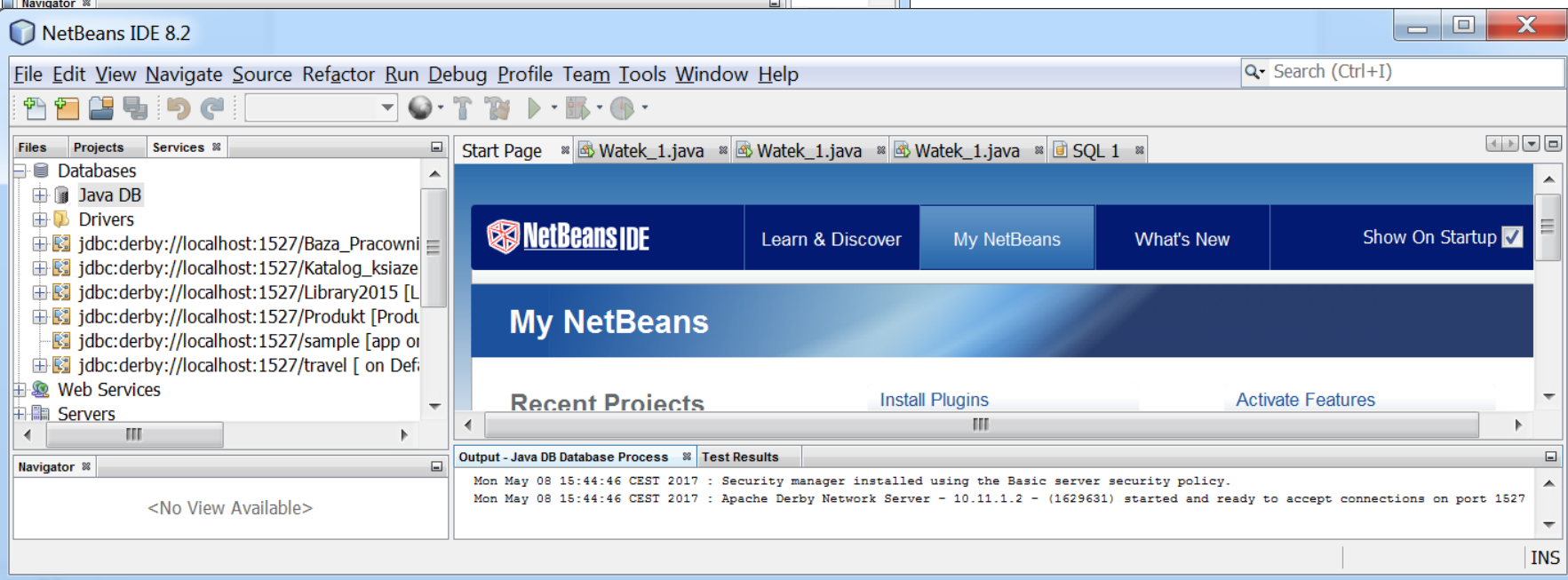
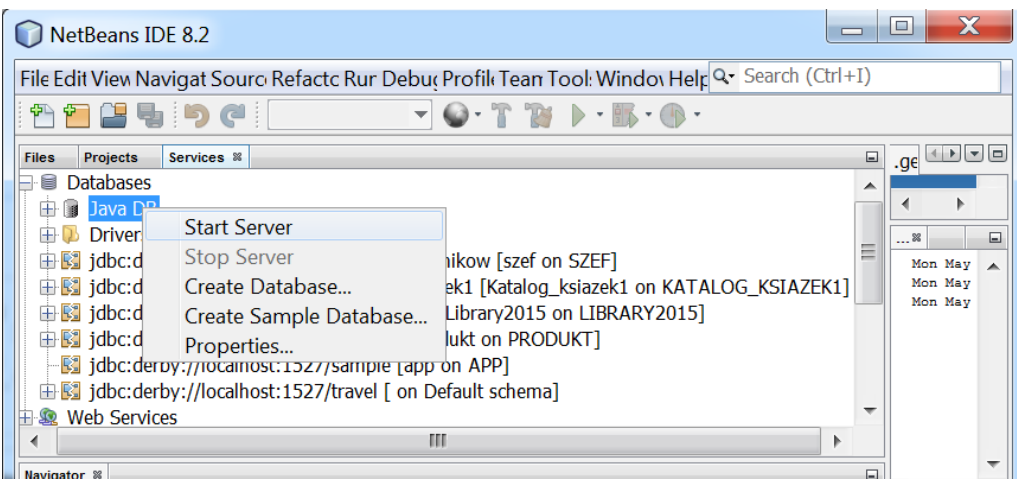


Protokół JDBC – współpraca z relacyjnymi bazami danych lab3

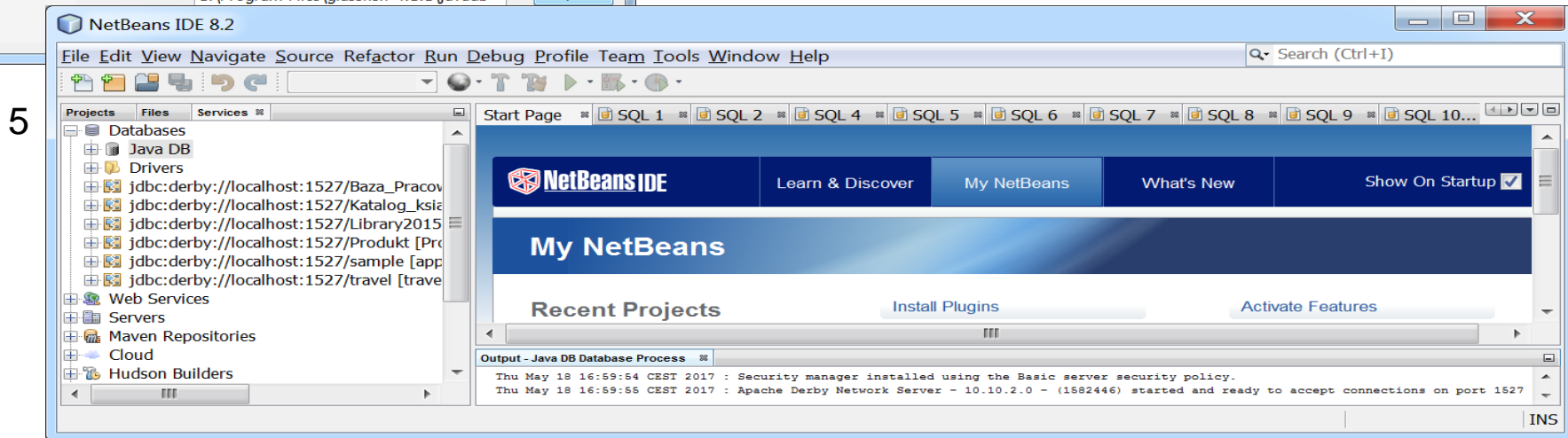
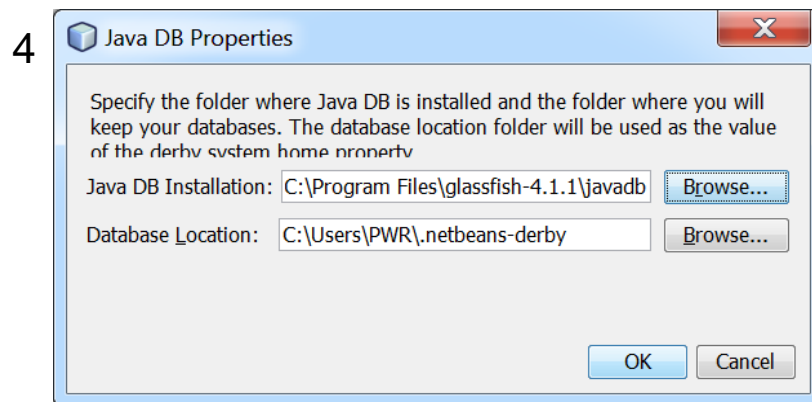
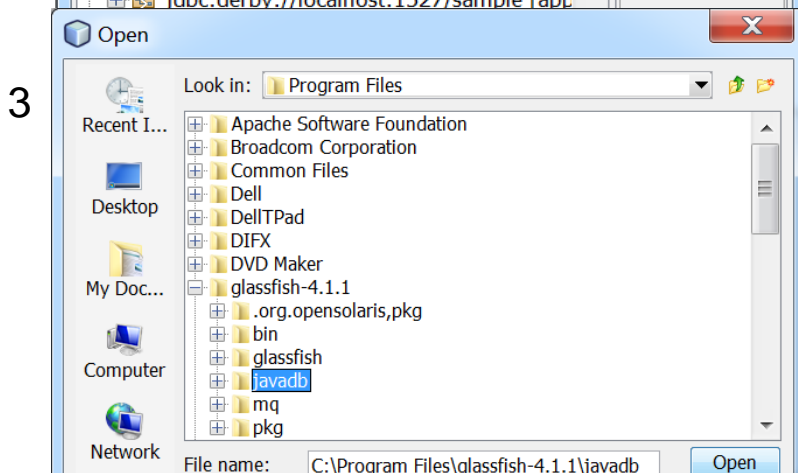
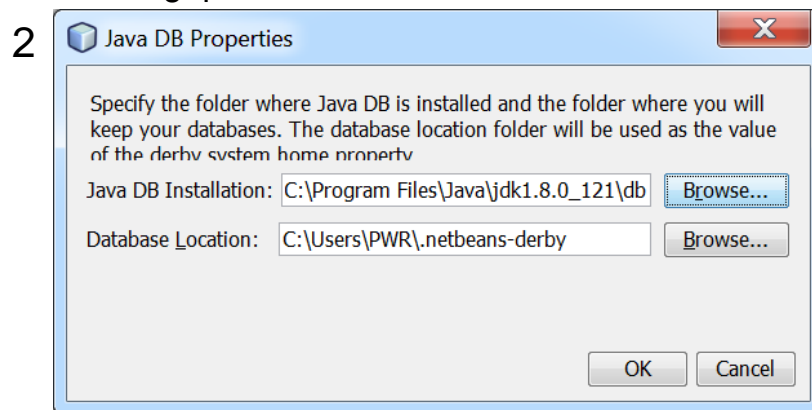
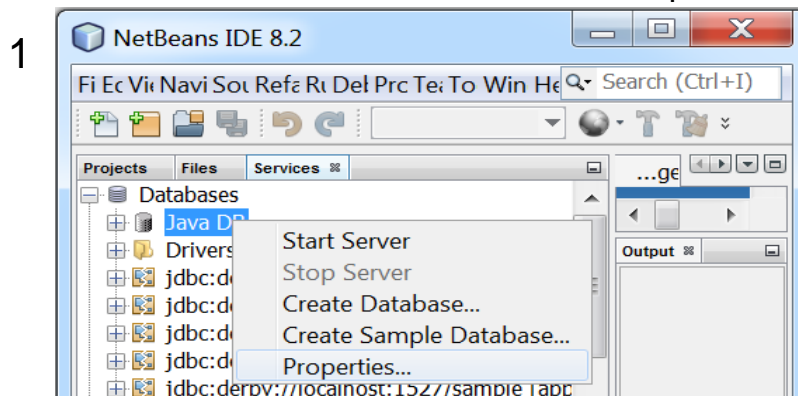
Dr inż. Zofia Kruczkiewicz
Programowanie aplikacji
internetowych

Zadanie1 – Połączenie z bazą danych Sample systemu bazodanowego Derby (metoda void polaczenie_z_baza()), wyświetlanie zawartości tabeli CUSTOMER (metoda ArrayList<String> dane_tablicy_osob())

