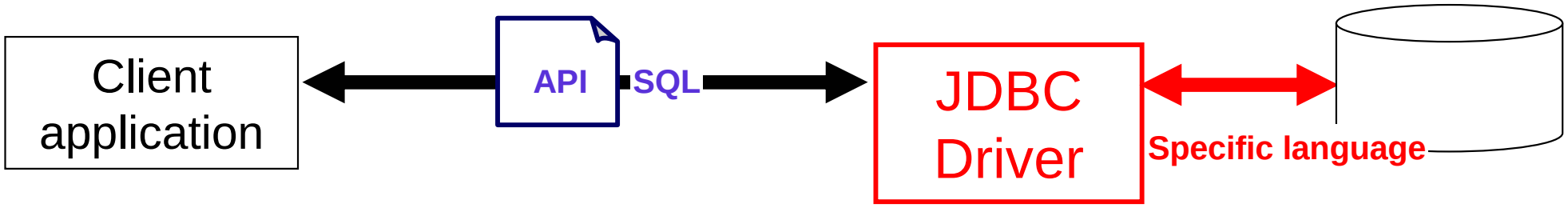
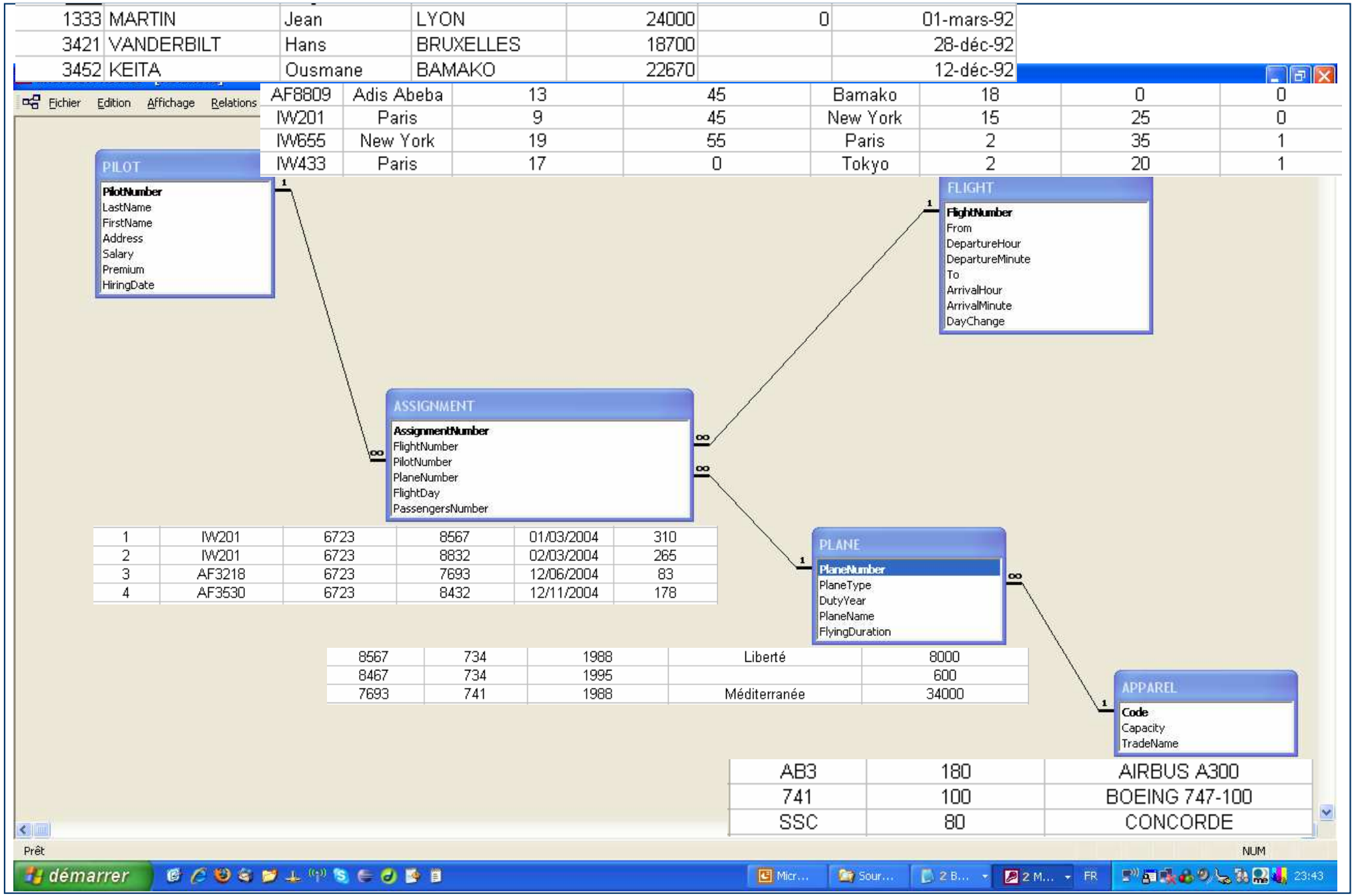
JDBC

