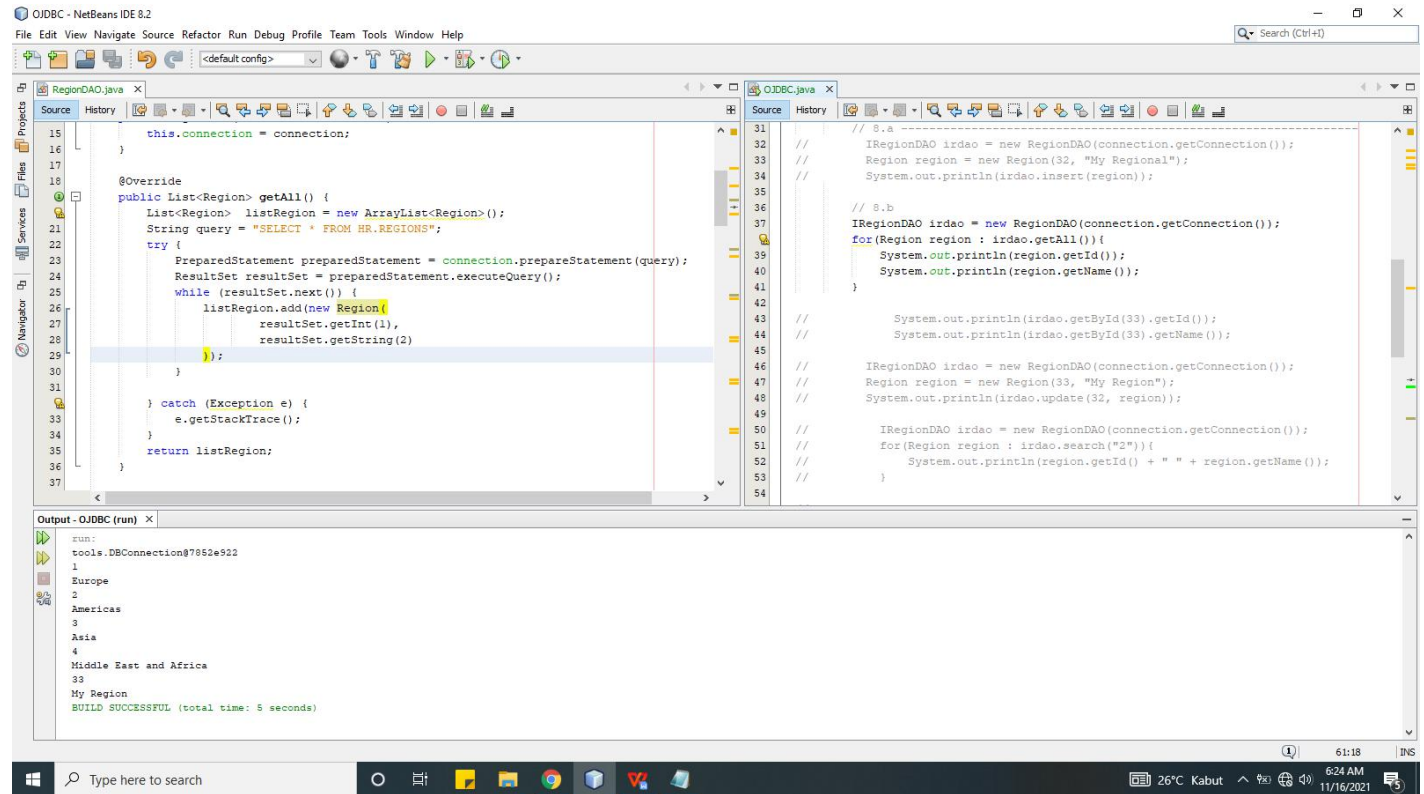


TUGAS 7 (Tambahan) - Hadi Prasetyo

Buat CRUD (6 method) untuk 2 table (Region & Countries)
Lengkap dengan manual test & javadoc

Screenshoot berisi : Code, Hasil, Waktu pengerjaan
Masukan kedalam bentuk word.