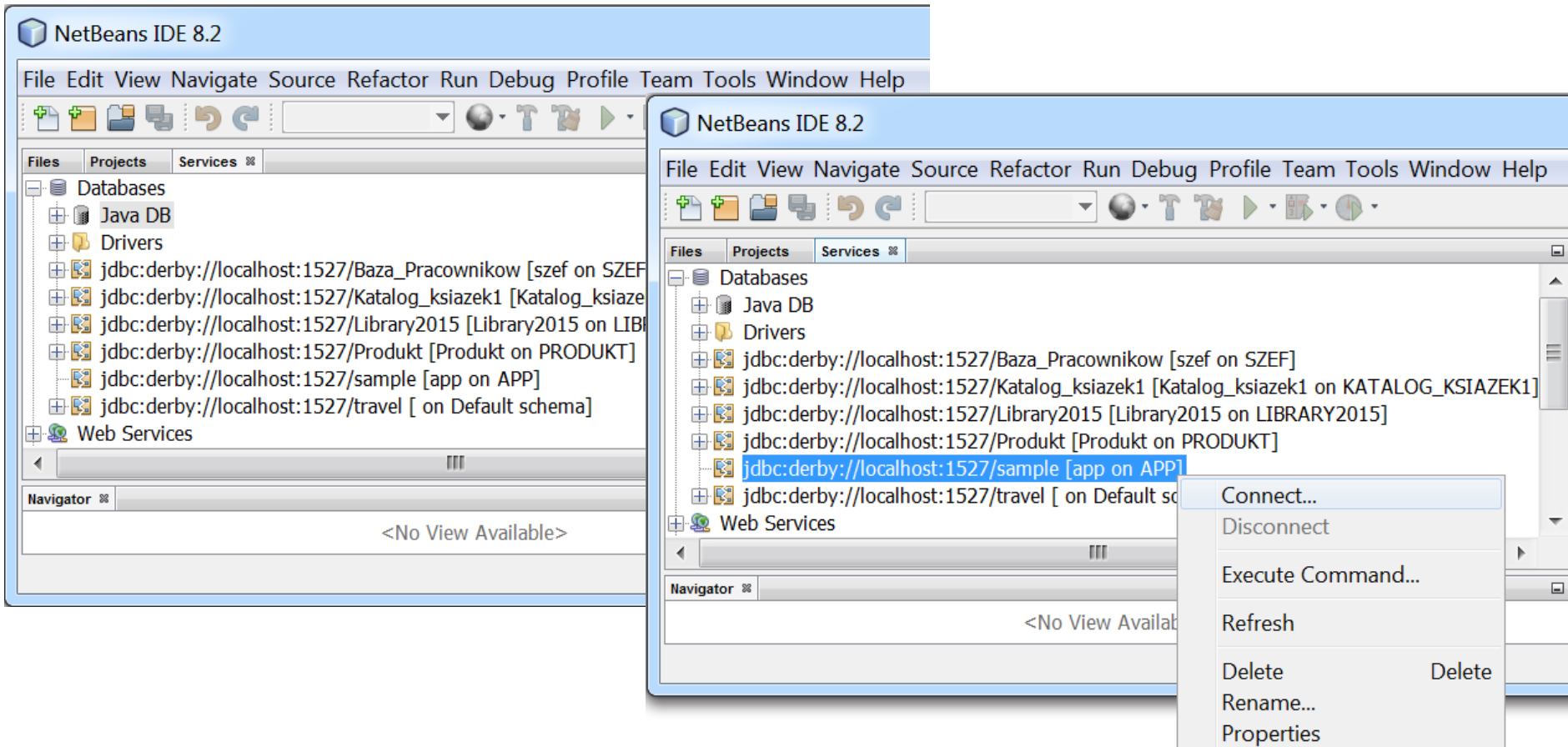
1a. Uruchom program Baza_1 w środowisku NetBeans 8.2 – załącznik do laboratorium. Należy uruchomić serwer bazy danych. W tym celu należy wykonać czynności pokazane poniżej w okienku Services. W p. 1b pokazano, jak zmienić program Java DB, gdy wystąpią problemy uruchomieniem serwera Java DB.



1b. Jeśli serwer **Java DB** nie startuje, należy zmienić jego program pobrany z zainstalowanego serwera glassfish 4.1.1 i ponownie uruchomić wg p. 1a.



2. W okienku Services należy otworzyć pozycję Databases. Następnie należy kliknąć prawym klawiszem myszy na rozerwany prostokąt bazy danych **jdbc:derby://localhost:1527/sample**. W menu podręcznym należy kliknąć na pozycję Connect. Po chwili prostokąt zostanie scalony, co symbolizuje połączenie z bazą danych.



Output - Java DB Database Process % Test Results

Mon May 08 15:44:46 CEST 2017 : Security manager installed using the Basic server security policy.

Mon May 08 15:44:46 CEST 2017 : Apache Derby Network Server - 10.11.1.2 - (1629631) started and ready to accept connections on port 1527

Widok bazy danych **Sample**

The screenshot displays the NetBeans IDE 8.2 interface. The top menu bar includes File, Edit, View, Navigat, Sourc, Refacto, Rur, Debuç, Profil, Team, Tool, Window, and Help. A search bar is present with the text "Search (Ctrl+I)".

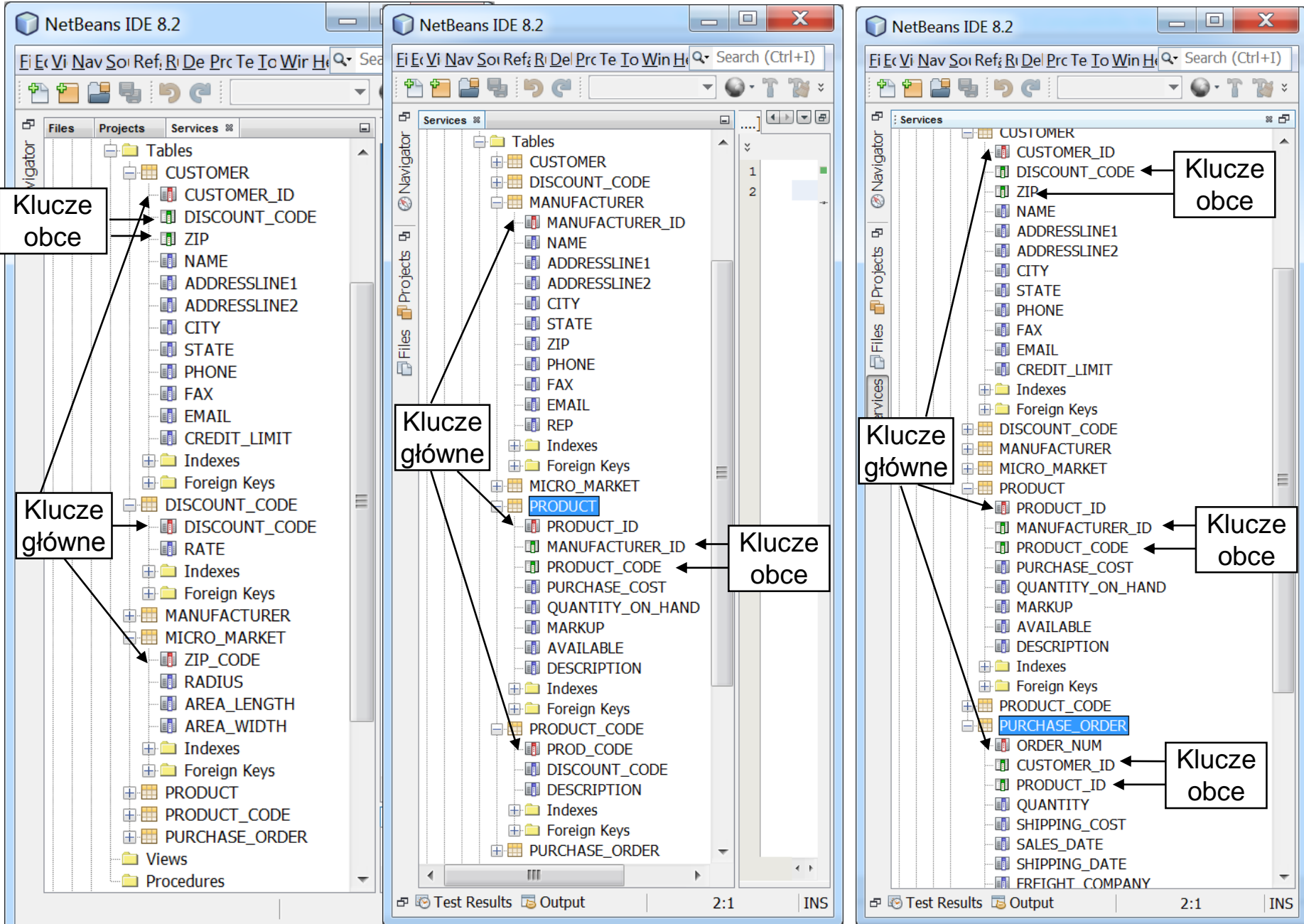
The Database Navigator window is open, showing a tree view of databases. The selected database is "jdbc:derby://localhost:1527/sample [app on APP]". Underneath, the "APP" schema is expanded, showing a folder named "Tables" containing the following tables:

- CUSTOMER
- DISCOUNT_CODE
- MANUFACTURER
- MICRO_MARKET
- PRODUCT
- PRODUCT_CODE
- PURCHASE_ORDER

Below the tables are folders for "Views" and "Procedures". At the bottom of the tree, another database connection is visible: "jdbc:derby://localhost:1527/travel [on Default schema]".

The Navigator window at the bottom of the IDE shows the text "<No View Available>".

Schemat bazy danych Sample



Wyświetlenie danych z tabeli **PURCHASE_ORDER**

The screenshot shows the NetBeans IDE 8.2 interface. On the left, the 'Projects' pane displays a database schema with several tables. The 'PURCHASE_ORDER' table is selected, and a context menu is open over it, listing actions such as 'View Data...', 'Execute Command...', 'Add Column...', 'Refresh', 'Delete', 'Grab Structure...', and 'Recreate Table...'. The main editor window shows the source code for 'baza_2.java', which includes package declarations, imports for SQL and Properties, and the start of a 'public class baza_2' with fields for 'data', 'sql', 'polaczenie', 'polecenie', and 'krotki'. The 'Output' window at the bottom shows the execution of 'GlassFish Server 4.1.1' and a successful connection message: 'SUCCESSFUL (total time: 26 minutes 50 seconds)'. The status bar at the bottom indicates '1:1' and 'INS'.

Zawartość tabeli PURCHASE_ORDER

The screenshot shows the NetBeans IDE 8.2 interface. The left sidebar contains a project tree with the following structure:

- CUSTOMER
 - CUSTOMER_ID
 - DISCOUNT_CODE
 - ZIP
 - NAME
 - ADDRESSLINE1
 - ADDRESSLINE2
 - CITY
 - STATE
 - PHONE
 - FAX
 - EMAIL
 - CREDIT_LIMIT
- Indexes
- Foreign Keys
- DISCOUNT_CODE
- MANUFACTURER
- MICRO_MARKET
- PRODUCT
- PRODUCT_CODE
- PURCHASE_ORDER
 - ORDER_NUM
 - CUSTOMER_ID
 - PRODUCT_ID
 - QUANTITY
 - SHIPPING_COST
 - SALES_DATE
 - SHIPPING_DATE
 - FREIGHT_COMPANY

The main window displays a SQL query in the editor:

```
SELECT * FROM APP.PURCHASE_ORDER FETCH FIRST 100 ROWS ONLY;
```

The results are shown in a table with the following columns: #, ORDER_NUM, CUSTOMER_ID, PRODUCT_ID, QUANTITY, SHIPPING_COST, SALES_DATE, SHIPPING_DATE, and FREIGHT_COMPANY. The table contains 15 rows of data.

