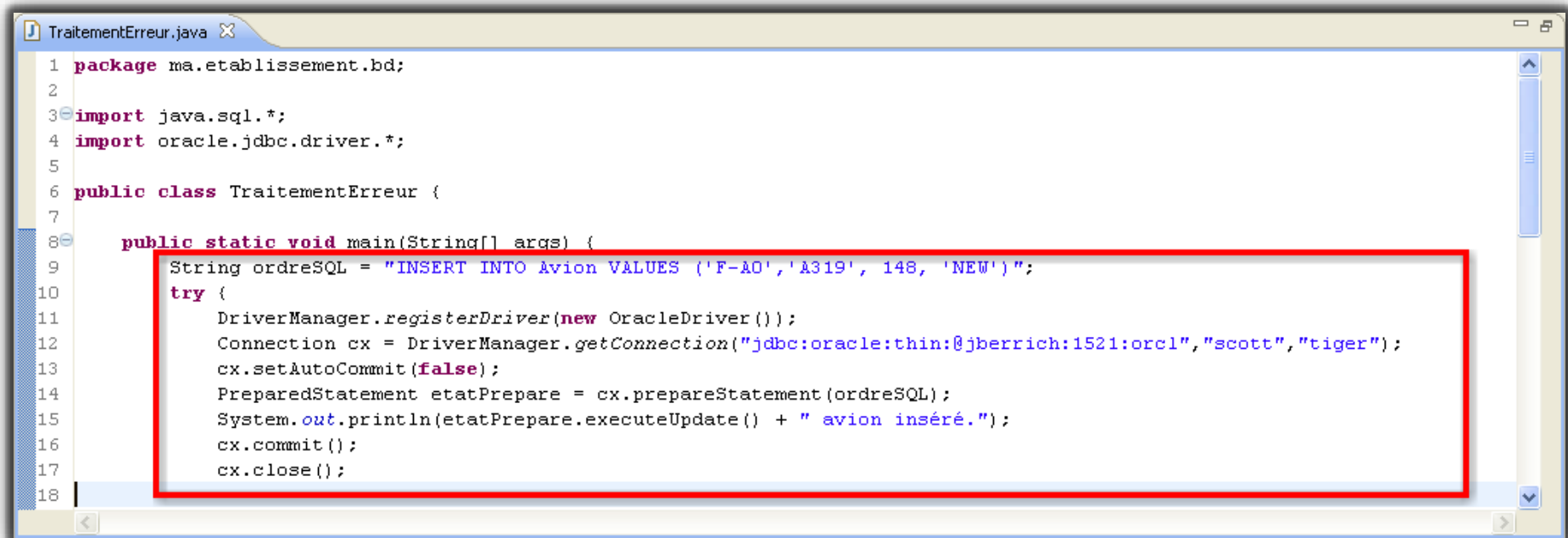
The screenshot shows a Java IDE with a file named `TraitementException.java` open. The code defines a package `ma.etablissement.bd`, imports `java.sql.*` and `oracle.jdbc.driver.*`, and defines a public class `TraitementException`. The `main` method attempts to establish a database connection using `DriverManager`. It registers the `OracleDriver`, gets a connection with the URL `jdbc:oracle:thin:@jberrich:1521:orcl`, and sets the username to `NULL` and password to `NULL`. It then enters a `try` block where it catches a `SQLException` and prints the error details: `Erreur :`, `Statut :`, `Message :`, and `Code Erreur base :`. The `finally` block closes the connection if it was not null. The console output shows the error details for the connection attempt.

```
1 package ma.etablissement.bd;
2
3 import java.sql.*;
4 import oracle.jdbc.driver.*;
5
6 public class TraitementException {
7
8     public static void main(String[] args) {
9         Connection cx = null;
10        try {
11            DriverManager.registerDriver(new OracleDriver());
12            cx = DriverManager.getConnection("jdbc:oracle:thin:@jberrich:1521:orcl", "NULL", "NULL");
13            cx.close();
14        } catch (SQLException ex) {
15            System.err.println("Erreur : ");
16            while ((ex != null)) {
17                System.err.println("Statut : " + ex.getSQLState());
18                System.err.println("Message : " + ex.getMessage());
19                System.err.println("Code Erreur base : " + ex.getErrorCode());
20                ex = ex.getNextException();
21            }
22        } finally {
23            if (cx != null) {
24                try {
25                    cx.close();
26                } catch (SQLException e) {
27                    e.printStackTrace();
28                }
29            }
30        }
31    }
32 }
33
34
```