1. Get all data pada tabel Region

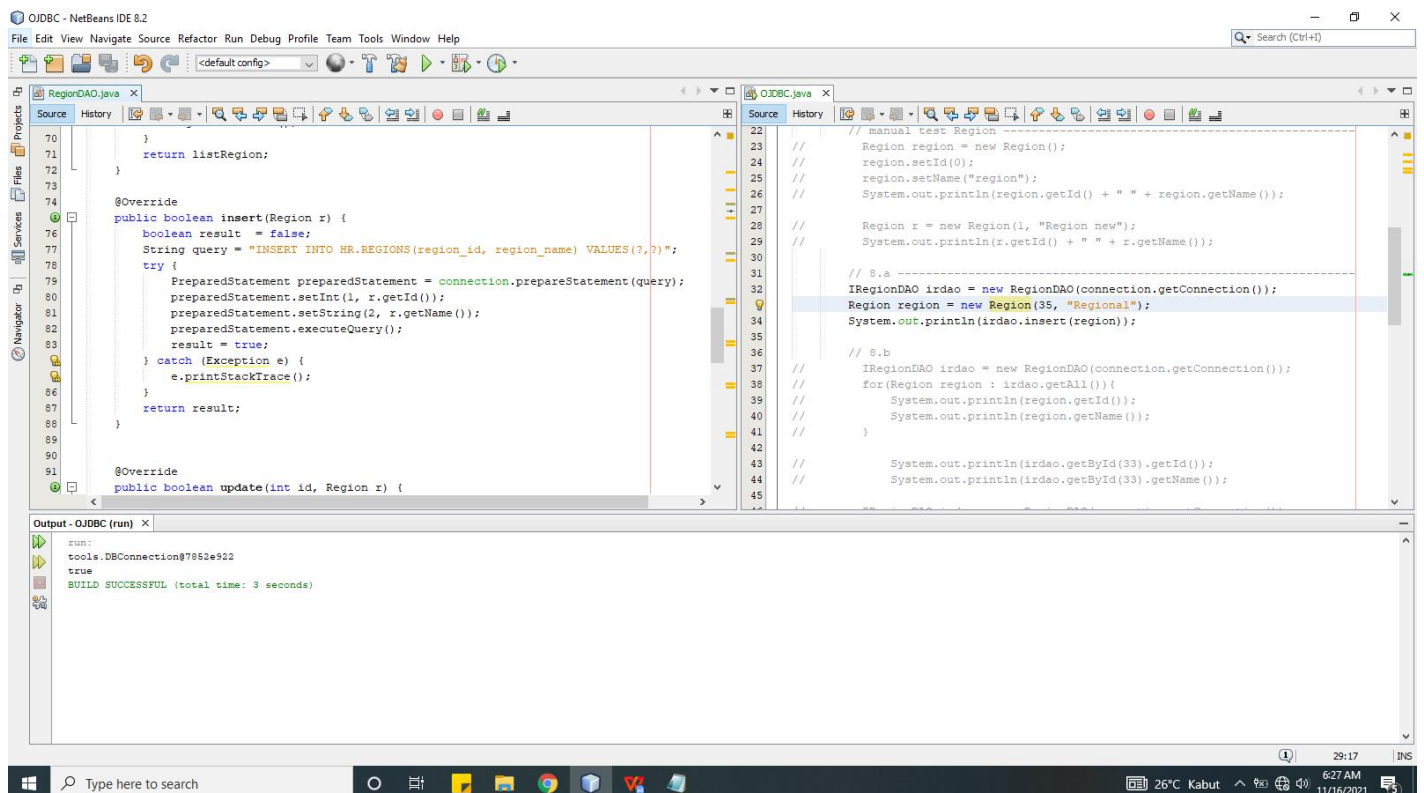


```
RegionDAO.java
15 this.connection = connection;
16
17
18 @Override
19 public List<Region> getAll() {
20     List<Region> listRegion = new ArrayList<Region>();
21     String query = "SELECT * FROM HR.REGIONS";
22     try {
23         PreparedStatement preparedStatement = connection.prepareStatement(query);
24         ResultSet resultSet = preparedStatement.executeQuery();
25         while (resultSet.next()) {
26             listRegion.add(new Region(
27                 resultSet.getInt(1),
28                 resultSet.getString(2)
29             ));
30         }
31     } catch (Exception e) {
32         e.printStackTrace();
33     }
34     return listRegion;
35 }
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
```

```
OJDBC.java
31 //
32 // IRegionDAO irdao = new RegionDAO(connection.getConnection());
33 // Region region = new Region(32, "My Regional");
34 // System.out.println(irdao.insert(region));
35 //
36 //
37 // 8.a
38 // IRegionDAO irdao = new RegionDAO(connection.getConnection());
39 // for (Region region : irdao.getAll()) {
40 //     System.out.println(region.getId());
41 //     System.out.println(region.getName());
42 // }
43 //
44 // System.out.println(irdao.getById(33).getId());
45 // System.out.println(irdao.getById(33).getName());
46 //
47 // IRegionDAO irdao = new RegionDAO(connection.getConnection());
48 // Region region = new Region(33, "My Region");
49 // System.out.println(irdao.update(32, region));
50 //
51 // IRegionDAO irdao = new RegionDAO(connection.getConnection());
52 // for (Region region : irdao.search("2")) {
53 //     System.out.println(region.getId() + " " + region.getName());
54 // }
55 //
56 //
57 //
58 //
59 //
60 //
61 //
62 //
63 //
64 //
65 //
66 //
67 //
68 //
69 //
70 //
71 //
72 //
73 //
74 //
75 //
76 //
77 //
78 //
79 //
80 //
81 //
82 //
83 //
84 //
85 //
86 //
87 //
88 //
89 //
90 //
91 //
92 //
93 //
94 //
95 //
96 //
97 //
98 //
99 //
100 //
```

```
Output - OJDBC (run)
run:
tools.DBConnection@7952e922
1
Europe
2
Americas
3
Asia
4
Middle East and Africa
33
My Region
BUILD SUCCESSFUL (total time: 5 seconds)
```

2. Insert pada tabel region



```
RegionDAO.java
70
71 return listRegion;
72
73
74 @Override
75 public boolean insert(Region r) {
76     boolean result = false;
77     String query = "INSERT INTO HR.REGIONS(region_id, region_name) VALUES(?,?)";
78     try {
79         PreparedStatement preparedStatement = connection.prepareStatement(query);
80         preparedStatement.setInt(1, r.getId());
81         preparedStatement.setString(2, r.getName());
82         preparedStatement.executeQuery();
83         result = true;
84     } catch (Exception e) {
85         e.printStackTrace();
86     }
87     return result;
88 }
89
90
91 @Override
92 public boolean update(int id, Region r) {
93
94
95
96
97
98
99
100
```

```
OJDBC.java
22 // manual test Region
23 Region region = new Region();
24 region.setId(0);
25 region.setName("region");
26 System.out.println(region.getId() + " " + region.getName());
27
28 Region r = new Region(1, "Region new");
29 System.out.println(r.getId() + " " + r.getName());
30 //
31 // 8.a
32 IRegionDAO irdao = new RegionDAO(connection.getConnection());
33 Region region = new Region(35, "Regional");
34 System.out.println(irdao.insert(region));
35 //
36 // 8.b
37 IRegionDAO irdao = new RegionDAO(connection.getConnection());
38 for (Region region : irdao.getAll()) {
39     System.out.println(region.getId());
40     System.out.println(region.getName());
41 }
42 //
43 System.out.println(irdao.getById(33).getId());
44 System.out.println(irdao.getById(33).getName());
45 //
46 //
47 //
48 //
49 //
50 //
51 //
52 //
53 //
54 //
55 //
56 //
57 //
58 //
59 //
60 //
61 //
62 //
63 //
64 //
65 //
66 //
67 //
68 //
69 //
70 //
71 //
72 //
73 //
74 //
75 //
76 //
77 //
78 //
79 //
80 //
81 //
82 //
83 //
84 //
85 //
86 //
87 //
88 //
89 //
90 //
91 //
92 //
93 //
94 //
95 //
96 //
97 //
98 //
99 //
100 //
```

```
Output - OJDBC (run)
run:
tools.DBConnection@7952e922
true
BUILD SUCCESSFUL (total time: 3 seconds)
```

