```xml
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns="http://java.sun.com/xml/ns/j2ee" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://java.sun.com/xml/ns/j2ee http://java.sun.com/xml/ns/j2ee/web-app_2_4.xsd"
    version="2.4">

    <display-name>IBM_emp_0201MVC_Single-Table</display-name>

    <welcome-file-list>
        <welcome-file>index.html</welcome-file>
        <welcome-file>index.htm</welcome-file>
        <welcome-file>index.jsp</welcome-file>
        <welcome-file>default.html</welcome-file>
        <welcome-file>default.htm</welcome-file>
        <welcome-file>default.jsp</welcome-file>
    </welcome-file-list>

   <resource-ref>
        <description>DB Connection</description>
        <res-ref-name>jdbc/TestDB</res-ref-name>
        <res-type>javax.sql.DataSource</res-type>
        <res-auth>Container</res-auth>
    </resource-ref>

    <servlet>
        <servlet-name>EmpServlet</servlet-name>
        <servlet-class>com.emp.controller.EmpServlet</servlet-class>
    </servlet>
    <servlet-mapping>
        <servlet-name>EmpServlet</servlet-name>
        <url-pattern>/emp/emp.do</url-pattern>
    </servlet-mapping>

</web-app>

```

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
  Licensed to the Apache Software Foundation (ASF) under one or more
  contributor license agreements.  See the NOTICE file distributed with
  this work for additional information regarding copyright ownership.
  The ASF licenses this file to You under the Apache License, Version 2.0
  (the "License"); you may not use this file except in compliance with
  the License.  You may obtain a copy of the License at

      http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software
  distributed under the License is distributed on an "AS IS" BASIS,
  WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
  See the License for the specific language governing permissions and
  limitations under the License.
--><!-- Note:  A "Server" is not itself a "Container", so you may not
     define subcomponents such as "Valves" at this level.
     Documentation at /docs/config/server.html
 --><Server port="8005" shutdown="SHUTDOWN">
  <Listener className="org.apache.catalina.startup.VersionLoggerListener"/>
  <!-- Security listener. Documentation at /docs/config/listeners.html
  <Listener className="org.apache.catalina.security.SecurityListener" />
  -->
  <!--APR library loader. Documentation at /docs/apr.html -->
  <Listener SSLEngine="on" className="org.apache.catalina.core.AprLifecycleListener"/>
  <!--Initialize Jasper prior to webapps are loaded. Documentation at /docs/jasper-howto.html -->
  <Listener className="org.apache.catalina.core.JasperListener"/>
  <!-- Prevent memory leaks due to use of particular java/javax APIs-->
  <Listener
```

```
className="org.apache.catalina.core.JreMemoryLeakPreventionListener
"/>
  <Listener
className="org.apache.catalina.mbeans.GlobalResourcesLifecycleListe
ner"/>
  <Listener
className="org.apache.catalina.core.ThreadLocalLeakPreventionListen
er"/>

  <!-- Global JNDI resources
       Documentation at /docs/jndi-resources-howto.html
  -->
  <GlobalNamingResources>
    <!-- Editable user database that can also be used by
         UserDatabaseRealm to authenticate users
    -->
    <Resource auth="Container" description="User database that can
be updated and saved"
factory="org.apache.catalina.users.MemoryUserDatabaseFactory"
name="UserDatabase" pathname="conf/tomcat-users.xml"
type="org.apache.catalina.UserDatabase"/>
  </GlobalNamingResources>

  <!-- A "Service" is a collection of one or more "Connectors" that
share
       a single "Container" Note:  A "Service" is not itself a
"Container",
       so you may not define subcomponents such as "Valves" at this
level.
       Documentation at /docs/config/service.html
   -->
  <Service name="Catalina">

    <!--The connectors can use a shared executor, you can define
one or more named thread pools-->
    <!--
    <Executor name="tomcatThreadPool" namePrefix="catalina-exec-"
        maxThreads="150" minSpareThreads="4"/>
    -->


    <!-- A "Connector" represents an endpoint by which requests are
received
         and responses are returned. Documentation at :
         Java HTTP Connector: /docs/config/http.html (blocking &
```

```
  non-blocking)
61          Java AJP  Connector: /docs/config/ajp.html
62          APR (HTTP/AJP) Connector: /docs/apr.html
63          Define a non-SSL HTTP/1.1 Connector on port 8080
64      -->
65      <Connector connectionTimeout="20000" port="8081"
  protocol="HTTP/1.1" redirectPort="8443"/>
66      <!-- A "Connector" using the shared thread pool-->
67      <!--
68      <Connector executor="tomcatThreadPool"
69                 port="8080" protocol="HTTP/1.1"
70                 connectionTimeout="20000"
71                 redirectPort="8443" />
72      -->
73      <!-- Define a SSL HTTP/1.1 Connector on port 8443
74          This connector uses the BIO implementation that requires
  the JSSE
75          style configuration. When using the APR/native
  implementation, the
76          OpenSSL style configuration is required as described in
  the APR/native
77          documentation -->
78      <!--
79      <Connector port="8443"
  protocol="org.apache.coyote.http11.Http11Protocol"
80                 maxThreads="150" SSLEnabled="true" scheme="https"
  secure="true"
81                 clientAuth="false" sslProtocol="TLS" />
82      -->
83
84      <!-- Define an AJP 1.3 Connector on port 8009 -->
85      <Connector port="8009" protocol="AJP/1.3" redirectPort="8443"/>
86
87
88      <!-- An Engine represents the entry point (within Catalina)
  that processes
89          every request.  The Engine implementation for Tomcat stand
  alone
90          analyzes the HTTP headers included with the request, and
  passes them
91          on to the appropriate Host (virtual host).
92          Documentation at /docs/config/engine.html -->
93
94      <!-- You should set jvmRoute to support load-balancing via AJP
  ie :
```

```
 95    <Engine name="Catalina" defaultHost="localhost"
   jvmRoute="jvm1">
 96    -->
 97    <Engine defaultHost="localhost" name="Catalina">
 98
 99      <!--For clustering, please take a look at documentation at:
100          /docs/cluster-howto.html  (simple how to)
101          /docs/config/cluster.html (reference documentation) -->
102      <!--
103      <Cluster
   className="org.apache.catalina.ha.tcp.SimpleTcpCluster"/>
104      -->
105
106      <!-- Use the LockOutRealm to prevent attempts to guess user
   passwords
107           via a brute-force attack -->
108      <Realm className="org.apache.catalina.realm.LockOutRealm">
109        <!-- This Realm uses the UserDatabase configured in the
   global JNDI
110             resources under the key "UserDatabase".  Any edits
111             that are performed against this UserDatabase are
   immediately
112             available for use by the Realm.  -->
113        <Realm
   className="org.apache.catalina.realm.UserDatabaseRealm"
   resourceName="UserDatabase"/>
114      </Realm>
115
116      <Host appBase="webapps" autoDeploy="true" name="localhost"
   unpackWARs="true">
117
118        <!-- SingleSignOn valve, share authentication between web
   applications
119             Documentation at: /docs/config/valve.html -->
120        <!--
121        <Valve
   className="org.apache.catalina.authenticator.SingleSignOn" />
122        -->
123
124        <!-- Access log processes all example.
125             Documentation at: /docs/config/valve.html
126             Note: The pattern used is equivalent to using
   pattern="common" -->
127        <Valve
   className="org.apache.catalina.valves.AccessLogValve"
```

```
    directory="logs" pattern="%h %l %u %t &quot;%r&quot; %s %b"
    prefix="localhost_access_log." suffix=".txt"/>
128
129       <Context docBase="SL314" path="/SL314" reloadable="true"
    source="org.eclipse.jst.j2ee.server:SL314"/><Context
    docBase="WebApp_ch04_FileUpload" path="/WebApp_ch04_FileUpload"
    reloadable="true"
    source="org.eclipse.jst.jee.server:WebApp_ch04_FileUpload"/
    ><Context docBase="IBM_MVC" path="/IBM_MVC" reloadable="true"
    source="org.eclipse.jst.j2ee.server:IBM_MVC"/><Context
    docBase="IBM_emp_01MVC_Basic" path="/IBM_emp_01MVC_Basic"
    reloadable="true"
    source="org.eclipse.jst.j2ee.server:IBM_emp_01MVC_Basic"/><Context
    docBase="IBM_emp_0201MVC_Single-Table" path="/
    IBM_emp_0201MVC_Single-Table" reloadable="true"
    source="org.eclipse.jst.j2ee.server:IBM_emp_0201MVC_Single-Table"/
    ><Context docBase="BA103G5_Jacky" path="/BA103G5_Jacky"
    reloadable="true"
    source="org.eclipse.jst.jee.server:BA103G5_Jacky"/></Host>
130     </Engine>
131   </Service>
132 </Server>
```

```java
/** 自行取得DataSource的 servlet
import java.io.*;

public class Test_DataSource extends HttpServlet {

    public void doGet(HttpServletRequest req, HttpServletResponse res)
            throws ServletException, IOException {

        res.setContentType("text/plain; charset=Big5");
        PrintWriter out = res.getWriter();

        try {
            Context ctx = new javax.naming.InitialContext();
            DataSource ds = (DataSource) ctx.lookup("java:comp/env/jdbc/TestDB");
            if (ds != null) {
                Connection conn = ds.getConnection();

                if (conn != null) {
                    out.println("Got Connection: " + conn.toString());
                    Statement stmt = conn.createStatement();
                    ResultSet rs = stmt.executeQuery("select * from emp2");
                    while (rs.next()) {
                        out.println("empNo = " + rs.getString(1));
                    }
                    conn.close();
                }
            }
        } catch (NamingException e) {
            e.printStackTrace();
        } catch (SQLException e) {
            e.printStackTrace();
        } catch (Exception e) {
            e.printStackTrace();
        }

    }
}
```

DeptDAO_interface.java

```java
package com.dept.model;

import java.util.*;

public interface DeptDAO_interface {
          public void insert(DeptVO deptVO);
          public void update(DeptVO deptVO);
          public void delete(Integer deptno);
          public DeptVO findByPrimaryKey(Integer deptno);
          public List<DeptVO> getAll();
          //查詢某部門的員工(一對多)(回傳 Set)
          public Set<EmpVO> getEmpsByDeptno(Integer deptno);
}

```

```java
package com.dept.model;

import java.util.List;

public class DeptService {

    private DeptDAO_interface dao;

    public DeptService() {
        dao = new DeptDAO();
    }

    public List<DeptVO> getAll() {
        return dao.getAll();
    }

    public DeptVO getOneDept(Integer deptno) {
        return dao.findByPrimaryKey(deptno);
    }

    public Set<EmpVO> getEmpsByDeptno(Integer deptno) {
        return dao.getEmpsByDeptno(deptno);
    }

    public void deleteDept(Integer deptno) {
        dao.delete(deptno);
    }
}

```

```java
package com.dept.model;

import java.util.*;

public class DeptDAO implements DeptDAO_interface {

    // 一個應用程式中,針對一個資料庫 ,共用一個DataSource即可
    private static DataSource ds = null;
    static {
        try {
            Context ctx = new InitialContext();
            ds = (DataSource) ctx.lookup("java:comp/env/jdbc/TestDB");
        } catch (NamingException e) {
            e.printStackTrace();
        }
    }

    private static final String INSERT_STMT = "INSERT INTO dept2 (deptno,dname,loc) VALUES (dept2_seq.NEXTVAL, ?, ?)";
    private static final String GET_ALL_STMT = "SELECT deptno , dname, loc FROM dept2";
    private static final String GET_ONE_STMT = "SELECT deptno , dname, loc FROM dept2 where deptno = ?";
    private static final String GET_Emps_ByDeptno_STMT = "SELECT empno,ename,job,to_char(hiredate,'yyyy-mm-dd') hiredate,sal,comm,deptno FROM emp2 where deptno = ? order by empno";

    private static final String DELETE_EMPs = "DELETE FROM emp2 where deptno = ?";
    private static final String DELETE_DEPT = "DELETE FROM dept2 where deptno = ?";

    private static final String UPDATE = "UPDATE dept2 set dname=?, loc=? where deptno = ?";