Console Output:

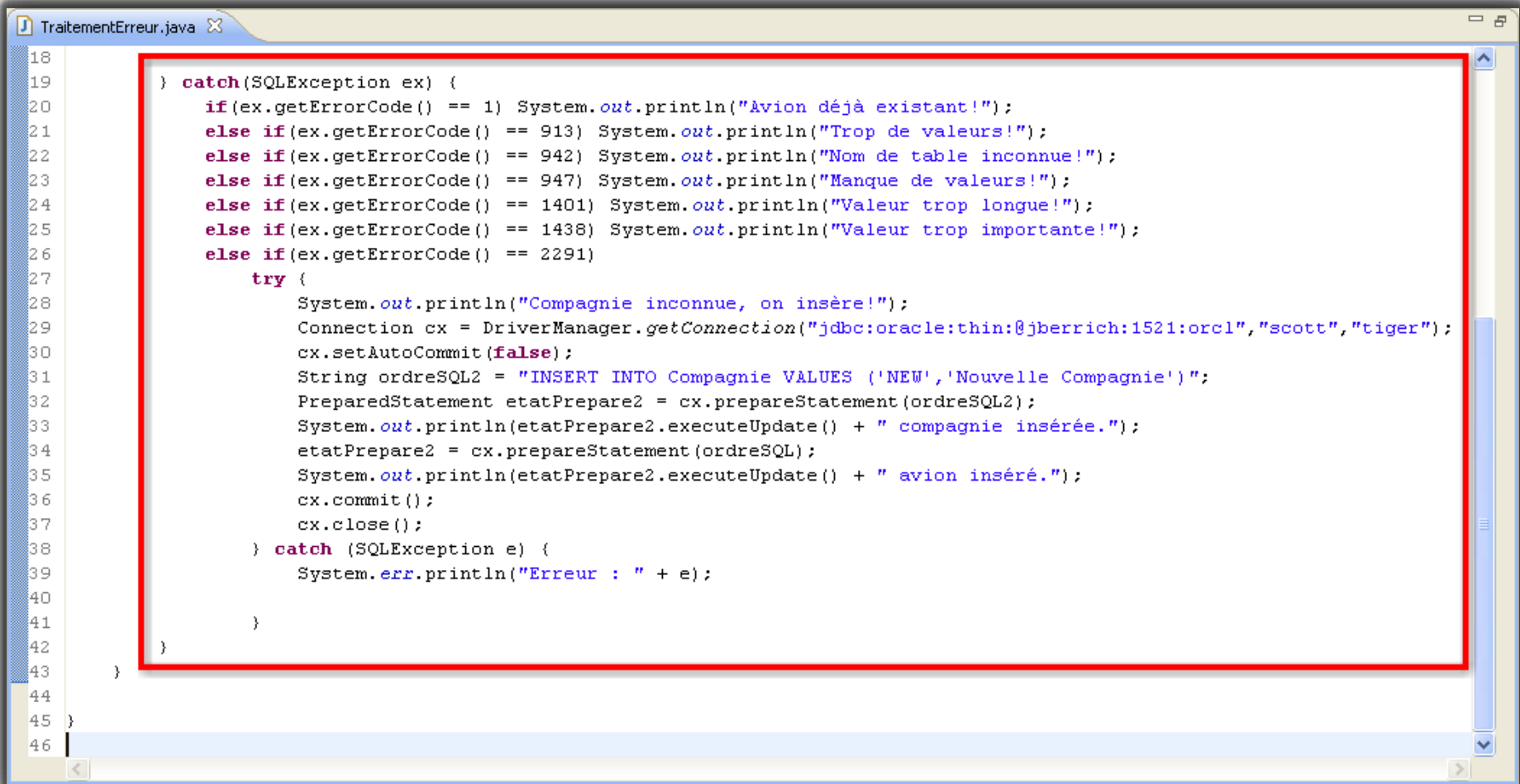
```
<terminated> TraitementException [Java Application] C:\Program Files\Java\jre6\bin\javaw.exe (1 juil. 20
Erreur :
Statut : 72000
Message : ORA-01017: invalid username/password; logon denied
Code Erreur base : 1017
```