3. Get by id

```
RegionDAO.java
35 return listRegion;
36 }
37
38 @Override
39 public Region getById(int id) {
40     Region region = new Region();
41     String query = "SELECT * FROM HR.REGIONS WHERE region_id = ?";
42     try {
43         PreparedStatement preparedStatement = connection.prepareStatement(query);
44         preparedStatement.setInt(1, id);
45         ResultSet resultSet = preparedStatement.executeQuery();
46         if(resultSet.next()){
47             region = new Region(resultSet.getInt(1), resultSet.getString(2));
48         }
49     } catch (Exception e) {
50         e.printStackTrace();
51     }
52     return region;
53 }
54
55 @Override
56 public List<Region> search(String key) {
57     List<Region> listRegion = new ArrayList<>();
58     String query = "SELECT * FROM HR.REGIONS WHERE region_name LIKE ?";
59     try {
60         PreparedStatement preparedStatement = connection.prepareStatement(query);
61         preparedStatement.setString(1, "%" + key + "%");
62         ResultSet resultSet = preparedStatement.executeQuery();
63         while(resultSet.next()){
64             Region region = new Region(resultSet.getInt(1), resultSet.getString(2));
65             listRegion.add(region);
66         }
67     } catch (Exception e) {
68         e.printStackTrace();
69     }
70     return listRegion;
71 }
72 }
```

```
OJDBC.java
31 // 8.a
32 IRegionDAO irdao = new RegionDAO(connection.getConnection());
33 Region region = new Region(35, "Regional");
34 System.out.println(irdao.insert(region));
35
36 // 8.b
37 IRegionDAO irdao = new RegionDAO(connection.getConnection());
38 for(Region region : irdao.getAll()){
39     System.out.println(region.getId());
40     System.out.println(region.getName());
41 }
42
43 // GetById
44 System.out.println(irdao.getById(33).getId() + " " +
45     irdao.getById(33).getName());
46
47 //
48 IRegionDAO irdao = new RegionDAO(connection.getConnection());
49 Region region = new Region(33, "My Region");
50 System.out.println(irdao.update(32, region));
51
52 //
53 IRegionDAO irdao = new RegionDAO(connection.getConnection());
54 for(Region region : irdao.search("2")){
55     System.out.println(region.getId() + " " + region.getName());
56 }
```

```
Output - OJDBC (run)
run:
tools.DBConnection@7852e922
33 My Region
BUILD SUCCESSFUL (total time: 3 seconds)
```

4. Update data region

```
RegionDAO.java
86 return result;
87 }
88
89 @Override
90 public boolean update(int id, Region r) {
91     boolean result = false;
92     String query = "UPDATE HR.REGIONS SET region_id=?, region_name=? WHERE region_id = ?";
93     try {
94         PreparedStatement preparedStatement = connection.prepareStatement(query);
95         preparedStatement.setInt(1, r.getId());
96         preparedStatement.setString(2, r.getName());
97         preparedStatement.setInt(3, id);
98         preparedStatement.executeUpdate();
99         result = true;
100     } catch (Exception e) {
101         e.printStackTrace();
102     }
103     return result;
104 }
105
106 @Override
107 public List<Region> search(String key) {
108     List<Region> listRegion = new ArrayList<>();
109     String query = "SELECT * FROM HR.REGIONS WHERE region_name LIKE ?";
110     try {
111         PreparedStatement preparedStatement = connection.prepareStatement(query);
112         preparedStatement.setString(1, "%" + key + "%");
113         ResultSet resultSet = preparedStatement.executeQuery();
114         while(resultSet.next()){
115             Region region = new Region(resultSet.getInt(1), resultSet.getString(2));
116             listRegion.add(region);
117         }
118     } catch (Exception e) {
119         e.printStackTrace();
120     }
121     return listRegion;
122 }
```

```
OJDBC.java
33 Region region = new Region(35, "Regional");
34 System.out.println(irdao.insert(region));
35
36 // 8.b
37 IRegionDAO irdao = new RegionDAO(connection.getConnection());
38 for(Region region : irdao.getAll()){
39     System.out.println(region.getId());
40     System.out.println(region.getName());
41 }
42
43 //
44 IRegionDAO irdao = new RegionDAO(connection.getConnection());
45 Region region = new Region(35, "Regional");
46 System.out.println(irdao.update(32, region));
47
48 // GetById
49 System.out.println(irdao.getById(35).getId() + " " +
50     irdao.getById(35).getName());
51
52 //
53 IRegionDAO irdao = new RegionDAO(connection.getConnection());
54 for(Region region : irdao.search("2")){
55     System.out.println(region.getId() + " " + region.getName());
56 }
```

```
Output - OJDBC (run)
run:
tools.DBConnection@7852e922
true
35 Regional
BUILD SUCCESSFUL (total time: 3 seconds)
```