    @Override
    public void insert(DeptVO deptVO) {

        Connection con = null;
        PreparedStatement pstmt = null;

        try {

```

```
44              con = ds.getConnection();
45              pstmt = con.prepareStatement(INSERT_STMT);
46
47              pstmt.setString(1, deptVO.getDname());
48              pstmt.setString(2, deptVO.getLoc());
49
50              pstmt.executeUpdate();
51
52              // Handle any SQL errors
53          } catch (SQLException se) {
54              throw new RuntimeException("A database error occured. "
55                      + se.getMessage());
56              // Clean up JDBC resources
57          } finally {
58              if (pstmt != null) {
59                  try {
60                      pstmt.close();
61                  } catch (SQLException se) {
62                      se.printStackTrace(System.err);
63                  }
64              }
65              if (con != null) {
66                  try {
67                      con.close();
68                  } catch (Exception e) {
69                      e.printStackTrace(System.err);
70                  }
71              }
72          }
73
74      }
75
76      @Override
77      public void update(DeptVO deptVO) {
78
79          Connection con = null;
80          PreparedStatement pstmt = null;
81
82          try {
83
84              con = ds.getConnection();
85              pstmt = con.prepareStatement(UPDATE);
86
87              pstmt.setString(1, deptVO.getDname());
88              pstmt.setString(2, deptVO.getLoc());
```

```
 89             pstmt.setInt(3, deptVO.getDeptno());
 90
 91             pstmt.executeUpdate();
 92
 93             // Handle any SQL errors
 94         } catch (SQLException se) {
 95             throw new RuntimeException("A database error occured. "
 96                     + se.getMessage());
 97             // Clean up JDBC resources
 98         } finally {
 99             if (pstmt != null) {
100                 try {
101                     pstmt.close();
102                 } catch (SQLException se) {
103                     se.printStackTrace(System.err);
104                 }
105             }
106             if (con != null) {
107                 try {
108                     con.close();
109                 } catch (Exception e) {
110                     e.printStackTrace(System.err);
111                 }
112             }
113         }
114
115     }
116
117     @Override
118     public void delete(Integer deptno) {
119         int updateCount_EMPs = 0;
120
121         Connection con = null;
122         PreparedStatement pstmt = null;
123
124         try {
125
126             con = ds.getConnection();
127
128             // 1●設定於 pstm.executeUpdate()之前
129             con.setAutoCommit(false);
130
131             // 先刪除員工
132             pstmt = con.prepareStatement(DELETE_EMPs);
133             pstmt.setInt(1, deptno);
```

```
134             updateCount_EMPs = pstmt.executeUpdate();
135             // 再刪除部門
136             pstmt = con.prepareStatement(DELETE_DEPT);
137             pstmt.setInt(1, deptno);
138             pstmt.executeUpdate();
139 
140             // 2•設定於 pstm.executeUpdate()之後
141             con.commit();
142             con.setAutoCommit(true);
143             System.out.println("刪除部門編號" + deptno + "時,共有員工" +
    updateCount_EMPs
144                     + "人同時被刪除");
145 
146             // Handle any SQL errors
147         } catch (SQLException se) {
148             if (con != null) {
149                 try {
150                     // 3•設定於當有exception發生時之catch區塊內
151                     con.rollback();
152                 } catch (SQLException excep) {
153                     throw new RuntimeException("rollback error
    occured. "
154                             + excep.getMessage());
155                 }
156             }
157             throw new RuntimeException("A database error occured. "
158                     + se.getMessage());
159         } finally {
160             if (pstmt != null) {
161                 try {
162                     pstmt.close();
163                 } catch (SQLException se) {
164                     se.printStackTrace(System.err);
165                 }
166             }
167             if (con != null) {
168                 try {
169                     con.close();
170                 } catch (Exception e) {
171                     e.printStackTrace(System.err);
172                 }
173             }
174         }
175 
176     }
```

```
177
178     @Override
179     public DeptVO findByPrimaryKey(Integer deptno) {
180
181         DeptVO deptVO = null;
182         Connection con = null;
183         PreparedStatement pstmt = null;
184         ResultSet rs = null;
185
186         try {
187
188             con = ds.getConnection();
189             pstmt = con.prepareStatement(GET_ONE_STMT);
190
191             pstmt.setInt(1, deptno);
192
193             rs = pstmt.executeQuery();
194
195             while (rs.next()) {
196                 // deptVO 也稱為 Domain objects
197                 deptVO = new DeptVO();
198                 deptVO.setDeptno(rs.getInt("deptno"));
199                 deptVO.setDname(rs.getString("dname"));
200                 deptVO.setLoc(rs.getString("loc"));
201             }
202
203             // Handle any SQL errors
204         } catch (SQLException se) {
205             throw new RuntimeException("A database error occured. "
206                     + se.getMessage());
207             // Clean up JDBC resources
208         } finally {
209             if (rs != null) {
210                 try {
211                     rs.close();
212                 } catch (SQLException se) {
213                     se.printStackTrace(System.err);
214                 }
215             }
216             if (pstmt != null) {
217                 try {
218                     pstmt.close();
219                 } catch (SQLException se) {
220                     se.printStackTrace(System.err);
221                 }
```

```
222             }
223             if (con != null) {
224                 try {
225                     con.close();
226                 } catch (Exception e) {
227                     e.printStackTrace(System.err);
228                 }
229             }
230         }
231         return deptVO;
232     }
233 
234     @Override
235     public List<DeptVO> getAll() {
236         List<DeptVO> list = new ArrayList<DeptVO>();
237         DeptVO deptVO = null;
238 
239         Connection con = null;
240         PreparedStatement pstmt = null;
241         ResultSet rs = null;
242 
243         try {
244 
245             con = ds.getConnection();
246             pstmt = con.prepareStatement(GET_ALL_STMT);
247             rs = pstmt.executeQuery();
248 
249             while (rs.next()) {
250                 deptVO = new DeptVO();
251                 deptVO.setDeptno(rs.getInt("deptno"));
252                 deptVO.setDname(rs.getString("dname"));
253                 deptVO.setLoc(rs.getString("loc"));
254                 list.add(deptVO); // Store the row in the list
255             }
256 
257             // Handle any SQL errors
258         } catch (SQLException se) {
259             throw new RuntimeException("A database error occured. "
260                     + se.getMessage());
261         } finally {
262             if (rs != null) {
263                 try {
264                     rs.close();
265                 } catch (SQLException se) {
266                     se.printStackTrace(System.err);
```

```
267                 }
268             }
269             if (pstmt != null) {
270                 try {
271                     pstmt.close();
272                 } catch (SQLException se) {
273                     se.printStackTrace(System.err);
274                 }
275             }
276             if (con != null) {
277                 try {
278                     con.close();
279                 } catch (Exception e) {
280                     e.printStackTrace(System.err);
281                 }
282             }
283         }
284         return list;
285     }
286 
287     @Override
288     public Set<EmpVO> getEmpsByDeptno(Integer deptno) {
289         Set<EmpVO> set = new LinkedHashSet<EmpVO>();
290         EmpVO empVO = null;
291 
292         Connection con = null;
293         PreparedStatement pstmt = null;
294         ResultSet rs = null;
295 
296         try {
297 
298             con = ds.getConnection();
299             pstmt = con.prepareStatement(GET_Emps_ByDeptno_STMT);
300             pstmt.setInt(1, deptno);
301             rs = pstmt.executeQuery();
302 
303             while (rs.next()) {
304                 empVO = new EmpVO();
305                 empVO.setEmpno(rs.getInt("empno"));
306                 empVO.setEname(rs.getString("ename"));
307                 empVO.setJob(rs.getString("job"));
308                 empVO.setHiredate(rs.getDate("hiredate"));
309                 empVO.setSal(rs.getDouble("sal"));
310                 empVO.setComm(rs.getDouble("comm"));
311                 empVO.setDeptno(rs.getInt("deptno"));
```

```
312                 set.add(empVO); // Store the row in the vector
313             }
314 
315             // Handle any SQL errors
316         } catch (SQLException se) {
317             throw new RuntimeException("A database error occured. "
318                     + se.getMessage());
319         } finally {
320             if (rs != null) {
321                 try {
322                     rs.close();
323                 } catch (SQLException se) {
324                     se.printStackTrace(System.err);
325                 }
326             }
327             if (pstmt != null) {
328                 try {
329                     pstmt.close();
330                 } catch (SQLException se) {
331                     se.printStackTrace(System.err);
332                 }
333             }
334             if (con != null) {
335                 try {
336                     con.close();
337                 } catch (Exception e) {
338                     e.printStackTrace(System.err);
339                 }
340             }
341         }
342         return set;
343     }
344 }
```

```java
package com.dept.model;

import java.util.*;

public class DeptJDBCDAO implements DeptDAO_interface {
    String driver = "oracle.jdbc.driver.OracleDriver";
    String url = "jdbc:oracle:thin:@localhost:1521:XE";
    String userid = "hr";
    String passwd = "123456";

    private static final String INSERT_STMT = "INSERT INTO dept2 (deptno,dname,loc) VALUES (dept2_seq.NEXTVAL, ?, ?)";
    private static final String GET_ALL_STMT = "SELECT deptno , dname, loc FROM dept2";
    private static final String GET_ONE_STMT = "SELECT deptno , dname, loc FROM dept2 where deptno = ?";
    private static final String GET_Emps_ByDeptno_STMT = "SELECT empno,ename,job,to_char(hiredate,'yyyy-mm-dd') hiredate,sal,comm,deptno FROM emp2 where deptno = ? order by empno";

    private static final String DELETE_EMPs = "DELETE FROM emp2 where deptno = ?";
    private static final String DELETE_DEPT = "DELETE FROM dept2 where deptno = ?";

    private static final String UPDATE = "UPDATE dept2 set dname=?, loc=? where deptno = ?";

    @Override
    public void insert(DeptVO deptVO) {

        Connection con = null;
        PreparedStatement pstmt = null;

        try {

            Class.forName(driver);
            con = DriverManager.getConnection(url, userid, passwd);
            pstmt = con.prepareStatement(INSERT_STMT);

            pstmt.setString(1, deptVO.getDname());
            pstmt.setString(2, deptVO.getLoc());

            pstmt.executeUpdate();
```

```
40
41              // Handle any driver errors
42          } catch (ClassNotFoundException e) {
43              throw new RuntimeException("Couldn't load database
  driver. "
44                      + e.getMessage());
45              // Handle any SQL errors
46          } catch (SQLException se) {
47              throw new RuntimeException("A database error occured. "
48                      + se.getMessage());
49              // Clean up JDBC resources
50          } finally {
51              if (pstmt != null) {
52                  try {
53                      pstmt.close();
54                  } catch (SQLException se) {
55                      se.printStackTrace(System.err);
56                  }
57              }
58              if (con != null) {
59                  try {
60                      con.close();
61                  } catch (Exception e) {
62                      e.printStackTrace(System.err);
63                  }
64              }
65          }
66
67      }
68
69      @Override
70      public void update(DeptVO deptVO) {
71
72          Connection con = null;
73          PreparedStatement pstmt = null;
74
75          try {
76
77              Class.forName(driver);
78              con = DriverManager.getConnection(url, userid, passwd);
79              pstmt = con.prepareStatement(UPDATE);
80
81              pstmt.setString(1, deptVO.getDname());
82              pstmt.setString(2, deptVO.getLoc());
83              pstmt.setInt(3, deptVO.getDeptno());
```