- **JDBC: Java Data Base Connectivity**



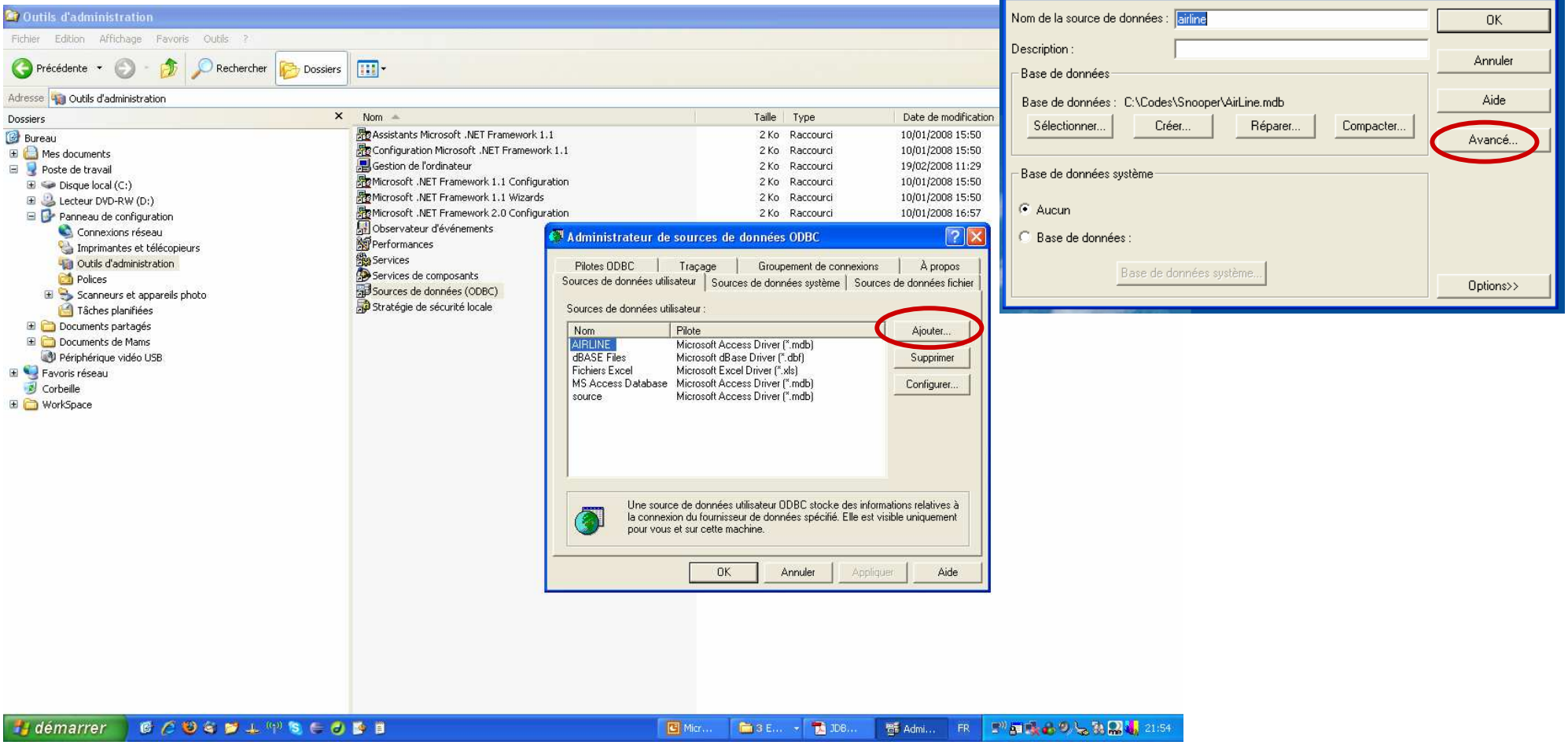
- **API: Java library ⇒ classes for connection to (and use of) DB**
- **Driver: provided by the DB constructor**
  - **Type 1: ⇔ ODBC_JDBC Bridge ⇔ ODBC Driver ⇔ DB**
  - **Type 2: ⇔ Proprietary JDBC ⇔ DB**
  - **Type 4: ⇔ 100% Java ⇔ DB**
  - **Type 3: ⇔ DB Server ⇔ Type 1/2/3 Driver ⇔ DB**