5. Search Region

The screenshot displays an IDE with two open Java files: `RegionDAO.java` and `JDBC.java`. The `RegionDAO.java` file contains a `search` method that takes a `String key` and returns a `List<Region>`. The `JDBC.java` file contains test code that creates a `RegionDAO` instance and calls the `search` method with the key "2". The output window shows the results of the search, listing the regions: 1 Americas, 35 Regional, and BUILD SUCCESSFUL (total time: 3 seconds).

```
RegionDAO.java
51 }
52
53 return region;
54 }
55
56 @Override
57 public List<Region> search(String key) {
58     List<Region> listRegion = new ArrayList<>();
59     String query = "SELECT * FROM HR.REGIONS WHERE region_id LIKE ? OR region_name LIKE ?";
60     try {
61         PreparedStatement preparedStatement = connection.prepareStatement(query);
62         preparedStatement.setString(1, "%" + key + "%");
63         preparedStatement.setString(2, "%" + key + "%");
64         ResultSet resultSet = preparedStatement.executeQuery();
65         while (resultSet.next()) {
66             listRegion.add(new Region(resultSet.getInt(1), resultSet.getString(2)));
67         }
68     } catch (Exception e) {
69         e.printStackTrace();
70     }
71     return listRegion;
72 }
73
74 @Override
```

```
JDBC.java
42 //
43 IRegionDAO irdao = new RegionDAO(connection.getConnection());
44 //
45 Region region = new Region(35, "Regional");
46 System.out.println(irdao.update(32, region));
47
48 // GetById
49 System.out.println(irdao.getById(35).getId() + " " +
50     irdao.getById(35).getName());
51
52 IRegionDAO irdao = new RegionDAO(connection.getConnection());
53 for (Region region : irdao.search("2")) {
54     System.out.println(region.getId() + " " + region.getName());
55 }
56
57 for (Region region : irdao.search("na1")) {
58     System.out.println(region.getId() + " " + region.getName());
59 }
60
61 IRegionDAO irdao = new RegionDAO(connection.getConnection());
62 irdao.delete(31);
63
64 //
```

```
Output - JDBC (run)
run:
tools.DBConnection@7852e922
2 Americas
35 Regional
BUILD SUCCESSFUL (total time: 3 seconds)
```

6. Delete data

The screenshot displays an IDE with two open Java files: `RegionDAO.java` and `JDBC.java`. The `RegionDAO.java` file contains a `delete` method that takes an `int id` and returns a `boolean`. The `JDBC.java` file contains test code that creates a `RegionDAO` instance and calls the `delete` method with the id 35. The output window shows the results of the delete operation, listing the regions: 33 My Region, 35 Regional, 1 Europe, 2 Americas, 3 Asia, 4 Middle East and Africa, 33 My Region, and BUILD SUCCESSFUL (total time: 3 seconds).

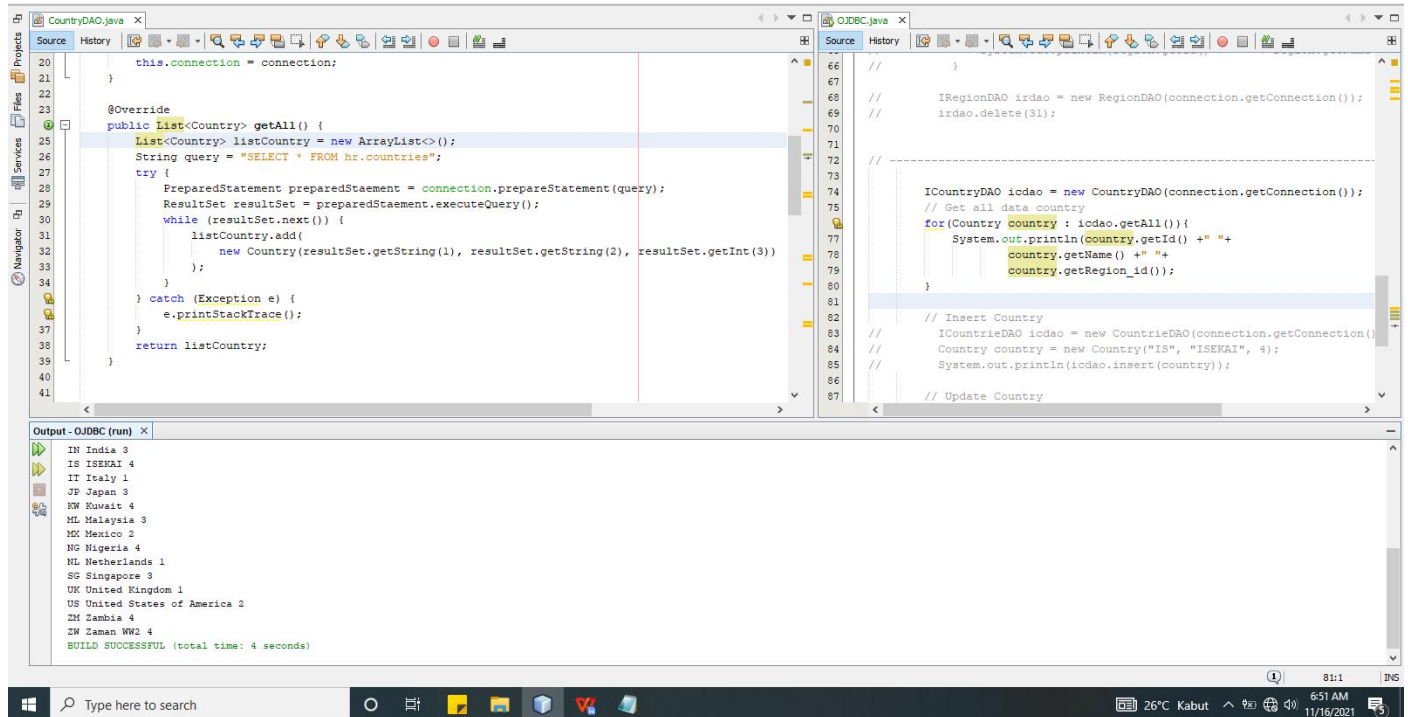
```
RegionDAO.java
107
108 @Override
109 public boolean delete(int id) {
110     boolean result = false;
111     String query = "DELETE FROM HR.REGIONS WHERE region_id = ?";
112     try {
113         PreparedStatement preparedStatement = connection.prepareStatement(query);
114         preparedStatement.setInt(1, id);
115         preparedStatement.executeUpdate();
116         result = true;
117         if (result) {
118             System.out.println("Data ber id " + id + " berhasil di hapus");
119         } else {
120             System.out.println("Data tidak ada");
121         }
122     } catch (Exception e) {
123         e.printStackTrace();
124     }
125     return result;
126 }
127
128
```

```
JDBC.java
29 //
30 System.out.println(r.getId() + " " + r.getName());
31
32 // 8.a -----
33 IRegionDAO irdao = new RegionDAO(connection.getConnection());
34 Region region = new Region(35, "Regional");
35 System.out.println(irdao.insert(region));
36
37 // 8.b
38 IRegionDAO irdao = new RegionDAO(connection.getConnection());
39 for (Region region : irdao.getAll()) {
40     System.out.println(region.getId() + " " + region.getName());
41 }
42
43 irdao.delete(35);
44
45 for (Region region : irdao.getAll()) {
46     System.out.println(region.getId() + " " + region.getName());
47 }
48
49
50 IRegionDAO irdao = new RegionDAO(connection.getConnection());
```

```
Output - JDBC (run)
33
My Region
35
Regional
Data ber id 35 berhasil di hapus
1
Europe
2
Americas
3
Asia
4
Middle East and Africa
33
My Region
BUILD SUCCESSFUL (total time: 3 seconds)
```