```
 84
 85              pstmt.executeUpdate();
 86
 87              // Handle any driver errors
 88          } catch (ClassNotFoundException e) {
 89              throw new RuntimeException("Couldn't load database
    driver. "
 90                      + e.getMessage());
 91              // Handle any SQL errors
 92          } catch (SQLException se) {
 93              throw new RuntimeException("A database error occured. "
 94                      + se.getMessage());
 95              // Clean up JDBC resources
 96          } finally {
 97              if (pstmt != null) {
 98                  try {
 99                      pstmt.close();
100                  } catch (SQLException se) {
101                      se.printStackTrace(System.err);
102                  }
103              }
104              if (con != null) {
105                  try {
106                      con.close();
107                  } catch (Exception e) {
108                      e.printStackTrace(System.err);
109                  }
110              }
111          }
112
113      }
114
115      @Override
116      public void delete(Integer deptno) {
117          int updateCount_EMPs = 0;
118
119          Connection con = null;
120          PreparedStatement pstmt = null;
121
122          try {
123
124              Class.forName(driver);
125              con = DriverManager.getConnection(url, userid, passwd);
126
127              // 1●設定於 pstm.executeUpdate()之前
```

```
128             con.setAutoCommit(false);
129
130             // 先刪除員工
131             pstmt = con.prepareStatement(DELETE_EMPs);
132             pstmt.setInt(1, deptno);
133             updateCount_EMPs = pstmt.executeUpdate();
134             // 再刪除部門
135             pstmt = con.prepareStatement(DELETE_DEPT);
136             pstmt.setInt(1, deptno);
137             pstmt.executeUpdate();
138
139             // 2•設定於 pstm.executeUpdate()之後
140             con.commit();
141             con.setAutoCommit(true);
142             System.out.println("刪除部門編號" + deptno + "時,共有員工" +
    updateCount_EMPs
143                     + "人同時被刪除");
144
145             // Handle any driver errors
146         } catch (ClassNotFoundException e) {
147             throw new RuntimeException("Couldn't load database
    driver. "
148                     + e.getMessage());
149             // Handle any SQL errors
150         } catch (SQLException se) {
151             if (con != null) {
152                 try {
153                     // 3•設定於當有exception發生時之catch區塊內
154                     con.rollback();
155                 } catch (SQLException excep) {
156                     throw new RuntimeException("rollback error
    occured. "
157                             + excep.getMessage());
158                 }
159             }
160             throw new RuntimeException("A database error occured. "
161                     + se.getMessage());
162         } finally {
163             if (pstmt != null) {
164                 try {
165                     pstmt.close();
166                 } catch (SQLException se) {
167                     se.printStackTrace(System.err);
168                 }
169             }
```

```
170             if (con != null) {
171                 try {
172                     con.close();
173                 } catch (Exception e) {
174                     e.printStackTrace(System.err);
175                 }
176             }
177         }
178 
179     }
180 
181     @Override
182     public DeptVO findByPrimaryKey(Integer deptno) {
183 
184         DeptVO deptVO = null;
185         Connection con = null;
186         PreparedStatement pstmt = null;
187         ResultSet rs = null;
188 
189         try {
190 
191             Class.forName(driver);
192             con = DriverManager.getConnection(url, userid, passwd);
193             pstmt = con.prepareStatement(GET_ONE_STMT);
194 
195             pstmt.setInt(1, deptno);
196 
197             rs = pstmt.executeQuery();
198 
199             while (rs.next()) {
200                 // deptVO 也稱為 Domain objects
201                 deptVO = new DeptVO();
202                 deptVO.setDeptno(rs.getInt("deptno"));
203                 deptVO.setDname(rs.getString("dname"));
204                 deptVO.setLoc(rs.getString("loc"));
205             }
206 
207             // Handle any driver errors
208         } catch (ClassNotFoundException e) {
209             throw new RuntimeException("Couldn't load database
    driver. "
210                     + e.getMessage());
211             // Handle any SQL errors
212         } catch (SQLException se) {
213             throw new RuntimeException("A database error occured. "
```

```
214                     + se.getMessage());
215             // Clean up JDBC resources
216         } finally {
217             if (rs != null) {
218                 try {
219                     rs.close();
220                 } catch (SQLException se) {
221                     se.printStackTrace(System.err);
222                 }
223             }
224             if (pstmt != null) {
225                 try {
226                     pstmt.close();
227                 } catch (SQLException se) {
228                     se.printStackTrace(System.err);
229                 }
230             }
231             if (con != null) {
232                 try {
233                     con.close();
234                 } catch (Exception e) {
235                     e.printStackTrace(System.err);
236                 }
237             }
238         }
239         return deptVO;
240     }
241 
242     @Override
243     public List<DeptVO> getAll() {
244         List<DeptVO> list = new ArrayList<DeptVO>();
245         DeptVO deptVO = null;
246 
247         Connection con = null;
248         PreparedStatement pstmt = null;
249         ResultSet rs = null;
250 
251         try {
252 
253             Class.forName(driver);
254             con = DriverManager.getConnection(url, userid, passwd);
255             pstmt = con.prepareStatement(GET_ALL_STMT);
256             rs = pstmt.executeQuery();
257 
258             while (rs.next()) {
```

```
259                 deptVO = new DeptVO();
260                 deptVO.setDeptno(rs.getInt("deptno"));
261                 deptVO.setDname(rs.getString("dname"));
262                 deptVO.setLoc(rs.getString("loc"));
263                 list.add(deptVO); // Store the row in the list
264             }
265 
266             // Handle any driver errors
267         } catch (ClassNotFoundException e) {
268             throw new RuntimeException("Couldn't load database
    driver. "
269                     + e.getMessage());
270             // Handle any SQL errors
271         } catch (SQLException se) {
272             throw new RuntimeException("A database error occured. "
273                     + se.getMessage());
274         } finally {
275             if (rs != null) {
276                 try {
277                     rs.close();
278                 } catch (SQLException se) {
279                     se.printStackTrace(System.err);
280                 }
281             }
282             if (pstmt != null) {
283                 try {
284                     pstmt.close();
285                 } catch (SQLException se) {
286                     se.printStackTrace(System.err);
287                 }
288             }
289             if (con != null) {
290                 try {
291                     con.close();
292                 } catch (Exception e) {
293                     e.printStackTrace(System.err);
294                 }
295             }
296         }
297         return list;
298     }
299 
300     @Override
301     public Set<EmpVO> getEmpsByDeptno(Integer deptno) {
302         Set<EmpVO> set = new LinkedHashSet<EmpVO>();
```

```
303         EmpVO empVO = null;
304
305         Connection con = null;
306         PreparedStatement pstmt = null;
307         ResultSet rs = null;
308
309         try {
310
311             Class.forName(driver);
312             con = DriverManager.getConnection(url, userid, passwd);
313             pstmt = con.prepareStatement(GET_Emps_ByDeptno_STMT);
314             pstmt.setInt(1, deptno);
315             rs = pstmt.executeQuery();
316
317             while (rs.next()) {
318                 empVO = new EmpVO();
319                 empVO.setEmpno(rs.getInt("empno"));
320                 empVO.setEname(rs.getString("ename"));
321                 empVO.setJob(rs.getString("job"));
322                 empVO.setHiredate(rs.getDate("hiredate"));
323                 empVO.setSal(rs.getDouble("sal"));
324                 empVO.setComm(rs.getDouble("comm"));
325                 empVO.setDeptno(rs.getInt("deptno"));
326                 set.add(empVO); // Store the row in the vector
327             }
328
329             // Handle any driver errors
330         } catch (ClassNotFoundException e) {
331             throw new RuntimeException("Couldn't load database
    driver. "
332                     + e.getMessage());
333             // Handle any SQL errors
334         } catch (SQLException se) {
335             throw new RuntimeException("A database error occured. "
336                     + se.getMessage());
337         } finally {
338             if (rs != null) {
339                 try {
340                     rs.close();
341                 } catch (SQLException se) {
342                     se.printStackTrace(System.err);
343                 }
344             }
345             if (pstmt != null) {
346                 try {
```

```
347                     pstmt.close();
348                 } catch (SQLException se) {
349                     se.printStackTrace(System.err);
350                 }
351             }
352             if (con != null) {
353                 try {
354                     con.close();
355                 } catch (Exception e) {
356                     e.printStackTrace(System.err);
357                 }
358             }
359         }
360         return set;
361     }
362 
363     public static void main(String[] args) {
364 
365         DeptJDBCDAO dao = new DeptJDBCDAO();
366 
367         // 新增
368 //      DeptVO deptVO1 = new DeptVO();
369 //      deptVO1.setDname("製造部");
370 //      deptVO1.setLoc("中國江西");
371 //      dao.insert(deptVO1);
372 
373         // 修改
374 //      DeptVO deptVO2 = new DeptVO();
375 //      deptVO2.setDeptno(10);
376 //      deptVO2.setDname("財務部2");
377 //      deptVO2.setLoc("臺灣台北2");
378 //      dao.update(deptVO2);
379 
380         // 刪除
381 //      dao.delete(30);
382 
383         // 查詢
384 //      DeptVO deptVO3 = dao.findByPrimaryKey(10);
385 //      System.out.print(deptVO3.getDeptno() + ",");
386 //      System.out.print(deptVO3.getDname() + ",");
387 //      System.out.println(deptVO3.getLoc());
388 //      System.out.println("---------------------");
389 
390         // 查詢部門
391         List<DeptVO> list = dao.getAll();
```

```
392         for (DeptVO aDept : list) {
393             System.out.print(aDept.getDeptno() + ",");
394             System.out.print(aDept.getDname() + ",");
395             System.out.print(aDept.getLoc());
396             System.out.println();
397         }
398 
399         // 查詢某部門的員工
400         Set<EmpVO> set = dao.getEmpsByDeptno(10);
401         for (EmpVO aEmp : set) {
402             System.out.print(aEmp.getEmpno() + ",");
403             System.out.print(aEmp.getEname() + ",");
404             System.out.print(aEmp.getJob() + ",");
405             System.out.print(aEmp.getHiredate() + ",");
406             System.out.print(aEmp.getSal() + ",");
407             System.out.print(aEmp.getComm() + ",");
408             System.out.print(aEmp.getDeptno());
409             System.out.println();
410         }
411     }
412 }
```

```java
package com.dept.model;

public class DeptVO  implements java.io.Serializable{
    private Integer deptno;
    private String dname;
    private String loc;

    public Integer getDeptno() {
        return deptno;
    }
    public void setDeptno(Integer deptno) {
        this.deptno = deptno;
    }
    public String getDname() {
        return dname;
    }
    public void setDname(String dname) {
        this.dname = dname;
    }
    public String getLoc() {
        return loc;
    }
    public void setLoc(String loc) {
        this.loc = loc;
    }
}

```