JDBC > Step 1: create Data Base



JDBC > Step 2: define Data Source Name

- Configuration panel > Administration tools > Data sources (ODBC)
- Add > Microsoft Access Driver
- Name the DSN and select the right DB
- Optional: in “Advanced”, define login and password

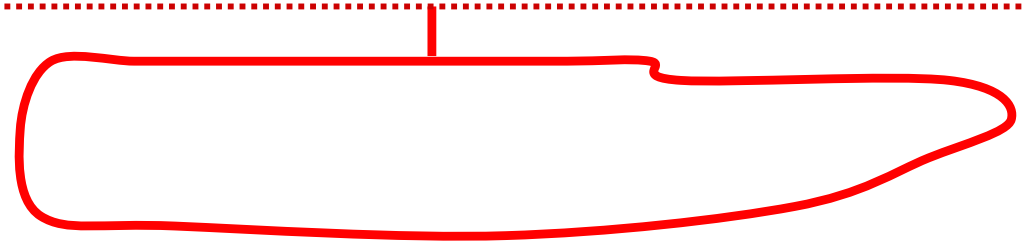


JDBC > Step 3: build application (DB updating)


```
import java.sql.*;                                   <-- API
class AirLiner {

    static Connection link;
    static String myURL = "jdbc:odbc:source";        <-- DSN

    public static void main(String args[ ]) throws SQLException {
            String query;
            int answer;
            try {
                    DriverManager.registerDriver(new sun.jdbc.odbc.JdbcOdbcDriver());
                    // Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
                    link = DriverManager.getConnection(myURL);
                    // link = DriverManager.getConnection(myURL,login,pwd);
            } catch (SQLException e) { System.out.println("Connection failed: "+e.getMessage()); }
            // catch (ClassNotFoundException e) { System.out.println("No driver! ";  }
            Statement question = link.createStatement();
            query = "UPDATE PILOT SET Premium = 0 WHERE Premium IS NULL";
            answer = question.executeUpdate(query);
             System.out.println(answer + "lines updated");
            question.close(); link.close();
    }

}
```

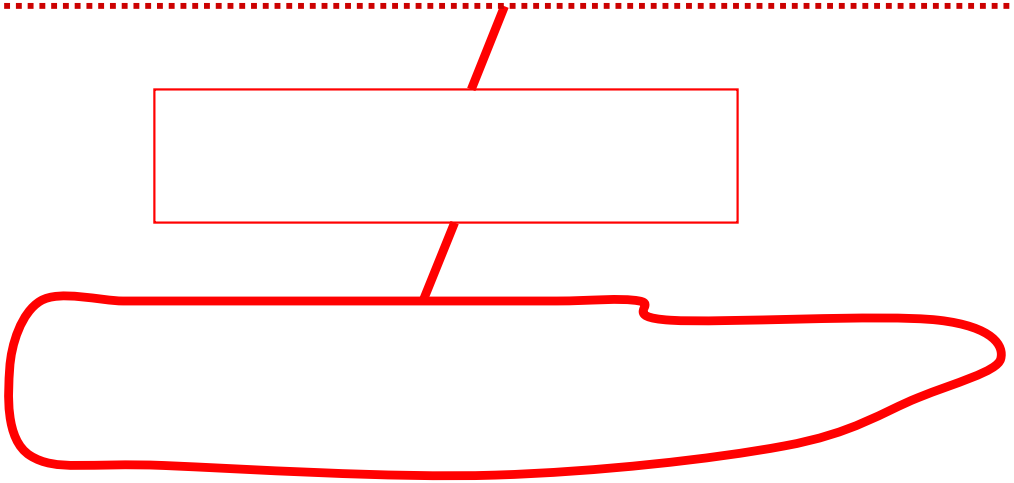
JDBC > Step 3': build application (DB querying)


```
Statement question = link.createStatement();
query = "SELECT * FROM PILOT";
ResultSet answer = question.executeQuery(query);
while (answer.next()!=false) {
        String pNb = answer.getString("PilotNumber");
        String pNm = answer.getString("LastName");
        String pAd = answer.getString("Address");
        float pSl = answer.getFloat("Salary");
        float pPr = answer.getFloat("Premium");
        Date pHd = answer.getDate("HiringDate");
        if (answer.wasNull() == false) {
                System.out.println("Pilot " + pNm);
                System.out.println("      matricule : " + pNb);
                System.out.println("      from: " + pAd);
                System.out.println("      salary: " + pSl);
                System.out.println("      since: " + pHd);
                if (pPr > 0 ) System.out.println("      premium: " + pPr + "\n");
                else System.out.println("      without premium\n");
        }
}
answer.close(); question.close(); link.close();
```