#	ORDER_NUM	CUSTOMER_ID	PRODUCT_ID	QUANTITY	SHIPPING_COST	SALES_DATE	SHIPPING_DATE	FREIGHT_COMPANY
1	10398001	1	980001	10	449.00	2011-05-24	2011-05-24	Pony Express
2	10398002	2	980005	8	359.99	2011-05-24	2011-05-24	Pony Express
3	10398003	2	980025	25	275.00	2011-05-24	2011-05-24	Pony Express
4	10398004	3	980030	10	275.00	2011-05-24	2011-05-24	Pony Express
5	10398005	1	980032	100	459.00	2011-05-24	2011-05-24	Pony Express
6	10398006	36	986710	60	55.00	2011-05-24	2011-05-24	Slow Snail
7	10398007	36	985510	120	65.00	2011-05-24	2011-05-24	Slow Snail
8	10398008	106	988765	500	265.00	2011-05-24	2011-05-24	Slow Snail
9	10398009	149	986420	1000	700.00	2011-05-24	2011-05-24	Western Fast
10	10398010	863	986712	100	25.00	2011-05-24	2011-05-24	Slow Snail
11	20198001	777	971266	75	105.00	2011-05-24	2011-05-24	We deliver
12	20598100	753	980601	100	200.99	2011-05-24	2011-05-24	We deliver
13	20598101	722	980500	250	2500.00	2011-05-24	2011-05-24	Coastal Freight
14	30198001	409	980001	50	2000.99	2011-05-24	2011-05-24	Southern Delivery Service
15	30298004	410	980031	100	700.00	2011-05-24	2011-05-24	FR Express

The bottom status bar shows the execution output:

```
Executed successfully in 0.031 s.  
Fetching resultset took 0.015 s.  
Line 1, column 1
```


Metoda do obsługi połączenia z bazą danych – 1-y sposób obsługi wyjątków

The screenshot shows the NetBeans IDE 8.2 interface. The main editor displays the source code for the class `baza_1`. The code defines a class with several attributes and a method `polaczenie_z_baza()` that attempts to establish a database connection using JDBC. The method uses try-catch blocks to handle various exceptions like `ClassNotFoundException`, `IllegalAccessException`, and `InstantiationException`.

```
5 public class baza_1 {
6
7     String data, sql;
8     Connection polaczenie;
9     Statement polecenie;
10    ResultSet krotki;
11
12    public void polaczenie_z_baza() throws SQLException {
13        data = "jdbc:derby://localhost:1527/sample";
14        try {
15            Class.forName("org.apache.derby.jdbc.ClientDriver").newInstance();
16        } catch (ClassNotFoundException e) {
17            System.out.println("Nie mozna zaladowac sterownika");
18            throw new SQLException(e.toString());
19        } catch (IllegalAccessException e) {
20            System.out.println("Nie mozna zaladowac sterownika");
21            throw new SQLException(e.toString());
22        } catch (InstantiationException e) {
23            System.out.println("Nie mozna zaladowac sterownika");
24            throw new SQLException(e.toString());
25        }
26        try {
27            polaczenie = DriverManager.getConnection(data, "app", "app");
28        } catch (SQLException e) {
29            System.out.println("Nie mozna polaczyc sie z baza danych, poniewaz:" + e);
30            throw e;
31        }
32    }
}
```

The IDE interface includes a menu bar (File, Edit, View, Navigate, Source, Refactor, Run, Debug, Profile, Team, Tools, Window, Help), a toolbar, and a project explorer on the left showing the project structure. The status bar at the bottom shows the current execution state: `Java DB Database Process`, `GlassFish Server 4.1.1`, `SQL 1 execution`, and `Baza_1 (run)`.

Metoda do obsługi połączenia z bazą danych – 2-i sposób obsługi wyjątków

The screenshot displays the NetBeans IDE 8.2 interface. The main editor window shows the source code for the class `baza_1`. The code defines a public class with several attributes and a method `polaczenie_z_baza()` that attempts to establish a database connection. The method uses a try-catch block to handle exceptions, specifically `ClassNotFoundException`, `IllegalAccessException`, and `InstantiationException`. If these exceptions occur, it prints a message: "Nie mozna zaladowac sterownika". Another catch block handles `SQLException`, printing a message: "Nie mozna polaczyc sie z baza danych, poniewaz:" followed by the exception details.

```
5 public class baza_1 {
6
7     String data, sql;
8     Connection polaczenie;
9     Statement polecenie;
10    ResultSet krotki;
11
12    public void polaczenie_z_baza() throws SQLException {
13        data = "jdbc:derby://localhost:1527/sample";
14        try {
15            Class.forName("org.apache.derby.jdbc.ClientDriver").newInstance();
16        } catch (ClassNotFoundException | IllegalAccessException | InstantiationException e) {
17            System.out.println("Nie mozna zaladowac sterownika");
18            throw new SQLException(e.toString());
19        }
20        try {
21            polaczenie = DriverManager.getConnection(data, "app", "app");
22        } catch (SQLException e) {
23            System.out.println("Nie mozna polaczyc sie z baza danych, poniewaz:" + e);
24            throw e;
25        }
26    }
27 }
```

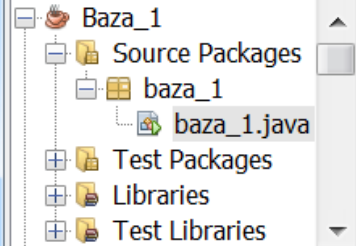
The left sidebar shows the project structure for "Baza_1", including source packages, test packages, and libraries. The "Members" view for the `baza_1` class is also visible, listing methods like `main(String[] arg)`, `polaczenie_z_baza()`, and `wyswietl_osoby()`, along with attributes like `data`, `krotki`, `polaczenie`, `polecenie`, and `sql`.

The bottom status bar shows the current location in the code: `baza_1.baza_1 > polaczenie_z_baza > try >`. The output window at the bottom displays the execution of the code, showing messages from the GlassFish Server and SQL execution logs.

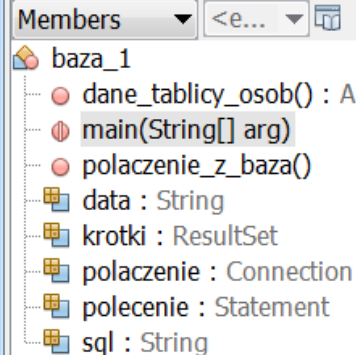


Proj... Files Services

.....] baza_1.java ResultSet.java build-impl.xml baza_2.java



main - Navigator



```

29 public ArrayList<String> dane_tablicy_osob() throws SQLException {
30     polecenie = polaczenie.createStatement();
31     sql = "SELECT * FROM CUSTOMER ORDER BY NAME";
32     krotki = polecenie.executeQuery(sql);
33     ResultSetMetaData metaDane = krotki.getMetaData();
34     int kolumny = metaDane.getColumnCount();
35     ArrayList<String> listakolumn=new ArrayList();
36     for (int i = 0; i < kolumny; i++) {
37         listakolumn.add("Nazwa kolumny " + i + " " + metaDane.getColumnName(i + 1));
38         listakolumn.add("\n");
39     }
40     listakolumn.add("\n");
41     for (int i = 1; i < kolumny - 1; i++) {
42         listakolumn.add(metaDane.getColumnName(i + 1) + "\t");
43     }
44     listakolumn.add("\n");
45     while (krotki.next()) {
46         listakolumn.add(krotki.getString("NAME") + "\t"
47             + krotki.getString("CITY") + "\t"
48             + krotki.getString("CREDIT_LIMIT"));
49         listakolumn.add("\n");
50     }
51     polecenie.close();
52     return listakolumn;
53 }

```

Wykonanie zapytania na tabeli PERSON

wykonanie działań pomocniczych

baza_1.baza_1 > main > try > listakolumn >

Output

>> Java DB Database Process SQL 1 execution SQL 2 execution Baza_1 (clean.jar)

Baza_1 - NetBeans IDE 8.2

File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help

Search (Ctrl+I)

Projects Files Services

Baza_1

- Source Packages
 - baza_1
 - baza_1.java
- Test Packages
- Libraries
- Test Libraries

main - Navigator

Members

- baza_1
 - dane_tablicy_osob(): ArrayList<String>
 - main(String[] arg)
 - polaczenie_z_baza()
 - data: String
 - krotki: ResultSet
 - polaczenie: Connection
 - polecenie: Statement
 - sql: String

Source

```

55 static public void main(String arg[]) {
56     baza_1 baza = new baza_1();
57     try {
58         baza.polaczenie_z_baza();
59         ArrayList<String> listakolumn=baza.dane_tablicy_osob();
60         System.out.println(listakolumn);
61     } catch (SQLException e) {
62         System.out.println(e.getMessage());
63         while (null != (e = e.getNextException())) {
64             System.out.println(e.getMessage());
65         }
66     }
67 }

```

Output

Java DB Database Process SQL 1 execution SQL 2 execution Baza_1 (clean,jar)

59:65 INS

Wynik działania programu

Test Libraries

- Baza_2
- Baza_3
- Baza_3_1
- Baza_4

dane_tablicy_osoby - Navi...

Members

- baza_1
 - dane_tablicy_osoby()
 - main(String[] arg)
 - polaczenie_z_baza()
 - data: String
 - krotki: ResultSet
 - polaczenie: Connecti
 - polecenie: Statement
 - sql: String

run:

```

[Nazwa kolumny 0 CUSTOMER_ID,
, Nazwa kolumny 1 DISCOUNT_CODE,
, Nazwa kolumny 2 ZIP,
, Nazwa kolumny 3 NAME,
, Nazwa kolumny 4 ADDRESSLINE1,
, Nazwa kolumny 5 ADDRESSLINE2,
, Nazwa kolumny 6 CITY,
, Nazwa kolumny 7 STATE,
, Nazwa kolumny 8 PHONE,
, Nazwa kolumny 9 FAX,
, Nazwa kolumny 10 EMAIL,
, Nazwa kolumny 11 CREDIT_LIMIT,
,
, DISCOUNT_CODE , ZIP , NAME , ADDRESSLINE1 , ADDRESSLINE2 , CITY , STATE , PHONE , FAX , EMAIL ,
, Big Car Parts Detroit 50000,
, Big Network Systems Redwood City 25000,
, Bob Hosting Corp. San Mateo 65000,
, Early CentralComp San Jose 26500,
, Jan Kowal M1 101,
, Jan Kowalski Wroclaw 101,
, Jan Nowak Miasto2 101,
, John Valley Computers Santa Clara 70000,
, Jumbo Eagle Corp Fort Lauderdale 100000,
, New Enterprises Miami 50000,
, Old Media Productions New York 10000,
, Small Bill Company Atlanta 90000,
, West Valley Inc. Dearborn 100000,
, Wren Computers Houston 25000,
, Yankee Computer Repair Ltd New York 25000,
, Zed Motor Co Dearborn 5000000,
]
BUILD SUCCESSFUL (total time: 0 seconds)

```

Baza_1 (run)