```java
package com.emp.controller;

import java.io.*;

public class EmpServlet extends HttpServlet {

    public void doGet(HttpServletRequest req, HttpServletResponse res)
            throws ServletException, IOException {
        doPost(req, res);
    }

    public void doPost(HttpServletRequest req, HttpServletResponse res)
            throws ServletException, IOException {

        req.setCharacterEncoding("UTF-8");
        String action = req.getParameter("action");


        if ("getOne_For_Display".equals(action)) { // 來自select_page.jsp的請求

            List<String> errorMsgs = new LinkedList<String>();
            // Store this set in the request scope, in case we need to
            // send the ErrorPage view.
            req.setAttribute("errorMsgs", errorMsgs);

            try {
                /***************************1.接收請求參數 - 輸入格式的錯誤處理**********************/
                String str = req.getParameter("empno");
                if (str == null || (str.trim()).length() == 0) {
                    errorMsgs.add("請輸入員工編號");
                }
                // Send the use back to the form, if there were errors
                if (!errorMsgs.isEmpty()) {
                    RequestDispatcher failureView = req
                            .getRequestDispatcher("/emp/select_page.jsp");
                    failureView.forward(req, res);
                    return;//程式中斷
                }
```

```
45
46                  Integer empno = null;
47                  try {
48                      empno = new Integer(str);
49                  } catch (Exception e) {
50                      errorMsgs.add("員工編號格式不正確");
51                  }
52                  // Send the use back to the form, if there were
  errors
53                  if (!errorMsgs.isEmpty()) {
54                      RequestDispatcher failureView = req
55                              .getRequestDispatcher("/emp/
  select_page.jsp");
56                      failureView.forward(req, res);
57                      return;//程式中斷
58                  }
59
60                  /***************************2.開始查詢資料
  *****************************************/
61                  EmpService empSvc = new EmpService();
62                  EmpVO empVO = empSvc.getOneEmp(empno);
63                  if (empVO == null) {
64                      errorMsgs.add("查無資料");
65                  }
66                  // Send the use back to the form, if there were
  errors
67                  if (!errorMsgs.isEmpty()) {
68                      RequestDispatcher failureView = req
69                              .getRequestDispatcher("/emp/
  select_page.jsp");
70                      failureView.forward(req, res);
71                      return;//程式中斷
72                  }
73
74                  /***************************3.查詢完成,準備轉交(Send
  the Success view)*************/
75                  req.setAttribute("empVO", empVO); // 資料庫取出的empVO
  物件,存入req
76                  String url = "/emp/listOneEmp.jsp";
77                  RequestDispatcher successView =
  req.getRequestDispatcher(url); // 成功轉交 listOneEmp.jsp
78                  successView.forward(req, res);
79
80                  /***************************其他可能的錯誤處理
  **********************************/
```

```
81              } catch (Exception e) {
82                  errorMsgs.add("無法取得資料:" + e.getMessage());
83                  RequestDispatcher failureView = req
84                          .getRequestDispatcher("/emp/
   select_page.jsp");
85                  failureView.forward(req, res);
86              }
87          }
88
89
90          if ("getOne_For_Update".equals(action)) { // 來自
   listAllEmp.jsp的請求
91
92              //why linkedlist? 介面一樣可以宣告變數？ 共同方法 add vector
   優點做鎖定
93              //多形的最佳栗子，左邊是介面，右邊是linkedList<E>()
94              List<String> errorMsgs = new LinkedList<String>();
95              // Store this set in the request scope, in case we need
   to
96              // send the ErrorPage view.
97              req.setAttribute("errorMsgs", errorMsgs);
98
99              try {
100                 /***************************1.接收請求參數
   ****************************************/
101                 Integer empno = new
   Integer(req.getParameter("empno"));
102
103                 /***************************2.開始查詢資料
   ****************************************/
104                 EmpService empSvc = new EmpService();
105                 EmpVO empVO = empSvc.getOneEmp(empno);
106
107                 /***************************3.查詢完成,準備轉交(Send
   the Success view)************/
108                 req.setAttribute("empVO", empVO);          // 資料庫取
   出的empVO物件,存入req
109                 String url = "/emp/update_emp_input.jsp";
110                 RequestDispatcher successView =
   req.getRequestDispatcher(url);// 成功轉交 update_emp_input.jsp
111                 successView.forward(req, res);
112
113                 /***************************其他可能的錯誤處理
   **********************************/
114             } catch (Exception e) {
```

```
115                 errorMsgs.add("無法取得要修改的資料:" +
    e.getMessage());
116                 RequestDispatcher failureView = req
117                         .getRequestDispatcher("/emp/
    listAllEmp.jsp");
118                 failureView.forward(req, res);
119             }
120         }
121 
122 
123         if ("update".equals(action)) { // 來自update_emp_input.jsp的
    請求
124 
125             List<String> errorMsgs = new LinkedList<String>();
126             // Store this set in the request scope, in case we need
    to
127             // send the ErrorPage view.
128             req.setAttribute("errorMsgs", errorMsgs);
129 
130             try {
131                 /***************************1.接收請求參數 - 輸入格式的
    錯誤處理**********************/
132                 Integer empno = new
    Integer(req.getParameter("empno").trim());
133                 String ename = req.getParameter("ename").trim();
134                 String job = req.getParameter("job").trim();

135 
136                 java.sql.Date hiredate = null;
137                 try {
138                     hiredate =
    java.sql.Date.valueOf(req.getParameter("hiredate").trim());
139                 } catch (IllegalArgumentException e) {
140                     hiredate=new
    java.sql.Date(System.currentTimeMillis());
141                     errorMsgs.add("請輸入日期!");
142                 }
143 
144                 Double sal = null;
145                 try {
146                     sal = new
    Double(req.getParameter("sal").trim());
147                 } catch (NumberFormatException e) {
148                     sal = 0.0;
149                     errorMsgs.add("薪水請填數字.");
```

```
150                 }
151
152                 Double comm = null;
153                 try {
154                     comm = new
    Double(req.getParameter("comm").trim());
155                 } catch (NumberFormatException e) {
156                     comm = 0.0;
157                     errorMsgs.add("獎金請填數字.");
158                 }
159
160                 Integer deptno = new
    Integer(req.getParameter("deptno").trim());
161
162                 EmpVO empVO = new EmpVO();
163                 empVO.setEmpno(empno);
164                 empVO.setEname(ename);
165                 empVO.setJob(job);
166                 empVO.setHiredate(hiredate);
167                 empVO.setSal(sal);
168                 empVO.setComm(comm);
169                 empVO.setDeptno(deptno);
170
171                 // Send the use back to the form, if there were
    errors
172                 if (!errorMsgs.isEmpty()) {
173                     req.setAttribute("empVO", empVO); // 含有輸入格式
    錯誤的empVO物件,也存入req
174                     RequestDispatcher failureView = req
175                             .getRequestDispatcher("/emp/
    update_emp_input.jsp");
176                     failureView.forward(req, res);
177                     return; //程式中斷
178                 }
179
180                 /***************************2.開始修改資料
    *****************************************/
181                 EmpService empSvc = new EmpService();
182                 empVO = empSvc.updateEmp(empno, ename, job,
    hiredate, sal,comm, deptno);
183
184                 /***************************3.修改完成,準備轉交(Send
    the Success view)*************/
185                 req.setAttribute("empVO", empVO); // 資料庫update成功
    後,正確的的empVO物件,存入req
```

```
186                 String url = "/emp/listOneEmp.jsp";
187                 RequestDispatcher successView =
    req.getRequestDispatcher(url); // 修改成功後,轉交listOneEmp.jsp
188                 successView.forward(req, res);
189
190                 /***************************其他可能的錯誤處理
    **************************************/
191                 //
192             } catch (Exception e) {
193                 errorMsgs.add("修改資料失敗:"+e.getMessage());
194                 RequestDispatcher failureView = req
195                         .getRequestDispatcher("/emp/
    update_emp_input.jsp");
196                 failureView.forward(req, res);
197             }
198         }
199
200         if ("insert".equals(action)) { // 來自addEmp.jsp的請求
201
202             List<String> errorMsgs = new LinkedList<String>();
203             // Store this set in the request scope, in case we need
    to
204             // send the ErrorPage view.
205             req.setAttribute("errorMsgs", errorMsgs);
206
207             try {
208                 /***********************1.接收請求參數 - 輸入格式的錯誤處
    理*************************/
209                 String ename = req.getParameter("ename").trim();
210                 String job = req.getParameter("job").trim();
211
212                 java.sql.Date hiredate = null;
213                 try {
214                     hiredate =
    java.sql.Date.valueOf(req.getParameter("hiredate").trim());
215                 } catch (IllegalArgumentException e) {
216                     hiredate=new
    java.sql.Date(System.currentTimeMillis());
217                     errorMsgs.add("請輸入日期!");
218                 }
219
220                 Double sal = null;
221                 try {
222                     sal = new
    Double(req.getParameter("sal").trim());
```

```
223                 } catch (NumberFormatException e) {
224                     sal = 0.0;
225                     errorMsgs.add("薪水請填數字.");
226                 }
227 
228                 Double comm = null;
229                 try {
230                     comm = new
    Double(req.getParameter("comm").trim());
231                 } catch (NumberFormatException e) {
232                     comm = 0.0;
233                     errorMsgs.add("獎金請填數字.");
234                 }
235 
236                 Integer deptno = new
    Integer(req.getParameter("deptno").trim());
237 
238                 EmpVO empVO = new EmpVO();
239                 empVO.setEname(ename);
240                 empVO.setJob(job);
241                 empVO.setHiredate(hiredate);
242                 empVO.setSal(sal);
243                 empVO.setComm(comm);
244                 empVO.setDeptno(deptno);
245 
246                 // Send the use back to the form, if there were
    errors
247                 if (!errorMsgs.isEmpty()) {
248                     req.setAttribute("empVO", empVO); // 含有輸入格式
    錯誤的empVO物件,也存入req
249                     RequestDispatcher failureView = req
250                             .getRequestDispatcher("/emp/
    addEmp.jsp");
251                     failureView.forward(req, res);
252                     return;
253                 }
254 
255                 /***************************2.開始新增資料
    ***************************************/
256                 EmpService empSvc = new EmpService();
257                 empVO = empSvc.addEmp(ename, job, hiredate, sal,
    comm, deptno);
258 
259                 /***************************3.新增完成,準備轉交(Send
    the Success view)***********/
```

```
260                 String url = "/emp/listAllEmp.jsp";
261                 RequestDispatcher successView =
    req.getRequestDispatcher(url); // 新增成功後轉交listAllEmp.jsp
262                 successView.forward(req, res);
263
264                 /***************************其他可能的錯誤處理
    **********************************/
265             } catch (Exception e) {
266                 errorMsgs.add(e.getMessage());
267                 RequestDispatcher failureView = req
268                         .getRequestDispatcher("/emp/addEmp.jsp");
269                 failureView.forward(req, res);
270             }
271         }
272
273
274         if ("delete".equals(action)) { // 來自listAllEmp.jsp
275
276             List<String> errorMsgs = new LinkedList<String>();
277             // Store this set in the request scope, in case we need
    to
278             // send the ErrorPage view.
279             req.setAttribute("errorMsgs", errorMsgs);
280
281             try {
282                 /***************************1.接收請求參數
    ****************************************/
283                 Integer empno = new
    Integer(req.getParameter("empno"));
284
285                 /***************************2.開始刪除資料
    ****************************************/
286                 EmpService empSvc = new EmpService();
287                 empSvc.deleteEmp(empno);
288
289                 /***************************3.刪除完成,準備轉交(Send
    the Success view)***********/
290                 String url = "/emp/listAllEmp.jsp";
291                 RequestDispatcher successView =
    req.getRequestDispatcher(url);// 刪除成功後,轉交回送出刪除的來源網頁
292                 successView.forward(req, res);
293
294                 /***************************其他可能的錯誤處理
    **********************************/
295             } catch (Exception e) {
```

```
296                 errorMsgs.add("刪除資料失敗:"+e.getMessage());
297                 RequestDispatcher failureView = req
298                         .getRequestDispatcher("/emp/
    listAllEmp.jsp");
299                 failureView.forward(req, res);
300             }
301         }
302     }
303 }
304
```

EmpDAO_interface.java