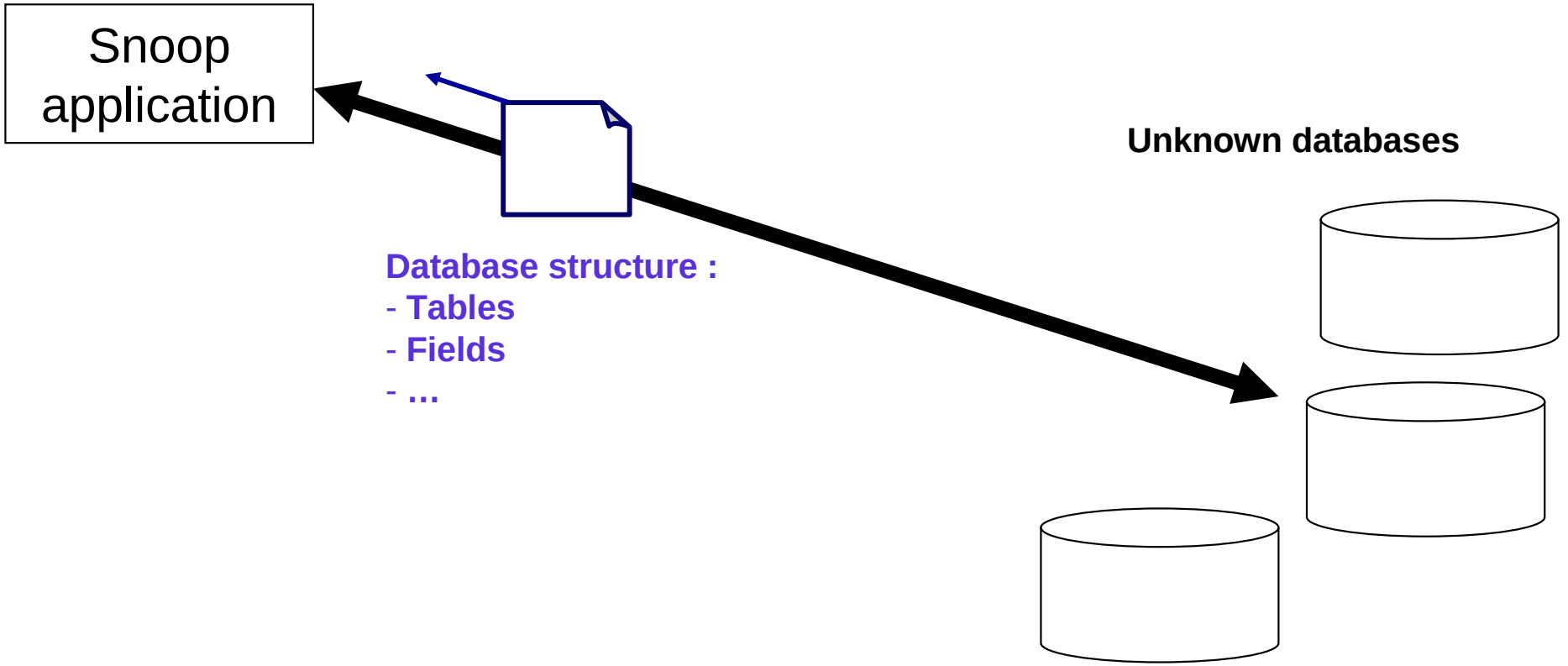


JDBC > Step 3'': build application (prepared query)


```
query = "SELECT * FROM PILOT WHERE Address=? AND Salary>?";
PreparedStatement question = link.prepareStatement(query);
question.setString(1,args[0]);
question.setInt(2, Integer.parseInt(args[1]));
answer = question.executeQuery();
```