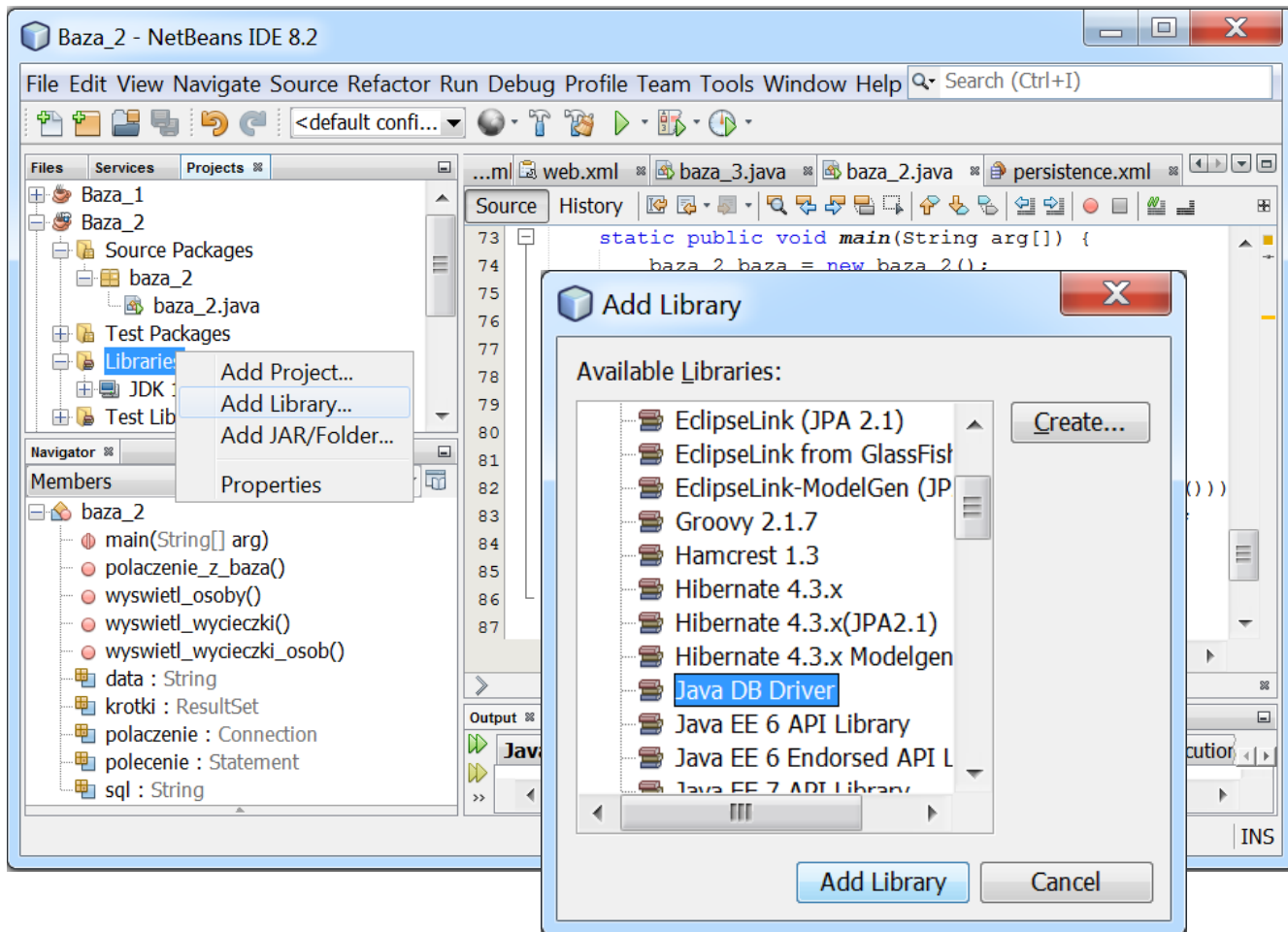
38:10 INS

Zadanie 2

cd zadania 1- należy dodać:

**wyświetlanie zawartość tabeli PURCHASE_ORDER (metoda dane_tablicy_zakupy()),
wyświetlanie zakupów każdej osoby (metoda dane_zakupy_osob())**

1. Wykonaj kopię programu Baza_1 jako Baza_2 (patrz instrukcja do lab3 - zad.2, pkt. 1)
2. Zmień nazwę pliku baza_1 na baza_2 (patrz instrukcja do lab3 – zad.2, pkt. 2)
3. Dodaj bibliotekę sterownika JDBC jak poniżej.
4. Wykonaj połączenie z bazą danych Sample wg Zadanie 1, p.2.



5. Zmodyfikuj zawartość metody `public ArrayList<String> dane_tablicy_osob()` wg slajdu poniżej.

The screenshot shows the NetBeans IDE 8.2 interface. The main editor window displays the `baza_2.java` file with the following code:

```
35 public ArrayList<String> dane_tablicy_osob() throws SQLException {
36     polecenie = polaczenie.createStatement();
37     sql = "SELECT * FROM CUSTOMER ORDER BY NAME";
38     krotki = polecenie.executeQuery(sql);
39     ArrayList<String> osoby = new ArrayList();
40     osoby.add("\nOsoby\n");
41     while (krotki.next()) {
42         osoby.add(krotki.getString("NAME") + "\t"
43                 + krotki.getString("CITY") + "\t"
44                 + krotki.getString("CREDIT_LIMIT"));
45         osoby.add("\n");
46     }
47     polecenie.close();
48     return osoby;
49 }
```

The left sidebar shows the project structure with `Baza_2` expanded to show `baza_2.java`. The `dane_tablicy_osob - Navigator` window shows the method `dane_tablicy_osob() : ArrayList<String>` selected. The bottom status bar indicates the current position is `35:5` and the file is `INS`.

6. Dodaj metodę `public ArrayList<String> dane_tablicy_zakupy()` - wyświetlanie zawartości tabeli `PURCHASE_ORDER`

The screenshot shows the NetBeans IDE 8.2 interface. The main editor displays the following Java code in `baza_2.java`:

```
72 public ArrayList<String> dane_tablicy_zakupy() throws SQLException {
73     polecenie = polaczenie.createStatement();
74     sql = "SELECT * FROM PURCHASE_ORDER ORDER BY SALES_DATE";
75     krotki = polecenie.executeQuery(sql);
76     ArrayList<String> zakupy = new ArrayList();
77     zakupy.add("\nZakupy\n");
78     while (krotki.next()) {
79         zakupy.add(krotki.getString("SALES_DATE") + "\t"
80                 + krotki.getString("SHIPPING_DATE") + "\t"
81                 + krotki.getString("SHIPPING_COST") + "\t"
82                 + krotki.getString("QUANTITY"));
83         zakupy.add("\n");
84     }
85     polecenie.close();
86     return zakupy;
87 }
```

The left sidebar shows the project structure with `Baza_2` and `baza_2.java`. The `dane_zakupy_osob - Navigator` window shows the following members:

- `dane_tablicy_osob() : ArrayList<String>`
- `dane_tablicy_zakupy() : ArrayList<String>`
- `dane_zakupy_osob() : ArrayList<String>`
- `main(String[] arg)`
- `polaczenie_z_baza()`
- `data : String`
- `krotki : ResultSet`
- `polaczenie : Connection`
- `polecenie : Statement`

The bottom status bar shows the current cursor position: `baza_2.baza_2 > dane_tablicy_zakupy > while (krotki.next()) >`. The `Output` window at the bottom shows the process: `Java DB Database Process > Baza_2 (run) >`.

7. Dodaj metodę `public ArrayList<String> dane_zakupy_osob()` -- wyświetlanie zakupów każdej osoby

The screenshot displays the NetBeans IDE 8.2 interface. The main editor window shows the source code for `baza_2.java`. The code implements the `dane_zakupy_osob()` method, which connects to a database, executes a SQL query to retrieve purchase data for all customers, and returns the results as an `ArrayList<String>`.

```
51 public ArrayList<String> dane_zakupy_osob() throws SQLException {
52     polecenie = polaczenie.createStatement();
53     sql = "SELECT * FROM CUSTOMER, PURCHASE_ORDER "
54         + " WHERE CUSTOMER.CUSTOMER_ID= PURCHASE_ORDER.CUSTOMER_ID"
55         + " ORDER BY NAME";
56     krotki = polecenie.executeQuery(sql);
57     ArrayList<String> zakupyosob = new ArrayList();
58     zakupyosob.add("\nZakupy poszczegolnych osob\n");
59     while (krotki.next()) {
60         zakupyosob.add(krotki.getString("name") + "\t"
61             + krotki.getString("CREDIT_LIMIT") + "\t"
62             + krotki.getString("SALES_DATE") + "\t"
63             + krotki.getString("SHIPPING_DATE") + "\t"
64             + krotki.getString("SHIPPING_COST") + "\t"
65             + krotki.getString("QUANTITY"));
66         zakupyosob.add("\n");
67     }
68     polecenie.close();
69     return zakupyosob;
70 }
```

The left sidebar shows the project structure with `Baza_2` selected. The `dane_zakupy_osob - Navigator` window lists the members of the class, including the `dane_zakupy_osob()` method. The bottom status bar shows the current file is `baza_2.baza_2 > dane_zakupy_osob >` and the output window is empty.

8. Uruchom program

The screenshot displays the NetBeans IDE 8.2 interface. The main editor window shows the source code for `baza_2.java`. The code defines a `main` method that performs the following actions:

```
88 static public void main(String arg[]) {
89     baza_2 baza = new baza_2();
90     ArrayList<String> lista;
91     try {
92         baza.polaczenie_z_baza();
93         lista = baza.dane_tablicy_osob();
94         System.out.println(lista);
95         lista = baza.dane_zakupy_osob();
96         System.out.println(lista);
97         lista = baza.dane_tablicy_zakupy();
98         System.out.println(lista);
99     } catch (SQLException e) {
100        System.out.println(e.getMessage());
101        while (null != (e = e.getNextException())) {
102            System.out.println(e.getMessage());
103        }
104    }
105 }
```

The left sidebar shows the project structure with `Baza_2` selected. Below it, the `dane_zakupy_osob - Navigator` window shows the members of the `baza_2` class, including `dane_tablicy_osob()`, `dane_tablicy_zakupy()`, `dane_zakupy_osob()`, `main(String[] arg)`, `polaczenie_z_baza()`, `data : String`, `krotki : ResultSet`, `polaczenie : Connection`, `polecenie : Statement`, and `sql : String`.

The bottom status bar shows the program is running, with the output window displaying `Java DB Database Process` and `Baza_2 (run)`. The time is 51:5 and the cursor is at INS.

Wynik działania programu (z lewej strony: zawartość tabeli **CUSTOMER**, z prawej strony zawartość tabeli **PURCHASE_ORDER**)

NetBeans IDE 8.2 interface showing the project structure and the output of a Java program. The project 'Baza_2' is selected, and the 'Output' window displays the contents of the 'CUSTOMER' table.