```java
package com.emp.model;

import java.util.*;

public interface EmpDAO_interface {
          public void insert(EmpVO empVO);
          public void update(EmpVO empVO);
          public void delete(Integer empno);
          public EmpVO findByPrimaryKey(Integer empno);
          public List<EmpVO> getAll();
          //萬用複合查詢(傳入參數型態Map)(回傳 List)
//        public List<EmpVO> getAll(Map<String, String[]> map);
}

```

```java
package com.emp.model;

import java.util.List;

public class EmpService {

    private EmpDAO_interface dao;

    public EmpService() {
        dao = new EmpDAO();
    }

    public EmpVO addEmp(String ename, String job, java.sql.Date hiredate,
            Double sal, Double comm, Integer deptno) {

        EmpVO empVO = new EmpVO();

        empVO.setEname(ename);
        empVO.setJob(job);
        empVO.setHiredate(hiredate);
        empVO.setSal(sal);
        empVO.setComm(comm);
        empVO.setDeptno(deptno);
        dao.insert(empVO);

        return empVO;
    }

    public EmpVO updateEmp(Integer empno, String ename, String job,
            java.sql.Date hiredate, Double sal, Double comm, Integer deptno) {

        EmpVO empVO = new EmpVO();

        empVO.setEmpno(empno);
        empVO.setEname(ename);
        empVO.setJob(job);
        empVO.setHiredate(hiredate);
        empVO.setSal(sal);
        empVO.setComm(comm);
        empVO.setDeptno(deptno);
        dao.update(empVO);

        return empVO;
```

```
44      }
45
46      public void deleteEmp(Integer empno) {
47          dao.delete(empno);
48      }
49
50      public EmpVO getOneEmp(Integer empno) {
51          return dao.findByPrimaryKey(empno);
52      }
53
54      public List<EmpVO> getAll() {
55          return dao.getAll();
56      }
57  }
58
```

```java
package com.emp.model;

import java.util.*;

public class EmpDAO implements EmpDAO_interface {

    // 一個應用程式中,針對一個資料庫 ,共用一個DataSource即可
    private static DataSource ds = null;
    static {
        try {
            Context ctx = new InitialContext();
            ds = (DataSource) ctx.lookup("java:comp/env/jdbc/TestDB");
        } catch (NamingException e) {
            e.printStackTrace();
        }
    }

    private static final String INSERT_STMT =
        "INSERT INTO emp2 (empno,ename,job,hiredate,sal,comm,deptno) VALUES (emp2_seq.NEXTVAL, ?, ?, ?, ?, ?, ?)";
    private static final String GET_ALL_STMT =
        "SELECT empno,ename,job,to_char(hiredate,'yyyy-mm-dd') hiredate,sal,comm,deptno FROM emp2 order by empno";
    private static final String GET_ONE_STMT =
        "SELECT empno,ename,job,to_char(hiredate,'yyyy-mm-dd') hiredate,sal,comm,deptno FROM emp2 where empno = ?";
    private static final String DELETE =
        "DELETE FROM emp2 where empno = ?";
    private static final String UPDATE =
        "UPDATE emp2 set ename=?, job=?, hiredate=?, sal=?, comm=?, deptno=? where empno = ?";

    @Override
    public void insert(EmpVO empVO) {

        Connection con = null;
        PreparedStatement pstmt = null;

        try {

            con = ds.getConnection();
            pstmt = con.prepareStatement(INSERT_STMT);

```

```
46              pstmt.setString(1, empVO.getEname());
47              pstmt.setString(2, empVO.getJob());
48              pstmt.setDate(3, empVO.getHiredate());
49              pstmt.setDouble(4, empVO.getSal());
50              pstmt.setDouble(5, empVO.getComm());
51              pstmt.setInt(6, empVO.getDeptno());
52
53              pstmt.executeUpdate();
54
55              // Handle any SQL errors
56          } catch (SQLException se) {
57              throw new RuntimeException("A database error occured. "
58                      + se.getMessage());
59              // Clean up JDBC resources
60          } finally {
61              if (pstmt != null) {
62                  try {
63                      pstmt.close();
64                  } catch (SQLException se) {
65                      se.printStackTrace(System.err);
66                  }
67              }
68              if (con != null) {
69                  try {
70                      con.close();
71                  } catch (Exception e) {
72                      e.printStackTrace(System.err);
73                  }
74              }
75          }
76
77      }
78
79      @Override
80      public void update(EmpVO empVO) {
81
82          Connection con = null;
83          PreparedStatement pstmt = null;
84
85          try {
86
87              con = ds.getConnection();
88              pstmt = con.prepareStatement(UPDATE);
89
90              pstmt.setString(1, empVO.getEname());
```

```
91              pstmt.setString(2, empVO.getJob());
92              pstmt.setDate(3, empVO.getHiredate());
93              pstmt.setDouble(4, empVO.getSal());
94              pstmt.setDouble(5, empVO.getComm());
95              pstmt.setInt(6, empVO.getDeptno());
96              pstmt.setInt(7, empVO.getEmpno());
97
98              pstmt.executeUpdate();
99
100             // Handle any driver errors
101         } catch (SQLException se) {
102             throw new RuntimeException("A database error occured. "
103                     + se.getMessage());
104             // Clean up JDBC resources
105         } finally {
106             if (pstmt != null) {
107                 try {
108                     pstmt.close();
109                 } catch (SQLException se) {
110                     se.printStackTrace(System.err);
111                 }
112             }
113             if (con != null) {
114                 try {
115                     con.close();
116                 } catch (Exception e) {
117                     e.printStackTrace(System.err);
118                 }
119             }
120         }
121
122     }
123
124     @Override
125     public void delete(Integer empno) {
126
127         Connection con = null;
128         PreparedStatement pstmt = null;
129
130         try {
131
132             con = ds.getConnection();
133             pstmt = con.prepareStatement(DELETE);
134
135             pstmt.setInt(1, empno);
```

```
136
137             pstmt.executeUpdate();
138
139             // Handle any driver errors
140         } catch (SQLException se) {
141             throw new RuntimeException("A database error occured. "
142                     + se.getMessage());
143             // Clean up JDBC resources
144         } finally {
145             if (pstmt != null) {
146                 try {
147                     pstmt.close();
148                 } catch (SQLException se) {
149                     se.printStackTrace(System.err);
150                 }
151             }
152             if (con != null) {
153                 try {
154                     con.close();
155                 } catch (Exception e) {
156                     e.printStackTrace(System.err);
157                 }
158             }
159         }
160
161     }
162
163     @Override
164     public EmpVO findByPrimaryKey(Integer empno) {
165
166         EmpVO empVO = null;
167         Connection con = null;
168         PreparedStatement pstmt = null;
169         ResultSet rs = null;
170
171         try {
172
173             con = ds.getConnection();
174             pstmt = con.prepareStatement(GET_ONE_STMT);
175
176             pstmt.setInt(1, empno);
177
178             rs = pstmt.executeQuery();
179
180             while (rs.next()) {
```

```
181                 // empVo 也稱為 Domain objects
182                 empVO = new EmpVO();
183                 empVO.setEmpno(rs.getInt("empno"));
184                 empVO.setEname(rs.getString("ename"));
185                 empVO.setJob(rs.getString("job"));
186                 empVO.setHiredate(rs.getDate("hiredate"));
187                 empVO.setSal(rs.getDouble("sal"));
188                 empVO.setComm(rs.getDouble("comm"));
189                 empVO.setDeptno(rs.getInt("deptno"));
190             }
191 
192             // Handle any driver errors
193         } catch (SQLException se) {
194             throw new RuntimeException("A database error occured. "
195                     + se.getMessage());
196             // Clean up JDBC resources
197         } finally {
198             if (rs != null) {
199                 try {
200                     rs.close();
201                 } catch (SQLException se) {
202                     se.printStackTrace(System.err);
203                 }
204             }
205             if (pstmt != null) {
206                 try {
207                     pstmt.close();
208                 } catch (SQLException se) {
209                     se.printStackTrace(System.err);
210                 }
211             }
212             if (con != null) {
213                 try {
214                     con.close();
215                 } catch (Exception e) {
216                     e.printStackTrace(System.err);
217                 }
218             }
219         }
220         return empVO;
221     }
222 
223     @Override
224     public List<EmpVO> getAll() {
225         List<EmpVO> list = new ArrayList<EmpVO>();
```

```
226         EmpVO empVO = null;
227
228         Connection con = null;
229         PreparedStatement pstmt = null;
230         ResultSet rs = null;
231
232         try {
233
234             con = ds.getConnection();
235             pstmt = con.prepareStatement(GET_ALL_STMT);
236             rs = pstmt.executeQuery();
237
238             while (rs.next()) {
239                 // empVO 也稱為 Domain objects
240                 empVO = new EmpVO();
241                 empVO.setEmpno(rs.getInt("empno"));
242                 empVO.setEname(rs.getString("ename"));
243                 empVO.setJob(rs.getString("job"));
244                 empVO.setHiredate(rs.getDate("hiredate"));
245                 empVO.setSal(rs.getDouble("sal"));
246                 empVO.setComm(rs.getDouble("comm"));
247                 empVO.setDeptno(rs.getInt("deptno"));
248                 list.add(empVO); // Store the row in the list
249             }
250
251             // Handle any driver errors
252         } catch (SQLException se) {
253             throw new RuntimeException("A database error occured. "
254                     + se.getMessage());
255             // Clean up JDBC resources
256         } finally {
257             if (rs != null) {
258                 try {
259                     rs.close();
260                 } catch (SQLException se) {
261                     se.printStackTrace(System.err);
262                 }
263             }
264             if (pstmt != null) {
265                 try {
266                     pstmt.close();
267                 } catch (SQLException se) {
268                     se.printStackTrace(System.err);
269                 }
270             }
```

```
271             if (con != null) {
272                 try {
273                     con.close();
274                 } catch (Exception e) {
275                     e.printStackTrace(System.err);
276                 }
277             }
278         }
279         return list;
280     }
281 }
```

```java
package com.emp.model;

import java.util.*;

public class EmpJDBCDAO implements EmpDAO_interface {
    String driver = "oracle.jdbc.driver.OracleDriver";
    String url = "jdbc:oracle:thin:@localhost:1521:XE";
    String userid = "hr";
    String passwd = "123456";

    private static final String INSERT_STMT =
        "INSERT INTO emp2
(empno,ename,job,hiredate,sal,comm,deptno) VALUES
(emp2_seq.NEXTVAL, ?, ?, ?, ?, ?, ?)";
    private static final String GET_ALL_STMT =
        "SELECT empno,ename,job,to_char(hiredate,'yyyy-mm-dd')
hiredate,sal,comm,deptno FROM emp2 order by empno";
    private static final String GET_ONE_STMT =
        "SELECT empno,ename,job,to_char(hiredate,'yyyy-mm-dd')
hiredate,sal,comm,deptno FROM emp2 where empno = ?";
    private static final String DELETE =
        "DELETE FROM emp2 where empno = ?";
    private static final String UPDATE =
        "UPDATE emp2 set ename=?, job=?, hiredate=?, sal=?, comm=?,
deptno=? where empno = ?";

    @Override
    public void insert(EmpVO empVO) {

        Connection con = null;
        PreparedStatement pstmt = null;

        try {

            Class.forName(driver);
            con = DriverManager.getConnection(url, userid, passwd);
            pstmt = con.prepareStatement(INSERT_STMT);

            pstmt.setString(1, empVO.getEname());
            pstmt.setString(2, empVO.getJob());
            pstmt.setDate(3, empVO.getHiredate());
            pstmt.setDouble(4, empVO.getSal());
            pstmt.setDouble(5, empVO.getComm());
            pstmt.setInt(6, empVO.getDeptno());

```