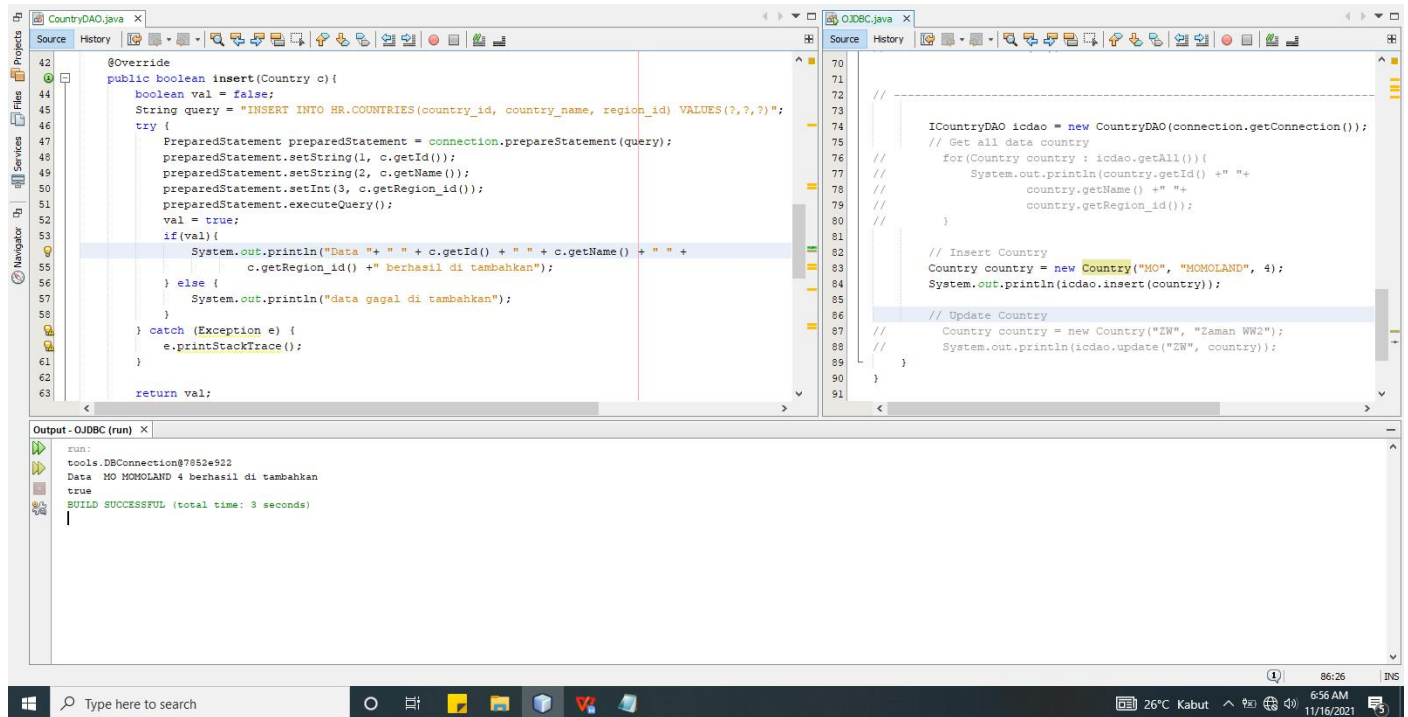
Tabel countries

1. Get all data countries



```
CountryDAO.java
20 this.connection = connection;
21 }
22
23 @Override
24 public List<Country> getAll() {
25     List<Country> listCountry = new ArrayList<>();
26     String query = "SELECT * FROM hr.countries";
27     try {
28         PreparedStatement preparedStatement = connection.prepareStatement(query);
29         ResultSet resultSet = preparedStatement.executeQuery();
30         while (resultSet.next()) {
31             listCountry.add(
32                 new Country(resultSet.getString(1), resultSet.getString(2), resultSet.getInt(3))
33             );
34         }
35     } catch (Exception e) {
36         e.printStackTrace();
37     }
38     return listCountry;
39 }
40
41
OJDBC.java
66 //
67 //
68 // IRegionDAO irdao = new RegionDAO(connection.getConnection());
69 // irdao.delete(31);
70
71
72 //
73 //
74 // ICountryDAO icdao = new CountryDAO(connection.getConnection());
75 // Get all data country
76 for (Country country : icdao.getAll()) {
77     System.out.println(country.getId() + " " +
78         country.getName() + " " +
79         country.getRegion_id());
80 }
81
82 // Insert Country
83 //
84 // ICountryDAO icdao = new CountryDAO(connection.getConnection());
85 // Country country = new Country("IS", "ISEKAI", 4);
86 // System.out.println(icdao.insert(country));
87
88 // Update Country
89
Output - OJDBC (run)
IN India 3
IS ISEKAI 4
IT Italy 1
JP Japan 3
KW Kuwait 4
ML Malaysia 3
MX Mexico 2
NG Nigeria 4
NL Netherlands 1
SG Singapore 3
UK United Kingdom 1
US United States of America 2
ZM Zambia 4
ZW Zaman WW2 4
BUILD SUCCESSFUL (total time: 4 seconds)
```

2. Insert data countries



```
CountryDAO.java
42 @Override
43 public boolean insert(Country c) {
44     boolean val = false;
45     String query = "INSERT INTO HR.COUNTRIES(country_id, country_name, region_id) VALUES(?, ?, ?)";
46     try {
47         PreparedStatement preparedStatement = connection.prepareStatement(query);
48         preparedStatement.setString(1, c.getId());
49         preparedStatement.setString(2, c.getName());
50         preparedStatement.setInt(3, c.getRegion_id());
51         preparedStatement.executeQuery();
52         val = true;
53         if (val) {
54             System.out.println("Data " + c.getId() + " " + c.getName() + " " +
55                 c.getRegion_id() + " berhasil di tambahkan");