Members:

- dane_tablicy_osoby() : ArrayList<String>
- dane_tablicy_zakupy() : ArrayList<String>
- dane_zakupy_osoby() : ArrayList<String>
- main(String[] arg)
- polaczenie_z_baza()
- data : String

Output:

```
run:
[
Osoby
, Big Car Parts Detroit 50000,
, Big Network Systems Redwood City 25000,
, Bob Hosting Corp. San Mateo 65000,
, Early CentralComp San Jose 26500,
, John Valley Computers Santa Clara 70000,
, Jumbo Eagle Corp Fort Lauderdale 100000,
, New Enterprises Miami 50000,
, Old Media Productions New York 10000,
, Small Bill Company Alanta 90000,
, West Valley Inc. Dearborn 100000,
, Wren Computers Houston 25000,
, Yankee Computer Repair Ltd New York 25000,
, Zed Motor Co Dearborn 5000000,
]
```

Members:

- dane_tablicy_osoby() : ArrayList<String>
- dane_tablicy_zakupy() : ArrayList<String>
- dane_zakupy_osoby() : ArrayList<String>
- main(String[] arg)
- polaczenie_z_baza()
- data : String
- krotki : ResultSet

Output:

```
Zakupy
, 2011-05-24 2011-05-24 700.00 100,
, 2011-05-24 2011-05-24 2000.99 50,
, 2011-05-24 2011-05-24 2500.00 250,
, 2011-05-24 2011-05-24 200.99 100,
, 2011-05-24 2011-05-24 105.00 75,
, 2011-05-24 2011-05-24 25.00 100,
, 2011-05-24 2011-05-24 700.00 1000,
, 2011-05-24 2011-05-24 265.00 500,
, 2011-05-24 2011-05-24 65.00 120,
, 2011-05-24 2011-05-24 55.00 60,
, 2011-05-24 2011-05-24 459.00 100,
, 2011-05-24 2011-05-24 275.00 10,
, 2011-05-24 2011-05-24 275.00 25,
, 2011-05-24 2011-05-24 359.99 8,
, 2011-05-24 2011-05-24 449.00 10,
]
```

BUILD SUCCESSFUL (total time: 0 seconds)

Wynik działania programu (zawartość złączonych tabel CUSTOMER i PURCHASE_ORDER)

The screenshot shows the NetBeans IDE 8.2 interface. The main window is titled "Baza_2 - NetBeans IDE 8.2". The menu bar includes File, Edit, View, Navigate, Source, Refactor, Run, Debug, Profile, Team, Tools, Window, and Help. The toolbar contains various icons for file operations and execution. The left sidebar shows the "Projects" view with "Baza_2" selected, and the "Members" view for "dana_tablicy_zakupy" showing several methods and variables.

The source editor displays the following code snippet:

```
baza_2.baza_2 > dane_tablicy_zakupy > while (krotki.next()) >
```

The output console shows the following text:

```
Java DB Database Process Baza_2 (run)
[
Zakupy poszczegolnych osob
, Big Car Parts 50000 2011-05-24 2011-05-24 2500.00 250,
, Big Network Systems 25000 2011-05-24 2011-05-24 25.00 100,
, Bob Hosting Corp. 65000 2011-05-24 2011-05-24 65.00 120,
, Bob Hosting Corp. 65000 2011-05-24 2011-05-24 55.00 60,
, Early CentralComp 26500 2011-05-24 2011-05-24 265.00 500,
, John Valley Computers 70000 2011-05-24 2011-05-24 700.00 1000,
, Jumbo Eagle Corp 100000 2011-05-24 2011-05-24 459.00 100,
, Jumbo Eagle Corp 100000 2011-05-24 2011-05-24 449.00 10,
, New Enterprises 50000 2011-05-24 2011-05-24 275.00 25,
, New Enterprises 50000 2011-05-24 2011-05-24 359.99 8,
, Old Media Productions 10000 2011-05-24 2011-05-24 2000.99 50,
, Small Bill Company 90000 2011-05-24 2011-05-24 275.00 10,
, West Valley Inc. 100000 2011-05-24 2011-05-24 105.00 75,
, Yankee Computer Repair Ltd 25000 2011-05-24 2011-05-24 700.00 100,
, Zed Motor Co 5000000 2011-05-24 2011-05-24 200.99 100,
]
```

The status bar at the bottom right shows the time "83:19" and the keyboard shortcut "INS".

Zadanie 3 - cd zadania 2

dodawanie nowych osób do tabeli CUSTOMER oraz dodawanie nowych zakupów wybranej osoby do tabeli PURCHASE_ORDER

1. Wykonaj kopię programu Baza_2 jako Baza_3 (patrz instrukcja do lab3 - zad.2, pkt. 1)
2. Zmień nazwę pliku baza_2 na baza_3 (patrz instrukcja do lab3 – zad.2, pkt. 2)
3. Dodaj sterownik JavaDBDrive wg pkt3 z zad2.
4. Wykonaj połączenie z bazą danych Sample wg Zadanie 1, p.2.
5. Wykonaj pakiet GUI
 - 5.1. dodaj pomocniczą klasę WeWy do pakietu GUI – do wprowadzania danych z klawiatury. Podczas edycji kodu zaimportuj brakujące biblioteki za pomocą Fix Imports (kliknij prawym klawiszem w oknie Edytora Javy i wybór opcji Fix Imports)

```
public class WeWy {  
    static public String weString(String menu) {  
        InputStreamReader wejście = new InputStreamReader(System.in);  
        BufferedReader bufor = new BufferedReader(wejście);  
        try {  
            System.out.print(menu);  
            return bufor.readLine();  
        } catch (IOException e) {  
            System.err.println("Bład IO String");  
            return "";  
        }  
    }  
  
    static public void wyString(ArrayList<String> lista) {  
        lista.forEach((wiersz) -> {  
            System.out.print(wiersz);  
        });  
    }  
}
```


5.2. Dodaj klasę **GUICustomer** w pakiecie GUI w celu wprowadzania danych do tabeli **CUSTOMER**

```
public class GUICustomer {
```

```
public String name, addressline1, adressline2, city, state, phone, fax, email;  
static public int credit_limit = 100;
```

```
public String[] wstaw_dane() {
```

```
    name          = WeWy.weString("Podaj nazwisko: ");
```

```
    addressline1  = WeWy.weString("Podaj adres1: ");
```

```
    adressline2   = WeWy.weString("Podaj adres2: ");
```

```
    city          = WeWy.weString("Podaj miasto: ");
```

```
    state         = WeWy.weString("Podaj stan: ");
```

```
    phone         = WeWy.weString("Podaj numer telefonu: ");
```

```
    fax          = WeWy.weString("Podaj numer faksu: ");
```

```
    email        = WeWy.weString("Podaj email: ");
```

```
    credit_limit++;
```

```
String[] dane = {name, addressline1, adressline2, city, state, phone, fax, email};
```

```
return dane;
```

```
}
```

```
}
```

5.2. cd. Dodana klasa **WeWy** do wprowadzania danych typu String (**weString**) i wyświetlania listy łańcuchów (**wyString**)

The screenshot displays the NetBeans IDE 8.2 interface. The main editor window shows the `WeWy.java` file with the following code:


```
1 package GUI;
2 import java.io.BufferedReader;
3 import java.io.IOException;
4 import java.io.InputStreamReader;
5 import java.util.ArrayList;
6
7 public class WeWy {
8
9     static public String weString(String menu) {
10         InputStreamReader wejście = new InputStreamReader(System.in);
11         BufferedReader bufor = new BufferedReader(wejście);
12         try {
13             System.out.print(menu);
14             return bufor.readLine();
15         } catch (IOException e) {
16             System.err.println("Bład IO String");
17             return "";
18         }
19     }
20
21     static public void wyString(ArrayList<String> lista) {
22         lista.forEach((wiersz) -> {
23             System.out.print(wiersz);
24         });
25     }
26 }
```

The left sidebar shows the project structure for `Baza_3`, including `Source Packages` and `GUI`. The `Navigator` window shows the `WeWy` class with its methods `weString(String menu) : String` and `wyString(ArrayList<String> lista)`. The bottom status bar indicates the application is running: `Java DB Database Process` and `Baza_3 (run)`.

5.3. Dodaj metodę **public String klucz_glowny(String sql1, String sql2)** do klasy **baza_3** w celu wyznaczenia wartości kolejnego klucza głównego. Metoda jest uniwersalna, ponieważ łańcuch `sql2 + "." + sql1` tworzy nazwę pola klucza głównego w dowolnej tabeli (`sql1` jest nazwą klucza głównego, a `sql2` jest nazwą tabeli). Metoda zwraca wartość kolejnego klucza głównego, jakim można nadać nowej krotce lub null.

```
sql = " SELECT MAX(" + sql2 + "." + sql1 + ") + 1 AS MAXID FROM " + sql2;
```

```
public String klucz_glowny(String sql1, String sql2) {  
    try {  
        sql = " SELECT MAX(" + sql2 + "." + sql1 + ") + 1 AS MAXID FROM " + sql2;  
        krotki = polecenie.executeQuery(sql);  
        krotki.next();  
        String pom = krotki.getString("MAXID");  
        return pom;  
    } catch (SQLException e) {  
        System.out.println("Bład w wyznaczaniu klucza" + e);  
    }  
    return null;  
}
```



5.4. Dodaj metodę **void blad()** w klasie **baza3** do obsługi błędów wstawiania danych do tabeli **CUSTOMER** i **PURCHASE_ORDER**

```
void blad(BatchUpdateException e) {  
    System.out.println("Wycofanie transakcji ");  
    SQLException e1 = e.getNextException();  
    while (e1 != null) {  
        System.out.println(e1.getMessage());  
        e1 = e1.getNextException();  
    }  
}
```

5.5. Dodaj metodę **public void wstaw_osobe(String[] dane)** do klasy **baza_3** w celu dodania nowej krotki do tabeli **CUSTOMER**

```
public void wstaw_osobe(String[] dane) throws SQLException {
    String id_osoby;
    polaczenie.setAutoCommit(false);
    try {
        polecenie = polaczenie.createStatement();
        if ((id_osoby = klucz_glowny("CUSTOMER_ID", "CUSTOMER")) == null)
            return;
        sql = "INSERT INTO CUSTOMER (customer_id, discount_code1, zip, name, "
            + "addressline1,addressline2, city, state, phone,fax, email, credit_limit)"
            + " VALUES (" + id_osoby + ",'" + "N" + "','" + "48128" + "','" + dane[0] + "','"
            + dane[1] + "','" + dane[2] + "','" + dane[3] + "','" + dane[4] + "','"
            + dane[5] + "','" + dane[6] + "','" + dane[7] + "','" + GUICustomer.credit_limit + ")";
        polecenie.addBatch(sql);
        polecenie.executeBatch();
        polaczenie.commit();
    } catch (BatchUpdateException e) {
        blad(e);
        polaczenie.rollback();
    }
    sql = "INSERT INTO CUSTOMER (CUSTOMER_ID, discount_code, zip, name, addressline1,"
        + "addressline2, city, state, phone,fax, email, credit_limit)"
        + " VALUES (" + id_osoby + "','" + "N" + "','" + "48128" + "','" + dane[0] + "','"
        + dane[1] + "','" + dane[2] + "','" + dane[3] + "','" + dane[4] + "','"
        + dane[5] + "','" + dane[6] + "','" + dane[7] + "','"
        + GUICustomer.credit_limit + ")";
}
```



5.5. cd. Działanie metody `public void wstaw_osobe(String[] dane)`, gdy wystąpił błąd nazwy jednej z kolumn tabeli **CUSTOMER**

The screenshot shows the NetBeans IDE interface for a project named 'Baza_3'. The left sidebar displays the project structure with packages 'GUI' and 'baza_3', and files 'GUICustomer.java' and 'WeWy.java'. The main editor area shows the 'Output' window with a list of SQL execution logs. The error message is highlighted in the output window, and an arrow points from the text in the top box to it.

```
Podaj nazwisko: Jan Kowalski
Podaj adres1: Adres1
Podaj adres2: Adres2
Podaj miasto: Miasto1
Podaj stan: DL
Podaj numer telefonu: 123-123-123
Podaj numer faksu: 234-234-2340
Podaj email: wl@wl.pl
id: 864
Wycofanien transakcji
Error for batch element #0: 'DISCOUNT_CODE1' is not a column in table or VTI 'APP.CUSTOMER'.
```


5.5. cd. Działanie metody **public void wstaw_osobe(String[] dane)**, gdy usunięto błąd nazwy jednej z kolumn tabeli **CUSTOMER**