```
42              pstmt.executeUpdate();
43
44              // Handle any driver errors
45          } catch (ClassNotFoundException e) {
46              throw new RuntimeException("Couldn't load database
  driver. "
47                      + e.getMessage());
48              // Handle any SQL errors
49          } catch (SQLException se) {
50              throw new RuntimeException("A database error occured. "
51                      + se.getMessage());
52              // Clean up JDBC resources
53          } finally {
54              if (pstmt != null) {
55                  try {
56                      pstmt.close();
57                  } catch (SQLException se) {
58                      se.printStackTrace(System.err);
59                  }
60              }
61              if (con != null) {
62                  try {
63                      con.close();
64                  } catch (Exception e) {
65                      e.printStackTrace(System.err);
66                  }
67              }
68          }
69
70      }
71
72      @Override
73      public void update(EmpVO empVO) {
74
75          Connection con = null;
76          PreparedStatement pstmt = null;
77
78          try {
79
80              Class.forName(driver);
81              con = DriverManager.getConnection(url, userid, passwd);
82              pstmt = con.prepareStatement(UPDATE);
83
84              pstmt.setString(1, empVO.getEname());
85              pstmt.setString(2, empVO.getJob());
```

```
 86              pstmt.setDate(3, empVO.getHiredate());
 87              pstmt.setDouble(4, empVO.getSal());
 88              pstmt.setDouble(5, empVO.getComm());
 89              pstmt.setInt(6, empVO.getDeptno());
 90              pstmt.setInt(7, empVO.getEmpno());
 91
 92              pstmt.executeUpdate();
 93
 94              // Handle any driver errors
 95          } catch (ClassNotFoundException e) {
 96              throw new RuntimeException("Couldn't load database
    driver. "
 97                      + e.getMessage());
 98              // Handle any SQL errors
 99          } catch (SQLException se) {
100              throw new RuntimeException("A database error occured. "
101                      + se.getMessage());
102              // Clean up JDBC resources
103          } finally {
104              if (pstmt != null) {
105                  try {
106                      pstmt.close();
107                  } catch (SQLException se) {
108                      se.printStackTrace(System.err);
109                  }
110              }
111              if (con != null) {
112                  try {
113                      con.close();
114                  } catch (Exception e) {
115                      e.printStackTrace(System.err);
116                  }
117              }
118          }
119
120      }
121
122      @Override
123      public void delete(Integer empno) {
124
125          Connection con = null;
126          PreparedStatement pstmt = null;
127
128          try {
129
```

```
130             Class.forName(driver);
131             con = DriverManager.getConnection(url, userid, passwd);
132             pstmt = con.prepareStatement(DELETE);
133 
134             pstmt.setInt(1, empno);
135 
136             pstmt.executeUpdate();
137 
138             // Handle any driver errors
139         } catch (ClassNotFoundException e) {
140             throw new RuntimeException("Couldn't load database
    driver. "
141                     + e.getMessage());
142             // Handle any SQL errors
143         } catch (SQLException se) {
144             throw new RuntimeException("A database error occured. "
145                     + se.getMessage());
146             // Clean up JDBC resources
147         } finally {
148             if (pstmt != null) {
149                 try {
150                     pstmt.close();
151                 } catch (SQLException se) {
152                     se.printStackTrace(System.err);
153                 }
154             }
155             if (con != null) {
156                 try {
157                     con.close();
158                 } catch (Exception e) {
159                     e.printStackTrace(System.err);
160                 }
161             }
162         }
163 
164     }
165 
166     @Override
167     public EmpVO findByPrimaryKey(Integer empno) {
168 
169         EmpVO empVO = null;
170         Connection con = null;
171         PreparedStatement pstmt = null;
172         ResultSet rs = null;
173 
```

```
174         try {
175 
176             Class.forName(driver);
177             con = DriverManager.getConnection(url, userid, passwd);
178             pstmt = con.prepareStatement(GET_ONE_STMT);
179 
180             pstmt.setInt(1, empno);
181 
182             rs = pstmt.executeQuery();
183 
184             while (rs.next()) {
185                 // empVo 也稱為 Domain objects
186                 empVO = new EmpVO();
187                 empVO.setEmpno(rs.getInt("empno"));
188                 empVO.setEname(rs.getString("ename"));
189                 empVO.setJob(rs.getString("job"));
190                 empVO.setHiredate(rs.getDate("hiredate"));
191                 empVO.setSal(rs.getDouble("sal"));
192                 empVO.setComm(rs.getDouble("comm"));
193                 empVO.setDeptno(rs.getInt("deptno"));
194             }
195 
196             // Handle any driver errors
197         } catch (ClassNotFoundException e) {
198             throw new RuntimeException("Couldn't load database
    driver. "
199                     + e.getMessage());
200             // Handle any SQL errors
201         } catch (SQLException se) {
202             throw new RuntimeException("A database error occured. "
203                     + se.getMessage());
204             // Clean up JDBC resources
205         } finally {
206             if (rs != null) {
207                 try {
208                     rs.close();
209                 } catch (SQLException se) {
210                     se.printStackTrace(System.err);
211                 }
212             }
213             if (pstmt != null) {
214                 try {
215                     pstmt.close();
216                 } catch (SQLException se) {
217                     se.printStackTrace(System.err);
```

```
218                 }
219             }
220             if (con != null) {
221                 try {
222                     con.close();
223                 } catch (Exception e) {
224                     e.printStackTrace(System.err);
225                 }
226             }
227         }
228         return empVO;
229     }
230 
231     @Override
232     public List<EmpVO> getAll() {
233         List<EmpVO> list = new ArrayList<EmpVO>();
234         EmpVO empVO = null;
235 
236         Connection con = null;
237         PreparedStatement pstmt = null;
238         ResultSet rs = null;
239 
240         try {
241 
242             Class.forName(driver);
243             con = DriverManager.getConnection(url, userid, passwd);
244             pstmt = con.prepareStatement(GET_ALL_STMT);
245             rs = pstmt.executeQuery();
246 
247             while (rs.next()) {
248                 // empVO 也稱為 Domain objects
249                 empVO = new EmpVO();
250                 empVO.setEmpno(rs.getInt("empno"));
251                 empVO.setEname(rs.getString("ename"));
252                 empVO.setJob(rs.getString("job"));
253                 empVO.setHiredate(rs.getDate("hiredate"));
254                 empVO.setSal(rs.getDouble("sal"));
255                 empVO.setComm(rs.getDouble("comm"));
256                 empVO.setDeptno(rs.getInt("deptno"));
257                 list.add(empVO); // Store the row in the list
258             }
259 
260             // Handle any driver errors
261         } catch (ClassNotFoundException e) {
262             throw new RuntimeException("Couldn't load database
```

```
    driver. "
263                     + e.getMessage());
264             // Handle any SQL errors
265         } catch (SQLException se) {
266             throw new RuntimeException("A database error occured. "
267                     + se.getMessage());
268             // Clean up JDBC resources
269         } finally {
270             if (rs != null) {
271                 try {
272                     rs.close();
273                 } catch (SQLException se) {
274                     se.printStackTrace(System.err);
275                 }
276             }
277             if (pstmt != null) {
278                 try {
279                     pstmt.close();
280                 } catch (SQLException se) {
281                     se.printStackTrace(System.err);
282                 }
283             }
284             if (con != null) {
285                 try {
286                     con.close();
287                 } catch (Exception e) {
288                     e.printStackTrace(System.err);
289                 }
290             }
291         }
292         return list;
293     }
294 
295     public static void main(String[] args) {
296 
297         EmpJDBCDAO dao = new EmpJDBCDAO();
298 
299         // 新增
300         EmpVO empVO1 = new EmpVO();
301         empVO1.setEname("吳永志1");
302         empVO1.setJob("MANAGER");
303         empVO1.setHiredate(java.sql.Date.valueOf("2005-01-01"));
304         empVO1.setSal(new Double(50000));
305         empVO1.setComm(new Double(500));
306         empVO1.setDeptno(10);
```

```java
        dao.insert(empVO1);

        // 修改
        EmpVO empVO2 = new EmpVO();
        empVO2.setEmpno(7001);
        empVO2.setEname("吳永志2");
        empVO2.setJob("MANAGER2");
        empVO2.setHiredate(java.sql.Date.valueOf("2002-01-01"));
        empVO2.setSal(new Double(20000));
        empVO2.setComm(new Double(200));
        empVO2.setDeptno(20);
        dao.update(empVO2);

        // 刪除
        dao.delete(7014);

        // 查詢
        EmpVO empVO3 = dao.findByPrimaryKey(7001);
        System.out.print(empVO3.getEmpno() + ",");
        System.out.print(empVO3.getEname() + ",");
        System.out.print(empVO3.getJob() + ",");
        System.out.print(empVO3.getHiredate() + ",");
        System.out.print(empVO3.getSal() + ",");
        System.out.print(empVO3.getComm() + ",");
        System.out.println(empVO3.getDeptno());
        System.out.println("---------------------");

        // 查詢
        List<EmpVO> list = dao.getAll();
        for (EmpVO aEmp : list) {
            System.out.print(aEmp.getEmpno() + ",");
            System.out.print(aEmp.getEname() + ",");
            System.out.print(aEmp.getJob() + ",");
            System.out.print(aEmp.getHiredate() + ",");
            System.out.print(aEmp.getSal() + ",");
            System.out.print(aEmp.getComm() + ",");
            System.out.print(aEmp.getDeptno());
            System.out.println();
        }
    }
}
```

```java
package com.emp.model;
import java.sql.Date;

public class EmpVO implements java.io.Serializable{
    private Integer empno;
    private String ename;
    private String job;
    private Date hiredate;
    private Double sal;
    private Double comm;
    private Integer deptno;

    public Integer getEmpno() {
        return empno;
    }
    public void setEmpno(Integer empno) {
        this.empno = empno;
    }
    public String getEname() {
        return ename;
    }
    public void setEname(String ename) {
        this.ename = ename;
    }
    public String getJob() {
        return job;
    }
    public void setJob(String job) {
        this.job = job;
    }
    public Date getHiredate() {
        return hiredate;
    }
    public void setHiredate(Date hiredate) {
        this.hiredate = hiredate;
    }
    public Double getSal() {
        return sal;
    }
    public void setSal(Double sal) {
        this.sal = sal;
    }
    public Double getComm() {
        return comm;
    }
    }
```

```
46      public void setComm(Double comm) {
47          this.comm = comm;
48      }
49      public Integer getDeptno() {
50          return deptno;
51      }
52      public void setDeptno(Integer deptno) {
53          this.deptno = deptno;
54      }
55  }
56
```

page1.file

```
<%@ page contentType="text/html; charset=UTF-8"
pageEncoding="Big5"%>
<%  int rowsPerPage = 3;  //每頁的筆數
    int rowNumber=0;      //總筆數
    int pageNumber=0;     //總頁數
    int whichPage=1;      //第幾頁
    int pageIndexArray[]=null;
    int pageIndex=0;
%>

<%
    rowNumber=list.size();
    if (rowNumber%rowsPerPage !=0)
     pageNumber=rowNumber/rowsPerPage +1;
    else pageNumber=rowNumber/rowsPerPage;

    pageIndexArray=new int[pageNumber];
    for (int i=1 ; i<=pageIndexArray.length ; i++)
    pageIndexArray[i-1]=i*rowsPerPage-rowsPerPage;
%>