56         } else {
57             System.out.println("data gagal di tambahkan");
58         }
59     } catch (Exception e) {
60         e.printStackTrace();
61     }
62     return val;
63 }
64
OJDBC.java
70 //
71 //
72 //
73 //
74 // ICountryDAO icdao = new CountryDAO(connection.getConnection());
75 // Get all data country
76 // for (Country country : icdao.getAll()) {
77 //     System.out.println(country.getId() + " " +
78 //         country.getName() + " " +
79 //         country.getRegion_id());
80 // }
81
82 // Insert Country
83 Country country = new Country("MO", "MOMOLAND", 4);
84 System.out.println(icdao.insert(country));
85
86 // Update Country
87 //
88 // Country country = new Country("ZW", "Zaman WW2");
89 // System.out.println(icdao.update("ZW", country));
90
91
Output - OJDBC (run)
run:
tools.DBConnection@7852e922
Data MO MOMOLAND 4 berhasil di tambahkan
true
BUILD SUCCESSFUL (total time: 3 seconds)
```


3. Update data countries

```
CountryDAO.java
63     return val;
64 }
65
66 public boolean update(String id, Country c) {
67     boolean res = false;
68     String query = "UPDATE HR.COUNTRIES SET country_name=? WHERE country_id = ?";
69     try {
70         PreparedStatement preparedStatement = connection.prepareStatement(query);
71         preparedStatement.setString(1, c.getId());
72         preparedStatement.setString(2, c.getName());
73         preparedStatement.setString(3, id);
74         preparedStatement.executeQuery();
75         res = true;
76         if(res){
77             System.out.println("Data berhasil di ubah");
78         }
79     } catch (Exception e) {
80         e.printStackTrace();
81     }
82     return res;
83 }
84
```

```
JDBC.java
72 //
73 //
74 ICountryDAO icdao = new CountryDAO(connection.getConnection());
75 // Get all data country
76 for(Country country : icdao.getAll()){
77     System.out.println(country.getId() + " " +
78         country.getName() + " " +
79         country.getRegion_id());
80 }
81 //
82 // Insert Country
83 Country country = new Country("MO", "MOMOLAND", 4);
84 System.out.println(icdao.insert(country));
85 //
86 // Update Country
87 Country country = new Country("ZW", "Zaman WW3");
88 System.out.println(icdao.update("ZW", country));
89 }
90
91
```

```
Output - JDBC (run)
run:
tools.DBConnection@7852e922
Data berhasil di ubah
true
BUILD SUCCESSFUL (total time: 3 seconds)
```