The screenshot shows the NetBeans IDE 8.2 interface. The 'Projects' pane on the left shows the project structure for 'Baza_3', including 'GUI' and 'baza_3' packages. The 'Members' pane shows the 'main' method of the 'baza_3' class, with 'wstaw_osobe(String[] dane)' highlighted. The 'Output' pane shows the execution of the program, displaying a list of people and their addresses. The 'wstaw_osobe' method is highlighted in the 'Members' pane, and an arrow points from the text in the top section to this method.

```
Podaj nazwisko: Jan Kowalski
Podaj adres1: Adres1
Podaj adres2: Adres2
Podaj miasto: Wroclaw
Podaj stan: DL
Podaj numer telefonu: 123-123-123
Podaj numer faksu: 123-123-1230
Podaj email: wk@w1.pl

Osoby
Big Car Parts      Detroit 50000
Big Network Systems  Redwood City 25000
Bob Hosting Corp.  San Mateo 65000
Early CentralComp  San Jose 26500
Jan Kowalski      Wroclaw 101
Jan Kowalski      Miastol 101
John Valley Computers  Santa Clara 70000
Jumbo Eagle Corp    Fort Lauderdale 100000
New Enterprises Miami 50000
Old Media Productions  New York 10000
Small Bill Company  Alanta 90000
West Valley Inc.    Dearborn 100000
Wren Computers     Houston 25000
Yankee Computer Repair Ltd  New York 25000
Zed Motor Co       Dearborn 5000000
```

6. Dodawanie nowej krotki do tablicy **PURCHASE_ORDER**

6.1. Dodaj klasę **GUIPurchase_order** do pakietu GUI do wprowadzania danych do tabeli **PURCHASE_ORDER**. Metoda **public String wstaw_date(String komunikat)** otrzymuje liczbę dni metodą **weString()** i dodaje je do daty bieżącej metodą **setTime**. Następnie wykorzystuje metodą **SimpleDateFormat** do nadania dacie formatu wymaganego w tabeli **PURCHASE_ORDER** w bazie danych.

```
package GUI;
```

```
import java.util.Date;
```

```
import java.text.SimpleDateFormat;
```

```
public class GUIPurchase_order {
```

```
    private String quantity, shipping_cost, freight_company, name, kod_produktu,  
        sales_date, shipping_date;
```

```
    public String wstaw_date(String komunikat) {
```

```
        Date pom1 = new Date();
```

```
        String pom2 = WeWy.weString(komunikat);           // liczbe dni od dnia bieżącego
```

```
        pom1.setTime(pom1.getTime() + Long.parseLong(pom2) * 24 * 60 * 60 * 1000);
```

```
        SimpleDateFormat format = new SimpleDateFormat("yyyy-MM-dd");
```

```
        String data = format.format(pom1.getTime());
```

```
        return data;
```

```
    }
```

```
}
```

6. Dodawanie nowej krotki do tablicy **PURCHASE_ORDER** cd

6.2. Dodaj metodę **String[] wstaw_dane_zlecenia()** do wprowadzania danych do tabeli **PURCHASE_ORDER**. Metoda umożliwia pobranie wszystkich danych do utworzenia nowej krotki w tabeli **PURCHASE_ORDER**

```
public String[] wstaw_dane_zlecenia() {
    sales_date          = wstaw_date("Podaj date sprzedazy (za ile dni): ");
    shipping_date       = wstaw_date("Podaj date wysylki (za ile dni): ");
    quantity            = WeWy.weString("Podaj koszt zakupu: ");
    shipping_cost       = WeWy.weString("Podaj koszt wysylki: ");
    freight_company    = WeWy.weString("Podaj firme przewozowa: ");
    name                = WeWy.weString("Podaj nazwisko osoby: ");
    kod_produktu        = WeWy.weString("Podaj kod produktu: ");
    String [] dane = {name, kod_produktu, quantity, shipping_cost,
                     sales_date, shipping_date, freight_company};
    return dane;
}
}
```

6.3. Należy dodać metodę **public String wyszukaj_osobe(String name)** w celu wyszukania **klucza głównego krotki z tabeli CUSTOMER** oraz metodę **public String wyszukaj_produkct(String kod)** w celu wyszukania **klucza głównego krotki z tabeli PRODUCT** - do klasy baza_3

```
sql = "SELECT * FROM CUSTOMER WHERE CUSTOMER.NAME = '" + name + "'";
```

```
public String wyszukaj_osobe(String name) {
```

```
try {
```

```
    sql = "SELECT * FROM CUSTOMER WHERE CUSTOMER.NAME = '" + name + "'";
```

```
    krotki = polecenie.executeQuery(sql);
```

```
    if (krotki.next()) {
```

```
        return krotki.getString("CUSTOMER_ID");
```

```
    }
```

```
} catch (SQLException e) {
```

```
    System.out.println("Bład w wyszukaniu osoby" + e);
```

```
}
```

```
return null;
```

```
}
```

```
sql = "SELECT * FROM Product WHERE PRODUCT_CODE = '" + kod + "'";
```

```
public String wyszukaj_produkct(String kod) {
```

```
try {
```

```
    sql = "SELECT * FROM Product WHERE PRODUCT_CODE = '" + kod + "'";
```

```
    krotki = polecenie.executeQuery(sql);
```

```
    if (krotki.next()) {
```

```
        return krotki.getString("Product_ID");
```

```
    }
```

```
} catch (SQLException e) {
```

```
    System.out.println("Bład w wyszukaniu produktu" + e);
```


```
}
```

```
return null;
```

```
}
```

6.4. Należy dodać metodę **public void wstaw_zleczenie(String[] dane)** do klasy **baza_3** w celu wstawienia nowej krotki do tabeli **PURCHASE_ORDER**

```
public void wstaw_zleczenie(String[] dane) throws SQLException {
    String id_osoby, id_produkt, id_order;
    polaczenie.setAutoCommit(false);
    try {
        polecenie = polaczenie.createStatement();
        if ((id_osoby = wyszukaj_osobe(dane[0])) == null) {
            System.out.println("Brak osoby");
            return; }
        if ((id_produkt = wyszukaj_produkt(dane[1])) == null) {
            System.out.println("Brak produktu");
            return; }
        if ((id_order = klucz_glowny("Order_num", "Purchase_order")) == null) {
            System.out.println("null");
            return; }
        sql = "INSERT INTO PURCHASE_ORDER (Order_num,customer_id,product_id,Quantity,shipping_cost,"
            + "sales_date,shipping_date,freight_company) VALUES ("
            + id_order + "," + id_osoby + "," + id_produkt + "," + dane[2] + "," + dane[3]
            + "," + dane[4] + "," + dane[5] + "," + dane[6] + ")";
        polecenie.executeUpdate(sql);
        polecenie.executeBatch();
        polaczenie.commit();
    } catch (BatchUpdateException e) {
        blad(e);
        polaczenie.rollback();
    }
}
```



```
    sql = "INSERT INTO PURCHASE_ORDER (Order_num,customer_id,product_id,Quantity,shipping_cost,"
        + "sales_date,shipping_date,freight_company) VALUES ("
        + id_order + "," + id_osoby + "," + id_produkt + "," + dane[2] + "," + dane[3]
        + "," + dane[4] + "," + dane[5] + "," + dane[6] + ")";
```

6.5. Metoda main

The screenshot displays the NetBeans IDE 8.2 interface. The main window title is "Baza_3 - NetBeans IDE 8.2". The menu bar includes File, Edit, View, Navigate, Source, Refactor, Run, Debug, Profile, Team, Tools, Window, and Help. A search bar is present with the text "Search (Ctrl+I)".

The left sidebar shows the "Projects" view with a tree structure:

- Baza_1
- Baza_2
- Baza_3
 - Source Packages
 - GUI
 - GUICustomer.java
 - GUIPurchase_order.java
 - WeWy.java
 - baza_3
 - baza_3.java
 - Test Packages

The "wstaw_zlecenie - Navigator" window shows the members of the `baza_3` class:

- `blad(BatchUpdateException e)`
- `dane_tablicy_osob() : ArrayList`
- `dane_tablicy_zakupy() : ArrayList`
- `dane_zakupy_osob() : ArrayList`
- `klucz_glowny(String sql1, String ...)`
- `main(String[] arg)`
- `polaczenie_z_baza()`
- `wstaw_osobe(String[] dane)`
- `wstaw_zlecenie(String[] dane)`
- `wyszukaj_osobe(String name) :`
- `wyszukaj_produkty(String kod) :`
- `data : String`

The main editor shows the source code for `baza_3.java`, specifically the `main` method:

```
187 static public void main(String arg[]) {
188     baza_3 baza = new baza_3();
189     GUICustomer guicustomer = new GUICustomer();
190     GUIPurchase_order guipurchase_order = new GUIPurchase_order();
191     String [] tabela;
192     ArrayList<String> lista;
193     try {
194         baza.polaczenie_z_baza();
195         lista=baza.dane_tablicy_osob();           WeWy.wyString(lista);
196         lista=baza.dane_zakupy_osob();         WeWy.wyString(lista);
197         lista=baza.dane_tablicy_zakupy();     WeWy.wyString(lista);
198         tabela=guicustomer.wstaw_dane();     baza.wstaw_osobe(tabela);
199         lista=baza.dane_tablicy_osob();       WeWy.wyString(lista);
200         tabela=guipurchase_order.wstaw_dane_zlecenia(); baza.wstaw_zlecenie(tabela);
201         lista=baza.dane_zakupy_osob();       WeWy.wyString(lista);
202     } catch (SQLException e) {
203         System.out.println(e.getMessage());
204         while (null != (e = e.getNextException()))
205             System.out.println(e.getMessage());
206     }
207 }
208 }
```

The "Output" window at the bottom shows the execution of the program:

```
>> Java DB Database Process
Baza_3 (run)
```


6.6. Wynik działania nowej metody `public void wstaw_zlecenie(String[] dane)`

The screenshot shows the NetBeans IDE 8.2 interface. The Project Explorer on the left shows the project structure for 'Baza_3', including source packages 'GUI' and 'baza_3'. The Source Editor in the center shows the code for 'baza_3.java', with a line of SQL code: `sql = "INSERT INTO PURCHASE_ORDER (Order_nur`. The Navigator on the left shows the 'main' method selected. The Output window on the right displays the execution results, including user input and a table of purchase data.