<%  try {
      whichPage =
Integer.parseInt(request.getParameter("whichPage"));
      pageIndex=pageIndexArray[whichPage-1];
    } catch (NumberFormatException e) { //第一次執行的時候
       whichPage=1;
       pageIndex=0;
    } catch (ArrayIndexOutOfBoundsException e) { //總頁數之外的錯誤頁數
         if (pageNumber>0){
              whichPage=pageNumber;
              pageIndex=pageIndexArray[pageNumber-1];
         }
    }
%>
<%if (pageNumber>0){%>
<b><font color= red>第<%=whichPage%>/<%=pageNumber%>頁  </font></b>
<%}%>
<b>•符 合 查 詢 條 件 如 下 所 示: 共<font color=red><%=rowNumber%></
font>筆</b>


```

```
<%@ page contentType="text/html; charset=UTF-8"
pageEncoding="Big5"%>
<table border="0">
 <tr>
  <%if (rowsPerPage<rowNumber) {%>
    <%if(pageIndex>=rowsPerPage){%>
        <td><A href="<%=request.getRequestURI()%>?whichPage=1">至第一
頁</A> </td>
        <td><A href="<%=request.getRequestURI()%>?whichPage=<
%=whichPage-1%>">上一頁 </A> </td>
    <%}%>

    <%if(pageIndex<pageIndexArray[pageNumber-1]){%>
        <td><A href="<%=request.getRequestURI()%>?whichPage=<
%=whichPage+1%>">下一頁 </A> </td>
        <td><A href="<%=request.getRequestURI()%>?whichPage=<
%=pageNumber%>">至最後一頁</A> </td>
    <%}%>
  <%}%>
 </tr>
</table>
<%if ( pageNumber > 1) {%>
<table border="0">
 <tr>

<td>          &nbs
p;     

           &n
bsp;   
        </td>
   <FORM METHOD="post" ACTION="<%=request.getRequestURI()%>">
        <td>
           <select size="1" name="whichPage">
        <%for (int i=1; i<=pageNumber; i++){%>
           <option value="<%=i%>">跳至第<%=i%>頁
        <%}%>
           </select>
           <input type="submit" value="確定" >
        </td>
   </FORM>
 </tr>
</table>
<%}%>
```

```
 1 建立表格: EMP2 和 DEPT2
 2 ------------------------------------------------------------------------
   ------------------------------------
 3 DROP TABLE EMP2;
 4 DROP TABLE DEPT2;
 5 DROP SEQUENCE emp2_seq;
 6 DROP SEQUENCE dept2_seq;
 7
 8
 9 CREATE TABLE DEPT2 (
10  DEPTNO              NUMBER(2) NOT NULL,
11  DNAME               VARCHAR2(14),
12  LOC                      VARCHAR2(13),
13  CONSTRAINT DEPT2_PRIMARY_KEY PRIMARY KEY (DEPTNO));
14
15 CREATE SEQUENCE dept2_seq
16 INCREMENT BY 10
17 START WITH 10
18 NOMAXVALUE
19 NOCYCLE
20 NOCACHE;
21
22 INSERT INTO DEPT2 VALUES (dept2_seq.NEXTVAL,'財務部','臺灣台北');
23 INSERT INTO DEPT2 VALUES (dept2_seq.NEXTVAL,'研發部','臺灣新竹');
24 INSERT INTO DEPT2 VALUES (dept2_seq.NEXTVAL,'業務部','美國紐約');
25 INSERT INTO DEPT2 VALUES (dept2_seq.NEXTVAL,'生管部','中國上海');
26
27
28
29 CREATE TABLE EMP2 (
30  EMPNO                NUMBER(4) NOT NULL,
31  ENAME                 VARCHAR2(10),
32  JOB                       VARCHAR2(9),
33  HIREDATE           DATE,
34  SAL                       NUMBER(7,2),
35  COMM                NUMBER(7,2),
36  DEPTNO               NUMBER(2) NOT NULL,
37  CONSTRAINT EMP2_DEPTNO_FK FOREIGN KEY (DEPTNO) REFERENCES DEPT2
   (DEPTNO),
38  CONSTRAINT EMP2_EMPNO_PK PRIMARY KEY (EMPNO));
39
40 CREATE SEQUENCE emp2_seq
41 INCREMENT BY 1
42 START WITH 7001
43 NOMAXVALUE
```

```
44 NOCYCLE
45 NOCACHE;
46
47 INSERT INTO EMP2 VALUES
   (emp2_seq.NEXTVAL,'KING','PRESIDENT',TO_DATE('1981-11-17','YYYY-MM-
   DD'),5000.5,0.0,10);
48 INSERT INTO EMP2 VALUES
   (emp2_seq.NEXTVAL,'BLAKE','MANAGER',TO_DATE('1981-05-01','YYYY-MM-
   DD'),2850,0.0,30);
49 INSERT INTO EMP2 VALUES
   (emp2_seq.NEXTVAL,'CLARK','MANAGER',TO_DATE('1981-01-09','YYYY-MM-
   DD'),2450,0.0,10);
50 INSERT INTO EMP2 VALUES
   (emp2_seq.NEXTVAL,'JONES','MANAGER',TO_DATE('1981-04-02','YYYY-MM-
   DD'),2975,0.0,20);
51 INSERT INTO EMP2 VALUES
   (emp2_seq.NEXTVAL,'MARTIN','SALESMAN',TO_DATE('1981-09-28','YYYY-MM-
   DD'),1250,1400,30);
52 INSERT INTO EMP2 VALUES
   (emp2_seq.NEXTVAL,'ALLEN','SALESMAN',TO_DATE('1981-02-2','YYYY-MM-
   DD'),1600,300,30);
53 INSERT INTO EMP2 VALUES
   (emp2_seq.NEXTVAL,'TURNER','SALESMAN',TO_DATE('1981-09-28','YYYY-MM-
   DD'),1500,0,30);
54 INSERT INTO EMP2 VALUES
   (emp2_seq.NEXTVAL,'JAMES','CLERK',TO_DATE('1981-12-03','YYYY-MM-
   DD'),950,0.0,30);
55 INSERT INTO EMP2 VALUES
   (emp2_seq.NEXTVAL,'WARD','SALESMAN',TO_DATE('1981-02-22','YYYY-MM-
   DD'),1250,500,30);
56 INSERT INTO EMP2 VALUES
   (emp2_seq.NEXTVAL,'FORD','ANALYST',TO_DATE('1981-12-03','YYYY-MM-
   DD'),3000,0.0,20);
57 INSERT INTO EMP2 VALUES
   (emp2_seq.NEXTVAL,'SMITH','CLERK',TO_DATE('1980-12-17','YYYY-MM-
   DD'),800,0.0,20);
58 INSERT INTO EMP2 VALUES
   (emp2_seq.NEXTVAL,'SCOTT','ANALYST',TO_DATE('1981-12-09','YYYY-MM-
   DD'),3000,0.0,20);
59 INSERT INTO EMP2 VALUES
   (emp2_seq.NEXTVAL,'ADAMS','CLERK',TO_DATE('1983-01-12','YYYY-MM-
   DD'),1100,0.0,20);
60 INSERT INTO EMP2 VALUES
   (emp2_seq.NEXTVAL,'MILLER','CLERK',TO_DATE('1982-01-23','YYYY-MM-
   DD'),1300,0.0,10);
```

```
61
62
63 commit;
64 ------------------------------------------------------------------------
   ------------------------------------
65
```

```
CREATE SEQUENCE emp2_seq
INCREMENT BY 1
START WITH 7001
NOMAXVALUE
NOCYCLE
NOCACHE;

查詢目前表格中有使用中sequence
 SELECT * FROM USER_SEQUENCES;

刪除sequence:
 DROP SEQUENCE emp2_seq;

●取用一組新的值:
 SELECT emp2_seq.NEXTVAL FROM DUAL;

得知目前使用到幾號 :
 SELECT emp2_seq.CURRVAL FROM DUAL;
```

```
CREATE SEQUENCE dept2_seq
INCREMENT BY 10
START WITH 10
NOMAXVALUE
NOCYCLE
NOCACHE;

查詢目前表格中有使用中sequence
 SELECT * FROM USER_SEQUENCES;

刪除sequence:
 DROP SEQUENCE dept2_seq;

●取用一組新的值:
 SELECT dept2_seq.NEXTVAL FROM DUAL;

得知目前使用到幾號 :
 SELECT dept2_seq.CURRVAL FROM DUAL;


```

```jsp
<%@ page contentType="text/html; charset=UTF-8"
pageEncoding="Big5"%>
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>
<%@ page import="com.emp.model.*"%>
<%
    EmpVO empVO = (EmpVO) request.getAttribute("empVO"); //
EmpServlet.java (Concroller), 存入req的empVO物件 (包括幫忙取出的empVO, 也
包括輸入資料錯誤時的empVO物件)
%>
<html>
<head>
<title>員工資料修改 - update_emp_input.jsp</title></head>
<link rel="stylesheet" type="text/css" href="js/calendar.css">
<script language="JavaScript" src="js/calendarcode.js"></script>
<div id="popupcalendar" class="text"></div>

<body bgcolor='white'>

<table border='1' cellpadding='5' cellspacing='0' width='400'>
    <tr bgcolor='#CCCCFF' align='center' valign='middle'
height='20'>
        <td>
        <h3>員工資料修改 - update_emp_input.jsp</h3>
        <a href="select_page.jsp"><img src="images/back1.gif"
width="100" height="32" border="0">回首頁</a></td>
    </tr>
</table>

<h3>資料修改:</h3>
<%-- 錯誤表列 --%>
<c:if test="${not empty errorMsgs}">
    <font color='red'>請修正以下錯誤:
    <ul>
        <c:forEach var="message" items="${errorMsgs}">
            <li>${message}</li>
        </c:forEach>
    </ul>
    </font>
</c:if>

<FORM METHOD="post" ACTION="emp.do" name="form1">
<table border="0">
    <tr>
        <td>員工編號:<font color=red><b>*</b></font></td>
        <td><%=empVO.getEmpno()%></td>
```

```
41      </tr>
42      <tr>
43          <td>員工姓名:</td>
44          <td><input type="TEXT" name="ename" size="45" value="<
  %=empVO.getEname()%>" /></td>
45      </tr>
46      <tr>
47          <td>職位:</td>
48          <td><input type="TEXT" name="job" size="45" value="<
  %=empVO.getJob()%>" /></td>
49      </tr>
50      <tr>
51          <td>雇用日期:</td>
52          <td bgcolor="#CCCCFF">
53              <input class="cal-TextBox"
54              onFocus="this.blur()" size="9" readonly type="text"
  name="hiredate" value="<%=empVO.getHiredate()%>">
55              <a class="so-BtnLink"
56              href="javascript:calClick();return false;"
57              onmouseover="calSwapImg('BTN_date',
  'img_Date_OVER',true);"
58              onmouseout="calSwapImg('BTN_date', 'img_Date_UP',true);"
59              onclick="calSwapImg('BTN_date',
  'img_Date_DOWN');showCalendar('form1','hiredate','BTN_date');return
  false;">
60              <img align="middle" border="0" name="BTN_date"
  src="images/btn_date_up.gif" width="22" height="17" alt="開始日期"></
  a>
61          </td>
62      </tr>
63      <tr>
64          <td>薪水:</td>
65          <td><input type="TEXT" name="sal" size="45" value="<
  %=empVO.getSal()%>" /></td>
66      </tr>
67      <tr>
68          <td>獎金:</td>
69          <td><input type="TEXT" name="comm" size="45" value="<
  %=empVO.getComm()%>" /></td>
70      </tr>
71
72      <jsp:useBean id="deptSvc" scope="page"
  class="com.dept.model.DeptService" />
73      <tr>
74          <td>部門:<font color=red><b>*</b></font></td>
```