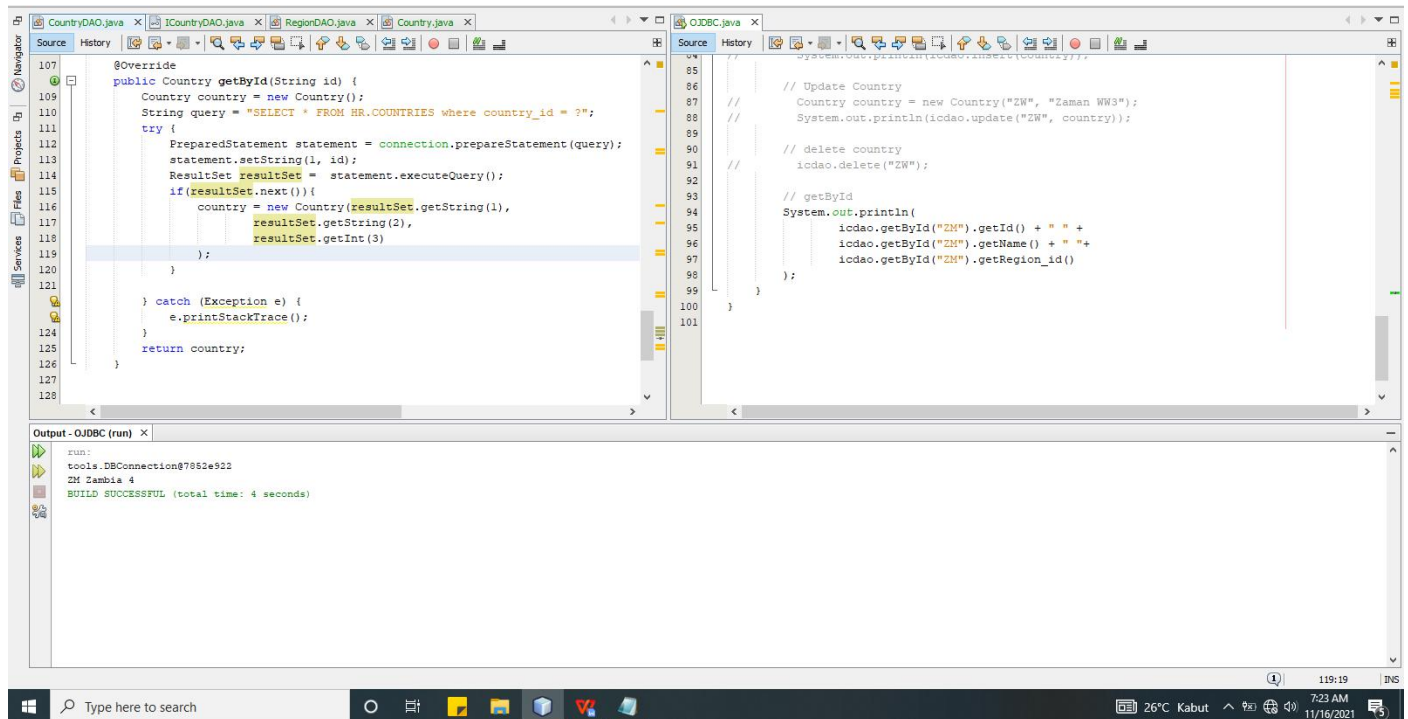
4. Delete data countries

```
CountryDAO.java
85 @Override
86 public boolean delete(String id) {
87     boolean res = false;
88     String query = "DELETE FROM HR.COUNTRIES WHERE country_id = ?";
89     try {
90         PreparedStatement preparedStatement = connection.prepareStatement(query);
91         preparedStatement.setString(1, id);
92         preparedStatement.executeQuery();
93         res = true;
94         if(res){
95             System.out.println("Data berhasil di hapus");
96         }
97     } catch (Exception e) {
98         e.printStackTrace();
99     }
100     return res;
101 }
102
103
104
105
106
```

```
JDBC.java
72 //
73 //
74 ICountryDAO icdao = new CountryDAO(connection.getConnection());
75 // Get all data country
76 for(Country country : icdao.getAll()){
77     System.out.println(country.getId() + " " +
78         country.getName() + " " +
79         country.getRegion_id());
80 }
81 //
82 // Insert Country
83 Country country = new Country("MO", "MOMOLAND", 4);
84 System.out.println(icdao.insert(country));
85 //
86 // Update Country
87 Country country = new Country("ZW", "Zaman WW3");
88 System.out.println(icdao.update("ZW", country));
89
90 icdao.delete("ZW");
91 }
92
93
```

```
Output - JDBC (run)
run:
tools.DBConnection@7852e922
Data berhasil di hapus
BUILD SUCCESSFUL (total time: 3 seconds)
```