Exemple :



```
TraitementErreur.java
1 package ma.etablissement.bd;
2
3 import java.sql.*;
4 import oracle.jdbc.driver.*;
5
6 public class TraitementErreur {
7
8     public static void main(String[] args) {
9         String ordreSQL = "INSERT INTO Avion VALUES ('F-AO','A319', 148, 'NEW')";
10        try {
11            DriverManager.registerDriver(new OracleDriver());
12            Connection cx = DriverManager.getConnection("jdbc:oracle:thin:@jberrich:1521:orcl","scott","tiger");
13            cx.setAutoCommit(false);
14            PreparedStatement etatPrepare = cx.prepareStatement(ordreSQL);
15            System.out.println(etatPrepare.executeUpdate() + " avion inséré.");
16            cx.commit();
17            cx.close();
18        }
```

Exemple :



```
18
19 } catch(SQLException ex) {
20     if(ex.getErrorCode() == 1) System.out.println("Avion déjà existant!");
21     else if(ex.getErrorCode() == 913) System.out.println("Trop de valeurs!");
22     else if(ex.getErrorCode() == 942) System.out.println("Nom de table inconnue!");
23     else if(ex.getErrorCode() == 947) System.out.println("Manque de valeurs!");
24     else if(ex.getErrorCode() == 1401) System.out.println("Valeur trop longue!");
25     else if(ex.getErrorCode() == 1438) System.out.println("Valeur trop importante!");
26     else if(ex.getErrorCode() == 2291)
27         try {
28             System.out.println("Compagnie inconnue, on insère!");
29             Connection cx = DriverManager.getConnection("jdbc:oracle:thin:@jberrich:1521:orcl","scott","tiger");
30             cx.setAutoCommit(false);
31             String ordreSQL2 = "INSERT INTO Compagnie VALUES ('NEW','Nouvelle Compagnie')";
32             PreparedStatement etatPrepare2 = cx.prepareStatement(ordreSQL2);
33             System.out.println(etatPrepare2.executeUpdate() + " compagnie insérée.");
34             etatPrepare2 = cx.prepareStatement(ordreSQL);
35             System.out.println(etatPrepare2.executeUpdate() + " avion inséré.");
36             cx.commit();
37             cx.close();
38         } catch (SQLException e) {
39             System.err.println("Erreur : " + e);
40         }
41     }
42 }
43
44
45 }
46
```

Exemple :

The screenshot displays a database application interface with four windows. The top-left window shows the 'SCOTT.AVION' table with 7 rows. The top-right window shows the 'SCOTT.COMPAGNIE' table with 4 rows. The bottom-left window shows the 'SCOTT.AVION' table with 6 rows. The bottom-right window shows the 'SCOTT.COMPAGNIE' table with 3 rows. A console window at the bottom left displays error messages.

SCOTT.AVION Table (Top Left):

	IMMAT	TYPEAVION	CAP	COMP
1	F-WTSS	Concorde	90	AF
2	F-FGFB	Concorde	95	AF
3	F-GLFS	A320	140	TAT
4	F-GLKT	A340	300	AERI
5	F-GKUB	A330	240	AERI
6	F-GLZV	A330	250	AERI
7	F-AO	A319	148	NEW

SCOTT.COMPAGNIE Table (Top Right):

	COMP	NOMCOMP
1	AF	Air France
2	TAT	Transport Air Tour
3	AERI	Air AERIS Tlse
4	NEW	Nouvelle Compagnie

SCOTT.AVION Table (Bottom Left):

	IMMAT	TYPEAVION	CAP	COMP
1	F-WTSS	Concorde	90	AF
2	F-FGFB	Concorde	95	AF
3	F-GLFS	A320	140	TAT
4	F-GLKT	A340	300	AERI
5	F-GKUB	A330	240	AERI
6	F-GLZV	A330	250	AERI

SCOTT.COMPAGNIE Table (Bottom Right):

	COMP	NOMCOMP
1	AF	Air France
2	TAT	Transport Air Tour
3	AERI	Air AERIS Tlse

Console Window (Bottom Left):

```
<terminated> TraitementErreur [Java Application] C:\Progr  
Compagnie inconnue, on insère!  
1 compagnie insérée.  
1 avion inséré.
```