```
75          <td><select size="1" name="deptno">
76              <c:forEach var="deptVO" items="${deptSvc.all}">
77                  <option value="${deptVO.deptno}" $
  {(empVO.deptno==deptVO.deptno)?'selected':'' } >${deptVO.dname}
78              </c:forEach>
79          </select></td>
80      </tr>
81
82 </table>
83 <br>
84 <input type="hidden" name="action" value="update">
85 <input type="hidden" name="empno" value="<%=empVO.getEmpno()%>">
86 <input type="submit" value="送出修改"></FORM>
87
88 </body>
89 </html>
90
```

```jsp
<%@ page contentType="text/html; charset=UTF-8"
pageEncoding="Big5"%>
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<html>
<head><title>IBM Emp: Home</title></head>
<body bgcolor='white'>

<table border='1' cellpadding='5' cellspacing='0' width='400'>
  <tr bgcolor='#CCCCFF' align='center' valign='middle' height='20'>
    <td><h3>IBM Emp: Home</h3><font color=red>( MVC )</font></td>
  </tr>
</table>

<p>This is the Home page for IBM Emp: Home</p>

<h3>資料查詢:</h3>
<%-- 錯誤表列 --%>
<c:if test="${not empty errorMsgs}">
    <font color='red'>請修正以下錯誤:
    <ul>
        <c:forEach var="message" items="${errorMsgs}">
            <li>${message}</li>
        </c:forEach>
    </ul>
    </font>
</c:if>

<ul>
  <li><a href='listAllEmp.jsp'>List</a> all Emps. </li> <br><br>

  <li>
    <FORM METHOD="post" ACTION="emp.do" >
        <b>輸入員工編號 (如7001):</b>
        <input type="text" name="empno">
        <input type="submit" value="送出">
        <input type="hidden" name="action"
value="getOne_For_Display">
    </FORM>
  </li>

  <jsp:useBean id="empSvc" scope="page"
class="com.emp.model.EmpService" />

  <li>
     <FORM METHOD="post" ACTION="emp.do" >
```

```
43        <b>選擇員工編號:</b>
44        <select size="1" name="empno">
45          <c:forEach var="empVO" items="${empSvc.all}" >
46           <option value="${empVO.empno}">${empVO.empno}
47          </c:forEach>
48        </select>
49        <input type="submit" value="送出">
50        <input type="hidden" name="action"
  value="getOne_For_Display">
51     </FORM>
52   </li>
53
54   <li>
55      <FORM METHOD="post" ACTION="emp.do" >
56        <b>選擇員工姓名:</b>
57        <select size="1" name="empno">
58          <c:forEach var="empVO" items="${empSvc.all}" >
59           <option value="${empVO.empno}">${empVO.ename}
60          </c:forEach>
61        </select>
62        <input type="submit" value="送出">
63        <input type="hidden" name="action"
  value="getOne_For_Display">
64      </FORM>
65   </li>
66 </ul>
67
68
69 <h3>員工管理</h3>
70
71 <ul>
72   <li><a href='addEmp.jsp'>Add</a> a new Emp.</li>
73 </ul>
74
75 </body>
76
77 </html>
78
```

```jsp
<%@ page contentType="text/html; charset=UTF-8"
pageEncoding="Big5"%>
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>
<%@ page import="com.emp.model.*"%>
<%
EmpVO empVO = (EmpVO) request.getAttribute("empVO");
%>

<html>
<head>
<title>員工資料新增 - addEmp.jsp</title></head>
<link rel="stylesheet" type="text/css" href="js/calendar.css">
<script language="JavaScript" src="js/calendarcode.js"></script>
<div id="popupcalendar" class="text"></div>

<body bgcolor='white'>

<table border='1' cellpadding='5' cellspacing='0' width='400'>
    <tr bgcolor='#CCCCFF' align='center' valign='middle'
height='20'>
        <td>
        <h3>員工資料新增 - addEmp.jsp</h3>
        </td>
        <td>
           <a href="select_page.jsp"><img src="images/tomcat.gif"
width="100" height="100" border="1">回首頁</a>
        </td>
    </tr>
</table>

<h3>資料員工:</h3>
<%-- 錯誤表列 --%>
<c:if test="${not empty errorMsgs}">
    <font color='red'>請修正以下錯誤:
    <ul>
        <c:forEach var="message" items="${errorMsgs}">
            <li>${message}</li>
        </c:forEach>
    </ul>
    </font>
</c:if>

<FORM METHOD="post" ACTION="emp.do" name="form1">
<table border="0">

```

```
43      <tr>
44          <td>員工姓名:</td>
45          <td><input type="TEXT" name="ename" size="45"
46              value="<%= (empVO==null)? "吳永志" : empVO.getEname()
  %>" /></td>
47      </tr>
48      <tr>
49          <td>職位:</td>
50          <td><input type="TEXT" name="job" size="45"
51              value="<%= (empVO==null)? "MANAGER" : empVO.getJob()
  %>" /></td>
52      </tr>
53      <tr>
54          <%java.sql.Date date_SQL = new
  java.sql.Date(System.currentTimeMillis());%>
55          <td>雇用日期:</td>
56          <td bgcolor="#CCCCFF">
57              <input class="cal-TextBox"
58              onFocus="this.blur()" size="9" readonly type="text"
  name="hiredate" value="<%= (empVO==null)? date_SQL :
  empVO.getHiredate()%>">
59              <a class="so-BtnLink"
60              href="javascript:calClick();return false;"
61              onmouseover="calSwapImg('BTN_date',
  'img_Date_OVER',true);"
62              onmouseout="calSwapImg('BTN_date', 'img_Date_UP',true);"
63              onclick="calSwapImg('BTN_date',
  'img_Date_DOWN');showCalendar('form1','hiredate','BTN_date');return
  false;">
64              <img align="middle" border="0" name="BTN_date"
  src="images/btn_date_up.gif" width="22" height="17" alt="開始日期"></
  a>
65          </td>
66      </tr>
67      <tr>
68          <td>薪水:</td>
69          <td><input type="TEXT" name="sal" size="45"
70              value="<%= (empVO==null)? "10000" : empVO.getSal()%>" /
  ></td>
71      </tr>
72      <tr>
73          <td>獎金:</td>
74          <td><input type="TEXT" name="comm" size="45"
75              value="<%= (empVO==null)? "100" : empVO.getComm()%>" /
  ></td>
```

```
76      </tr>
77
78      <jsp:useBean id="deptSvc" scope="page"
  class="com.dept.model.DeptService" />
79      <tr>
80          <td>部門:<font color=red><b>*</b></font></td>
81          <td><select size="1" name="deptno">
82              <c:forEach var="deptVO" items="${deptSvc.all}">
83                  <option value="${deptVO.deptno}" $
  {(empVO.deptno==deptVO.deptno)? 'selected':'' } >${deptVO.dname}
84              </c:forEach>
85          </select></td>
86      </tr>
87
88  </table>
89  <br>
90  <input type="hidden" name="action" value="insert">
91  <input type="submit" value="送出新增"></FORM>
92  </body>
93
94  </html>
95
```

```jsp
<%@ page contentType="text/html; charset=UTF-8"
pageEncoding="Big5"%>
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>
<%@ page import="java.util.*"%>
<%@ page import="com.emp.model.*"%>
<%-- 此頁練習採用 EL 的寫法取值 --%>

<%
    EmpService empSvc = new EmpService();
    List<EmpVO> list = empSvc.getAll();
    pageContext.setAttribute("list",list);
%>

<html>
<head>
<title>所有員工資料 - listAllEmp.jsp</title>
</head>
<body bgcolor='white'>
<b><font color=red>此頁練習採用 EL 的寫法取值:</font></b>
<table border='1' cellpadding='5' cellspacing='0' width='800'>
    <tr bgcolor='#CCCCFF' align='center' valign='middle'
height='20'>
        <td>
        <h3>所有員工資料 - ListAllEmp.jsp</h3>
        <a href="select_page.jsp"><img src="images/back1.gif"
width="100" height="32" border="0">回首頁</a>
        </td>
    </tr>
</table>

<%-- 錯誤表列 --%>
<c:if test="${not empty errorMsgs}">
    <font color='red'>請修正以下錯誤:
    <ul>
        <c:forEach var="message" items="${errorMsgs}">
            <li>${message}</li>
        </c:forEach>
    </ul>
    </font>
</c:if>

<table border='1' bordercolor='#CCCCFF' width='800'>
    <tr>
        <th>員工編號</th>
        <th>員工姓名</th>
```

```
43          <th>職位</th>
44          <th>雇用日期</th>
45          <th>薪水</th>
46          <th>獎金</th>
47          <th>部門</th>
48          <th>修改</th>
49          <th>刪除</th>
50      </tr>
51      <%@ include file="page1.file" %>
52      <c:forEach var="empVO" items="${list}" begin="<%=pageIndex%>"
  end="<%=pageIndex+rowsPerPage-1%>">
53          <tr align='center' valign='middle'>
54              <td>${empVO.empno}</td>
55              <td>${empVO.ename}</td>
56              <td>${empVO.job}</td>
57              <td>${empVO.hiredate}</td>
58              <td>${empVO.sal}</td>
59              <td>${empVO.comm}</td>
60              <td>${empVO.deptno}</td>
61              <td>
62                <FORM METHOD="post" ACTION="<
  %=request.getContextPath()%>/emp/emp.do">
63                  <input type="submit" value="修改">
64                  <input type="hidden" name="empno" value="$
  {empVO.empno}">
65                  <input type="hidden" name="action"
  value="getOne_For_Update"></FORM>
66              </td>
67              <td>
68                <FORM METHOD="post" ACTION="<
  %=request.getContextPath()%>/emp/emp.do">
69                  <input type="submit" value="刪除">
70                  <input type="hidden" name="empno" value="$
  {empVO.empno}">
71                  <input type="hidden" name="action"value="delete"></
  FORM>
72              </td>
73          </tr>
74      </c:forEach>
75  </table>
76  <%@ include file="page2.file" %>
77
78  </body>
79  </html>
80
```

```jsp
<%@ page contentType="text/html; charset=Big5"%>
<%@ page import="com.emp.model.*"%>
<%
EmpVO empVO = (EmpVO) request.getAttribute("empVO"); //
EmpServlet.java(Concroller), 存入req的empVO物件
%>
<html>
<head>
<title>員工資料 - listOneEmp.jsp</title>
</head>
<body bgcolor='white'>

<table border='1' cellpadding='5' cellspacing='0' width='600'>
    <tr bgcolor='#CCCCFF' align='center' valign='middle'
height='20'>
        <td>
        <h3>員工資料 - ListOneEmp.jsp</h3>
        <a href="select_page.jsp"><img src="images/back1.gif"
width="100" height="32" border="0">回首頁</a>
        </td>
    </tr>
</table>

<table border='1' bordercolor='#CCCCFF' width='600'>
    <tr>
        <th>員工編號</th>
        <th>員工姓名</th>
        <th>職位</th>
        <th>雇用日期</th>
        <th>薪水</th>
        <th>獎金</th>
        <th>部門</th>
    </tr>
    <tr align='center' valign='middle'>
        <td><%=empVO.getEmpno()%></td>
        <td><%=empVO.getEname()%></td>
        <td><%=empVO.getJob()%></td>
        <td><%=empVO.getHiredate()%></td>
        <td><%=empVO.getSal()%></td>
        <td><%=empVO.getComm()%></td>
        <td><%=empVO.getDeptno()%></td>
    </tr>
</table>

</body>
```

```
43 </html>
44
```