5. getById countries



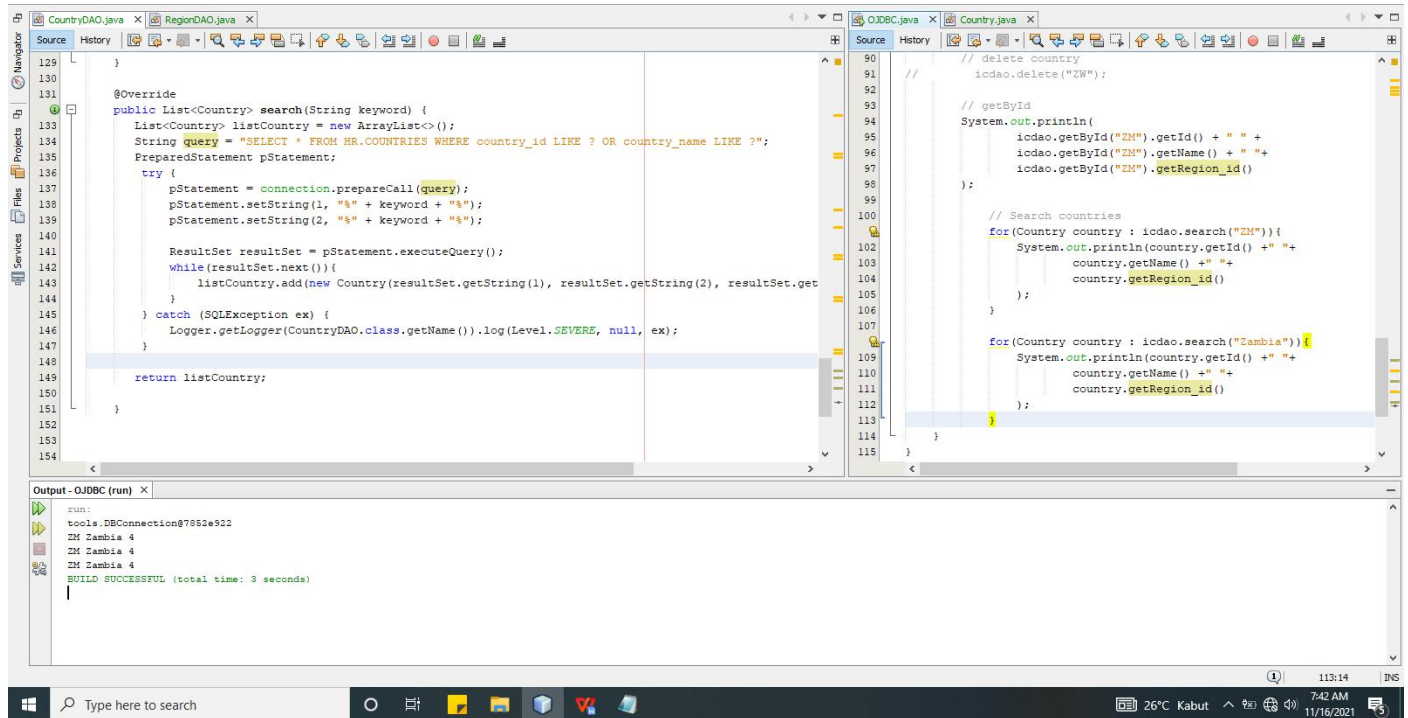
The screenshot shows an IDE with two open files: `CountryDAO.java` and `OJDBC.java`. The `CountryDAO.java` file contains the implementation of the `getById` method, which takes a country ID as a parameter and returns a `Country` object. The `OJDBC.java` file shows the usage of the `getById` method, including updating a country, deleting a country, and retrieving a country by ID.

```
CountryDAO.java
107 @Override
108 public Country getById(String id) {
109     Country country = new Country();
110     String query = "SELECT * FROM HR.COUNTRIES where country_id = ?";
111     try {
112         PreparedStatement statement = connection.prepareStatement(query);
113         statement.setString(1, id);
114         ResultSet resultSet = statement.executeQuery();
115         if(resultSet.next()){
116             country = new Country(resultSet.getString(1),
117                                   resultSet.getString(2),
118                                   resultSet.getInt(3));
119         }
120     } catch (Exception e) {
121         e.printStackTrace();
122     }
123     return country;
124 }
125
126
127
128

OJDBC.java
85 // Update Country
86 Country country = new Country("ZW", "Zaman WW3");
87 System.out.println(icdao.update("ZW", country));
88
89 // delete country
90 icdao.delete("ZW");
91
92 // getById
93 System.out.println(
94     icdao.getById("ZW").getId() + " " +
95     icdao.getById("ZW").getName() + " " +
96     icdao.getById("ZW").getRegion_id()
97 );
98
99
100
101

Output - OJDBC (run)
run:
tools.DBConnection@7852e922
ZW Zambia 4
BUILD SUCCESSFUL (total time: 4 seconds)
```

6. Search countries



The screenshot shows an IDE with two open files: `CountryDAO.java` and `OJDBC.java`. The `CountryDAO.java` file contains the implementation of the `search` method, which takes a keyword as a parameter and returns a list of `Country` objects. The `OJDBC.java` file shows the usage of the `search` method, including deleting a country, retrieving a country by ID, and searching for countries by keyword.

```
CountryDAO.java
129 }
130
131 @Override
132 public List<Country> search(String keyword) {
133     List<Country> listCountry = new ArrayList<>();
134     String query = "SELECT * FROM HR.COUNTRIES WHERE country_id LIKE ? OR country_name LIKE ?";
135     PreparedStatement pStatement;
136     try {
137         pStatement = connection.prepareStatement(query);
138         pStatement.setString(1, "%" + keyword + "%");
139         pStatement.setString(2, "%" + keyword + "%");
140     } catch (SQLException ex) {
141         Logger.getLogger(CountryDAO.class.getName()).log(Level.SEVERE, null, ex);
142     }
143     while(resultSet.next()){
144         listCountry.add(new Country(resultSet.getString(1), resultSet.getString(2), resultSet.getInt(3)));
145     }
146     return listCountry;
147 }
148
149
150
151
152
153
154

OJDBC.java
90 // delete country
91 icdao.delete("ZW");
92
93 // getById
94 System.out.println(
95     icdao.getById("ZW").getId() + " " +
96     icdao.getById("ZW").getName() + " " +
97     icdao.getById("ZW").getRegion_id()
98 );
99
100 // Search countries
101 for(Country country : icdao.search("ZW")){
102     System.out.println(country.getId() + " " +
103     country.getName() + " " +
104     country.getRegion_id()
105 );
106 }
107
108 for(Country country : icdao.search("Zambia")){
109     System.out.println(country.getId() + " " +
110     country.getName() + " " +
111     country.getRegion_id()
112 );
113 }
114
115 }
116

Output - OJDBC (run)
run:
tools.DBConnection@7852e922
ZW Zambia 4
ZW Zambia 4
BUILD SUCCESSFUL (total time: 3 seconds)
```