Podaj date sprzedazy (za ile dni): 4
Podaj date wysylki (za ile dni): 5
Podaj koszt zakupu: 234
Podaj koszt wysylki: 23
Podaj firme przewozowa: wvw
Podaj nazwisko osoby: Jan Kowalski
Podaj kod produktu: SW

Zakupy poszczegolnych osob

Big Car Parts	50000	2011-05-24	2011-05-24	2500.00	250
Big Network Systems	25000	2011-05-24	2011-05-24	25.00	100
Bob Hosting Corp.	65000	2011-05-24	2011-05-24	65.00	120
Bob Hosting Corp.	65000	2011-05-24	2011-05-24	55.00	60
Early Centralcamp	26500	2011-05-24	2011-05-24	265.00	500
Jan Kowalski	101	2017-05-14	2017-05-15	23.00	234
Jan Kowalski	101	2017-05-14	2017-05-15	12.00	12
John Valley Computers	70000	2011-05-24	2011-05-24	700.00	1000
Jumbo Eagle Corp	100000	2011-05-24	2011-05-24	459.00	100
Jumbo Eagle Corp	100000	2011-05-24	2011-05-24	449.00	10
New Enterprises 50000	50000	2011-05-24	2011-05-24	275.00	25
New Enterprises 50000	2011-05-24	2011-05-24	359.99	8	
Old Media Productions	10000	2011-05-24	2011-05-24	2000.99	50
Small Bill Company	90000	2011-05-24	2011-05-24	275.00	10
West Valley Inc.	100000	2011-05-24	2011-05-24	105.00	75
Yankee Computer Repair Ltd	25000	2011-05-24	2011-05-24	700.00	100
Zed Motor Co	5000000	2011-05-24	2011-05-24	200.99	100

BUILD SUCCESSFUL (total time: 9 minutes 42 seconds)

Zadanie 4

cd zadania 3

dodawanie przeszukiwania tabel **CUSTOMER** i **PURCHASE_ORDER**

dodawanie nowych krotek do wybranej tabeli z: **PRODUCT**, **MANUFACTURER**, lub **MICRO_MARKET**

dodanie dowolnej operacji na tabelach bazy danych **Sample**

1. Wykonaj kopię programu Baza_3 jako Baza_4 (patrz instrukcja do lab3 - zad.2, pkt. 1)
2. Zmień nazwę pliku baza_3 na baza_4 (patrz instrukcja do lab3 – zad.2, pkt. 2)
3. Dodaj sterownik JavaDBDrive wg pkt3 z zad2.
4. Wykonaj połączenie z bazą danych **Sample** wg Zadanie 1, p.2.
5. Dodaj metodę **public void wyszukaj_wg_miasta() throws SQLException**, która wykonuje zapytanie wyszukiwania osób wg miasta w tablicy **CUSTOMER** posiadających zakup w tablicy **PURCHASE_ORDER**
6. Dodaj metodę, która wstawia nową krotkę do tabeli **PRODUCT** lub **MANUFACTURER** lub **MICRO_MARKET**
7. Wykonaj dowolną operację w bazie danych **Sample**.

Ilustracja do p.5 - Dodaj metodę **public void wyszukaj_wg_miasta() throws SQLException**, która wykonuje zapytanie wyszukiwania osób wg miasta w tablicy **CUSTOMER** posiadających zakup w tablicy **PURCHASE_ORDER**

The screenshot shows the NetBeans IDE 8.2 interface. The main editor window displays the source code for `baza_4.java`. The code includes a `main` method that performs database operations and displays results. The left sidebar shows the project structure, including the `baza_4` package and its sub-packages `GUI` and `baza_4`. The bottom output window shows the execution results, including a prompt `Podaj miasto: New York` and a table of purchase data for a specific person.

```
207     static public void main(String arg[]) {
208         baza_4 baza = new baza_4();
209         // GUICustomer guicustomer = new GUICustomer();
210         // GUIPurchase_order guipurchase_order = new GUIPurchase_order();
211         // String [] tabela;
212         ArrayList<String> lista;
213         try {
214             baza.polaczenie_z_baza();
215             lista=baza.dane_tablicy_osob();           WeWy.wyString(lista);
216             lista=baza.dane_zakupy_osob();           WeWy.wyString(lista);
217             lista=baza.dane_tablicy_zakupy();        WeWy.wyString(lista);
218             // tabela=guicustomer.wstaw_dane();     baza.wstaw_osobe(tabela);
219             // lista=baza.dane_tablicy_osob();        WeWy.wyString(lista);
220             // tabela=guipurchase_order.wstaw_dane_zleczenia(); baza.wstaw_zleczenie(tabela);
221             // lista=baza.dane_zakupy_osob();        WeWy.wyString(lista);
222             lista=baza.wyszukaj_zakupy_wg_miasta(WeWy.weString("Podaj miasto: "));
223             WeWy.wyString(lista);
224         } catch (SQLException e) {
225             System.out.println(e.getMessage());
226             while (null != (e = e.getNextException())) {
227                 System.out.println(e.getMessage());
228             }
229         }
230     }
231 }
```

Output window shows the execution results:

```
Java DB Database Process  Baza_4 (run)
Podaj miasto: New York

Zakupy danej osoby
2011-05-24      2011-05-24      700.00  100
2011-05-24      2011-05-24      2000.99  50
BUILD SUCCESSFUL (total time: 6 seconds)
```

Zadanie 5

- 1) Wykonaj interfejs graficzny użytkownika, **podobnie jak dalej pokazano w p. 2-6**, do wybranych operacji na bazie danych wykonanych w zadaniach 1-4 np `public void wstaw_osobe(String[] dane)` i dodanej `items_` wyświetlającej dane tablicy `CUSTOMER` (podobnie jak metoda `public ArrayList<String> dane_tablicy_osob()`).
 - 2) Przykłady programów z GUI do wprowadzania danych do tabeli `PURCHASE_ORDER` oraz wyświetlania złączonych tabel `CUSTOMER` i `PURCHASE_ORDER` (po wykonaniu połączenia z bazą danych Sample wg Zadanie 1, p.2) zamieszczono w projekcie `Sklep_Gui_Baza3` i powiązanego z projektem `Baza_LB`, dołączonych do laboratorium (wg programu z lab6, Języki i metody programowania-Java) - <http://zofia.kruczkiewicz.staff.iiar.pwr.wroc.pl/wyklady/java/Zad5.rar>
4. Aplikacje współpracujące z bazami danych - sterowniki JDBC (2017).
1. [Instrukcja 4.](#)
 2. [Program pomocniczy Baza 1 do p. 1 instrukcji 4.](#)
 3. [Program pomocniczy do p. 5 instrukcji 4.](#)

The image displays two screenshots of a Java GUI application named "MenuDemo".

The left screenshot shows a form for adding a customer order. The form has the following fields and values:

- Name: Jan Kowalski
- Code product: BK
- Sales date: 15-05-2017
- shipping_date: 16-05-2017
- shipping_cost: 456
- Quantity: 34
- Freight company: www

There is a button labeled "Dodaj zakup klienta" at the bottom of the form.

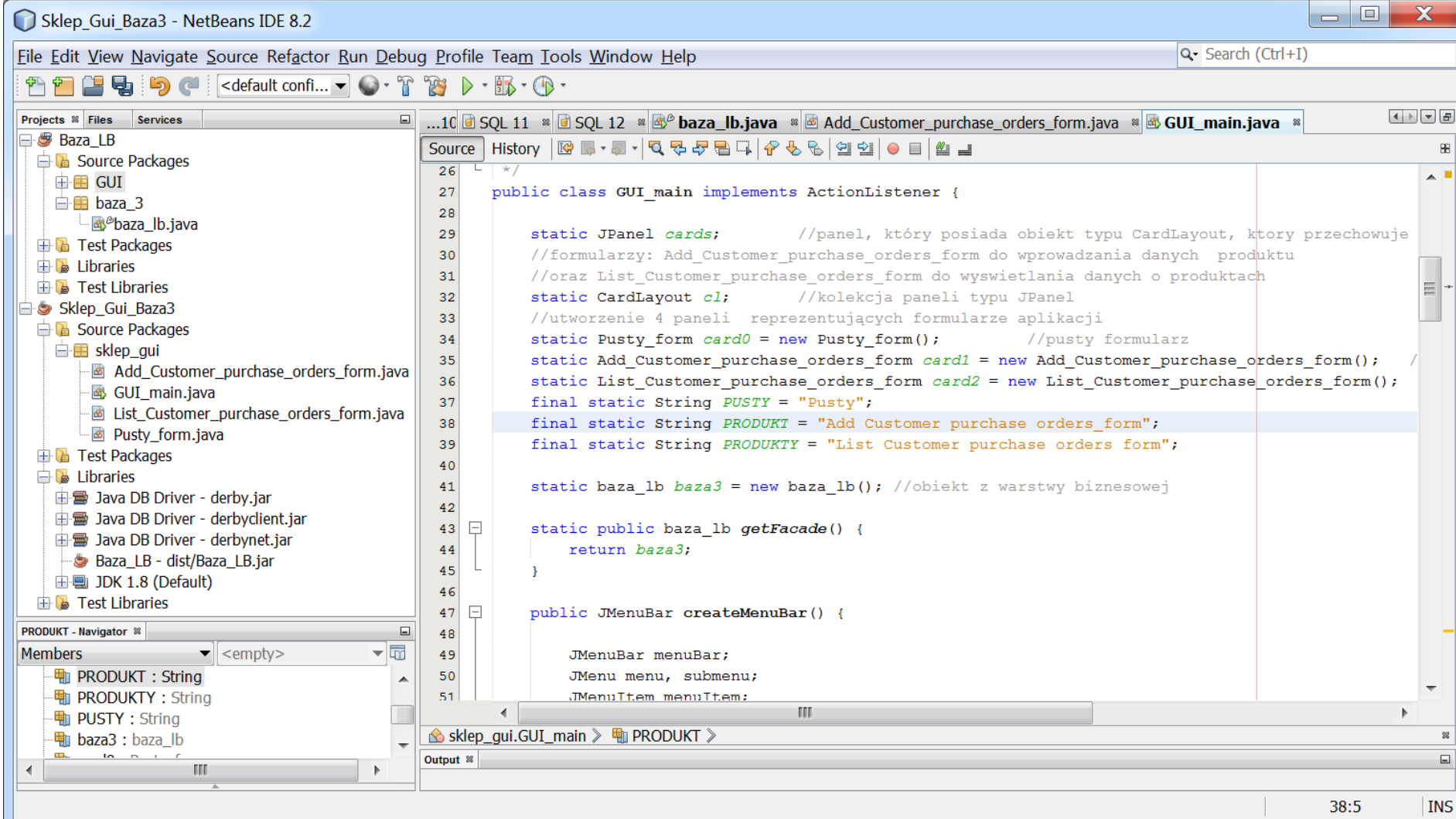
The right screenshot shows a table of order data. The table has the following columns: Name, CREDIT_LIMIT, SALES_DATE, SHIPPING_DATE, SHIPPING_COST, and QUANTITY. The data is as follows:

Name	CREDIT_LIMIT	SALES_DATE	SHIPPING_DATE	SHIPPING_COST	QUANTITY
Big Car Parts	50000	2011-05-24	2011-05-24	2500.00	250
Big Network Systems	25000	2011-05-24	2011-05-24	25.00	100
Bob Hosting Corp.	65000	2011-05-24	2011-05-24	65.00	120
Bob Hosting Corp.	65000	2011-05-24	2011-05-24	55.00	60
Early CentralComp	26500	2011-05-24	2011-05-24	265.00	500
Jan Kowalski	101	2017-05-15	2017-05-16	456.00	34
Jan Kowalski	101	2017-05-19	2017-05-19	44.00	444
Jan Kowalski	101	2017-05-12	2017-05-14	123.00	23
Jan Kowalski	101	2017-05-12	2017-05-14	123.00	23
Jan Kowalski	101	2017-05-14	2017-05-15	23.00	234

Below the table is a text area labeled "Produkty" containing a string representation of a row of data: "[Big Car Parts, 50000, 2011-05-24, 2011-05-24, 2500.00, 250]".

Zadanie 5 cd

- 3) Program **Sklep_Gui_Baza3** – warstwa klienta programu i projekt **Baza_LB** – warstwa biznesowa programu. Projekt **Baza_LB** jest okrojona wersja programu **Baza_3** z z zadanie 3.