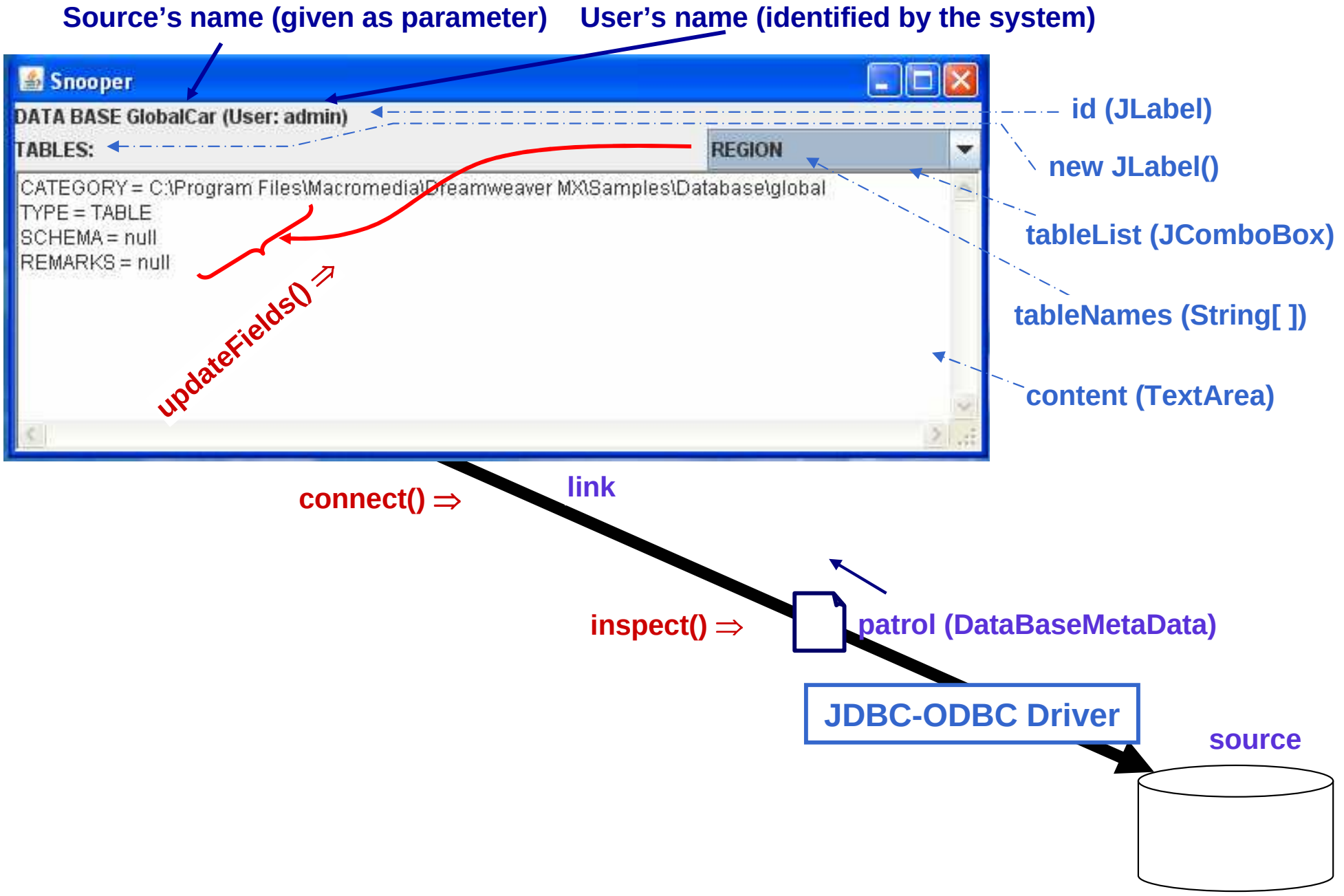


- **Snooper: connects to any database and get back information about it's structure**



- **Code available here: snooper class, airline database**
- **Deliverables: improved snooper**

JDBC > Snooper interface and behavior



- **Three improvements:**

1. **Only user's tables are displayed in the combo box:**
   - **all other tables are shown at the bottom of the interface and are gathered by categories**
2. **Fields of the selected table are also displayed in the content box:**
   - **fields information are: name, type and key status (not key, primary key, foreign key)**
3. **Application interface is user-friendly:**
   - **source is defined using a text field in the interface**
   - **Application is properly closed using a button**