The screenshot shows the NetBeans IDE 8.2 interface. The main window displays the source code for `GUI_main.java`. The code is as follows:

```
26  /*
27  public class GUI_main implements ActionListener {
28
29      static JPanel cards;           //panel, który posiada obiekt typu CardLayout, który przechowuje
30      //formularzy: Add_Customer_purchase_orders_form do wprowadzania danych produktu
31      //oraz List_Customer_purchase_orders_form do wyświetlania danych o produktach
32      static CardLayout cl;         //kolekcja paneli typu JPanel
33      //utworzenie 4 paneli reprezentujących formularze aplikacji
34      static Pusty_form card0 = new Pusty_form();           //pusty formularz
35      static Add_Customer_purchase_orders_form card1 = new Add_Customer_purchase_orders_form(); //
36      static List_Customer_purchase_orders_form card2 = new List_Customer_purchase_orders_form();
37      final static String PUSTY = "Pusty";
38      final static String PRODUKT = "Add Customer purchase orders form";
39      final static String PRODUKTY = "List Customer purchase orders form";
40
41      static baza_lb baza3 = new baza_lb(); //obiekt z warstwy biznesowej
42
43      static public baza_lb getFacade() {
44          return baza3;
45      }
46
47      public JMenuBar createMenuBar() {
48
49          JMenuBar menuBar;
50          JMenu menu, submenu;
51          JMenuItem menuItem;
```

The left sidebar shows the project structure for `Baza_LB` and `Sklep_Gui_Baza3`. The `Sklep_Gui_Baza3` project contains the `sklep_gui` package with the following files:

- `Add_Customer_purchase_orders_form.java`
- `GUI_main.java`
- `List_Customer_purchase_orders_form.java`
- `Pusty_form.java`

The `PRODUKT - Navigator` window shows the following members:

- `PRODUKT : String`
- `PRODUKTY : String`
- `PUSTY : String`
- `baza3 : baza_lb`

The status bar at the bottom right shows the time `38:5` and the mode `INS`.

Zadanie 5 cd

4) Program **Sklep_Gui_Baza3** – warstwa klienta programu i projekt **Baza_LB** – warstwa biznesowa programu. Projekt **Baza_LB** jest okrojoną wersją programu **Baza_3** z zadanie 3 i posiada dodatkową metodę **items**

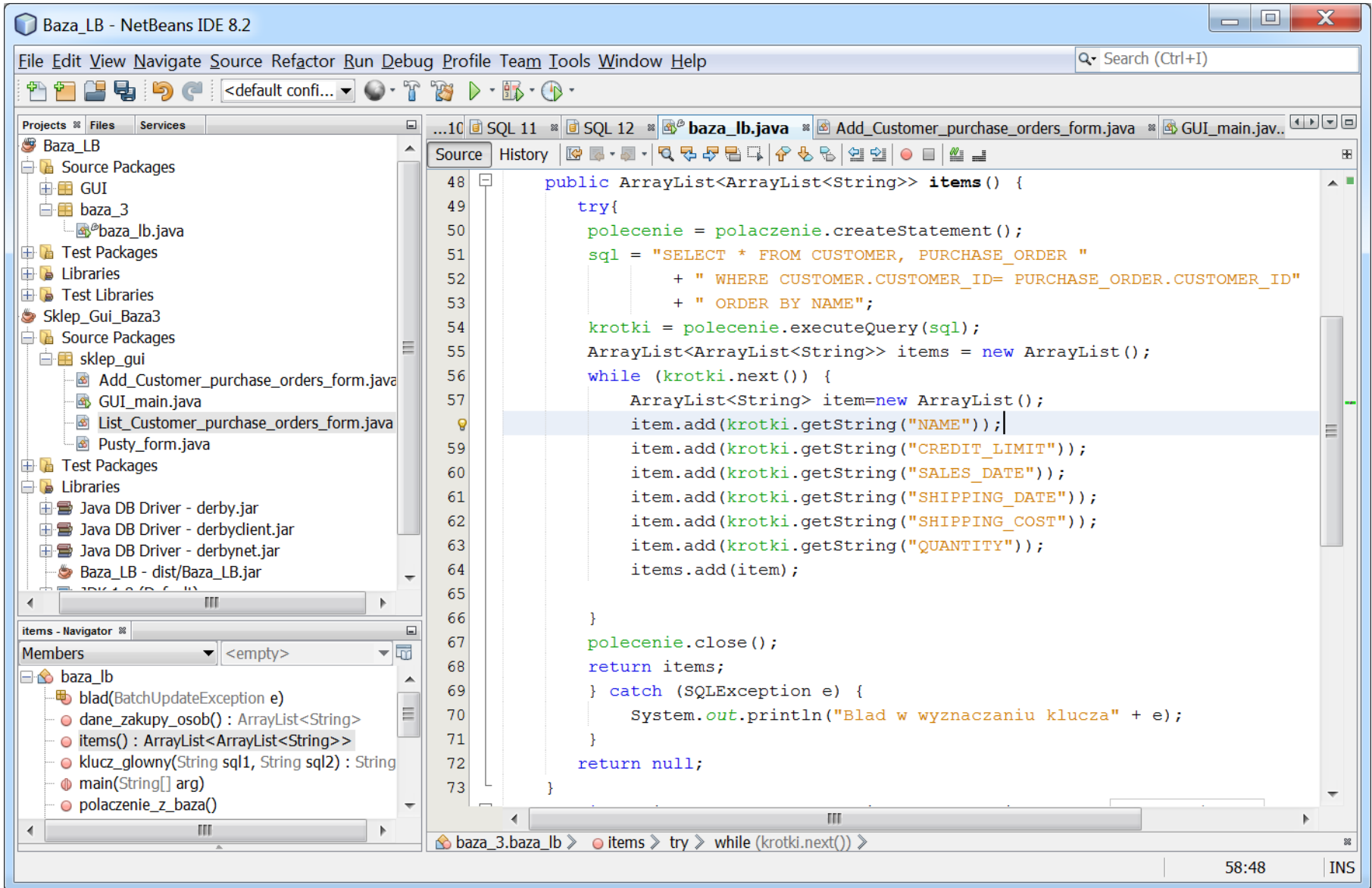
The screenshot shows the NetBeans IDE 8.2 interface. The left sidebar displays the project structure for 'Baza_LB', including source packages (GUI, baza_3), test packages, and libraries. The main editor window shows the source code of 'baza_lb.java'. The code is as follows:

```
1 package baza_3;
2 import GUI.GUIPurchase_order;
3 import GUI.WeWy;
4 import java.sql.*;
5 import java.util.ArrayList;
6
7 public class baza_lb {
8     String data, sql;
9     Connection polaczenie;
10    Statement polecenie;
11    ResultSet krotki;
12
13    public void polaczenie_z_baza() throws SQLException { ...15 lines }
14
15    public ArrayList<String> dane_zakupy_osob() throws SQLException { ...20 lines }
16
17    public ArrayList<ArrayList<String>> items() { ...26 lines }
18
19    public String klucz_glowny(String sql1, String sql2) { ...12 lines }
20
21    void blad(BatchUpdateException e) { ...8 lines }
22
23    public String wyszukaj_osobe(String name) { ...12 lines }
24
25    public String wyszukaj_produkt(String kod) { ...12 lines }
26
27    public void wstaw_zlecenie(String[] dane) throws SQLException { ...29 lines }
28
29    static public void main(String arg[]) {
30        baza_lb baza = new baza_lb();
31        GUIPurchase_order guipurchase_order = new GUIPurchase_order();
32        String [] tabela;
33        ArrayList<String> lista;
34        try {
35            baza.polaczenie_z_baza();
36            tabela=guipurchase_order.wstaw_dane_zlecenia(); baza.wstaw_zlecenie(tabela);
37            lista=baza.dane_zakupy_osob(); WeWy.wyString(lista);
38        } catch (SQLException e) {
39            System.out.println(e.getMessage());
40            while (null != (e = e.getNextException()))
41                System.out.println(e.getMessage());
42        }
43    }
44 }
```

The bottom right corner of the IDE shows the page number '1:11' and the text 'INS'.

Zadanie 5 cd

5) Dodana metoda **items** (podobna do metody **dane_zakupy_osob**) dostarcza model do komponentu **JTable** w formularzu **List_Customer_purchase_orders_form** w projekcie **Sklep_Gui_Baza3**



The screenshot displays the NetBeans IDE 8.2 interface for the project "Baza_LB". The main editor window shows the source code for the `items()` method in `baza_lb.java`. The code is as follows:

```
48 public ArrayList<ArrayList<String>> items() {
49     try{
50         polecenie = polaczenie.createStatement();
51         sql = "SELECT * FROM CUSTOMER, PURCHASE_ORDER "
52             + " WHERE CUSTOMER.CUSTOMER_ID= PURCHASE_ORDER.CUSTOMER_ID"
53             + " ORDER BY NAME";
54         krotki = polecenie.executeQuery(sql);
55         ArrayList<ArrayList<String>> items = new ArrayList();
56         while (krotki.next()) {
57             ArrayList<String> item=new ArrayList();
58             item.add(krotki.getString("NAME"));
59             item.add(krotki.getString("CREDIT_LIMIT"));
60             item.add(krotki.getString("SALES_DATE"));
61             item.add(krotki.getString("SHIPPING_DATE"));
62             item.add(krotki.getString("SHIPPING_COST"));
63             item.add(krotki.getString("QUANTITY"));
64             items.add(item);
65         }
66     }
67     polecenie.close();
68     return items;
69 } catch (SQLException e) {
70     System.out.println("Bład w wyznaczaniu klucza" + e);
71 }
72 return null;
73 }
```

The left sidebar shows the project structure, including the `GUI` package and the `List_Customer_purchase_orders_form.java` file. The bottom-left pane shows the "Members" view for the `baza_lb` class, listing methods such as `blad`, `dane_zakupy_osob`, `items`, `klucz_glowny`, `main`, and `polaczenie_z_baza`. The status bar at the bottom indicates the current cursor position: `baza_3.baza_lb > items > try > while (krotki.next()) >`. The system clock shows 58:48 and the user is identified as INS.

Zadanie 5 cd

6) Teraz metoda **wstaw_zlecenie** wywołana w formularzu **Add_Customer_purchase_orders_form** w projekcie **Sklep_Gui_Baza3** pozwala wstawić kolejną krotkę do tabeli **PURCHASE_ORDER**

The screenshot shows the NetBeans IDE 8.2 interface. The main window displays the source code of the `Add_Customer_purchase_orders_form.java` file. The code is as follows:

```
128     String[] data2 = data1.split("-"); //podzial
129     rok = Integer.parseInt(data2[2]);
130     miesiac = Integer.parseInt(data2[1]);
131     dzien = Integer.parseInt(data2[0]);
132     } catch (PatternSyntaxException | NumberFormatException e) {
133         return null;
134     }
135     GregorianCalendar gc = new GregorianCalendar();
136     gc.set(rok, miesiac - 1, dzien);
137     SimpleDateFormat format = new SimpleDateFormat("yy-MM-dd");
138     String data2 = format.format(gc.getTime()); //tworzenie daty
139     return data2; //pobranie daty
140 }
141
142 @Override
143 public void actionPerformed(ActionEvent evt) {
144     String[] dane = form_produkt();
145     if (dane == null) {
146         return;
147     }
148
149     try {
150         GUI_main.getFacade().wstaw_zlecenie(dane);
151     } catch (SQLException e) {
152         System.out.println("Blad 2");
153     }
154 }
155
156 }
```

The left sidebar shows the project structure for `Sklep_Gui_Baza3`, including source packages like `sklep_gui` and test packages. The `Members` window shows the methods of the `Add_Customer_purchase_orders_form` class, with `actionPerformed(ActionEvent evt)` selected. The status bar at the bottom indicates the current location in the code: `sklep_gui.Add_Customer_purchase_orders_form > actionPerformed > try >`, with line numbers 151:1 and INS.

Schemat bazy danych **Sample**

A. Wyszukiwanie w tabeli wszystkich krotek Product wg Manufacturer

B. Wyszukiwanie w tabeli wszystkich krotek Product wg Product_Code

