|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **□ 수행평가 - 빅데이터를 활용한 IoT 시스템 개발(feat.커넥티드카)** | | | | | | |
|  |  |  | |  | |  |
| **과정명** | | 빅데이터를 활용한 IoT 시스템 개발(feat.커넥티드카) | | | | |
| **교과목명** | | IoT 운영시스템 구축 기반기술 | | **훈련교사** | | 이진만 |
| **과정명** | | SW기초기술이해 | | | | |
| **수행날짜** | | 2019.06.11 | 훈련생명 | | 황성민 | |
| **과제개요** | | | | | | |
| 패키지명: jdbc04  1. Workshop 교재 124Page의 테이블을 구축 한다.  2. DML, DDL을 작성하고 Table의 구조를 작성한다.  3. UML을 이용하여 프로그램을 설계한다.  4. JDBC API를 이용하여 상품에 대한 CRUD 프로그램을 작성 한다.  작성 내용  1. ERD    2. DDL, DML  DDL   |  |  |  |  | | --- | --- | --- | --- | | |  |  |  | | --- | --- | --- | | 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20 | CREATE TABLE PRODUCTS(      PDNO NUMBER(10, 0),      PDNAME VARCHAR2(10),      PDSUBNAME VARCHAR2(10),      FACTNO VARCHAR2(5),      PDDATE DATE,      PDCOST NUMBER(10, 0),      PDPRICE NUMBER(10, 0),      PDAMOUNT NUMBER(10, 0)  );    CREATE TABLE FACTORY(      FACTNO VARCHAR2(5),      FACNAME VARCHAR2(14),      FACLOC VARCHAR2(13)  );  ALTER TABLE PRODUCTS ADD PRIMARY KEY(PDNO);  ALTER TABLE FACTORY ADD PRIMARY KEY(FACTNO);  ALTER TABLE PRODUCTS ADD FOREIGN KEY(FACTNO) REFERENCES FACTORY(FACTNO);    [*Colored by Color Scripter*](http://colorscripter.com/info#e) | [cs](http://colorscripter.com/info#e) | |   DML   |  |  |  |  | | --- | --- | --- | --- | | |  |  |  | | --- | --- | --- | | 1  2  3  4  5  6  7  8  9  10  11  12  13 | INSERT INTO FACTORY VALUES('A01', 'SSCOM', 'SEOUL');  INSERT INTO FACTORY VALUES('A03', 'VEGA', 'GUMI');  INSERT INTO FACTORY VALUES('C02', 'APPLE', 'CHINA');  INSERT INTO PRODUCTS VALUES (195020, 'PHONE', 'GALAXY','A01',                               TO\_DATE('2019-05-30','YYYY-MM-DD'), 30, 103, 3);  INSERT INTO PRODUCTS VALUES (195030, 'PHONE', 'IPHONE', 'C02',                               TO\_DATE('2019-05-30','YYYY-MM-DD'), 25, 102, 5);  INSERT INTO PRODUCTS VALUES (195040, 'PHONE', 'V50', 'A03',                               TO\_DATE('2019-05-30','YYYY-MM-DD'), 20, 88, 10);  INSERT INTO PRODUCTS VALUES (197501, 'TV', 'OLEDTV', 'A03',                               TO\_DATE('2019-06-02','YYYY-MM-DD'), 100, 290, 8);  INSERT INTO PRODUCTS VALUES (197505, 'TV', 'QLEDTV', 'A01',                               TO\_DATE('2019-06-04','YYYY-MM-DD'), 101, 300, 10); | [cs](http://colorscripter.com/info#e) | |   3. UML    4. 작성 코드  vo   |  |  |  |  | | --- | --- | --- | --- | | |  |  |  | | --- | --- | --- | | 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90  91  92  93  94  95  96  97  98  99  100  101  102  103  104  105  106  107  108 | package vo;    import java.sql.Date;    public class Products {        int pdno;      String pdname;      String pdsubname;      String factno;      Date pddate;      int pdcost;      int pdprice;      int pdamount;        public Products()      {      }        public Products(int pdno, String pdname, String pdsubname, String factno,              Date pddate, int pdcost, int pdprice, int pdamount) {          this.pdno = pdno;          this.pdname = pdname;          this.pdsubname = pdsubname;          this.factno = factno;          this.pddate = pddate;          this.pdcost = pdcost;          this.pdprice = pdprice;          this.pdamount = pdamount;      }        @Override      public String toString() {          return "Products [pdno=" + pdno + ", pdname=" + pdname + ", pdsubname=" + pdsubname                  + ", factno=" + factno + ", pddate=" + pddate + ", pdcost=" + pdcost                  + ", pdprice=" + pdprice + ", pdamount=" + pdamount + "]";      }        public String toStringd() {           return "[pdno=" + pdno + ", pdname=" + pdname + ", pdsubname=" + pdsubname                   + ", pddate=" + pddate + ", pdcost=" + pdcost + ", pdprice=" + pdprice                   + ", pdamount=" + pdamount;      }        public int getPdno() {          return pdno;      }        public void setPdno(int pdno) {          this.pdno = pdno;      }        public String getPdname() {          return pdname;      }        public void setPdname(String pdname) {          this.pdname = pdname;      }        public String getPdsubname() {          return pdsubname;      }        public void setPdsubname(String pdsubname) {          this.pdsubname = pdsubname;      }        public String getFactno() {          return factno;      }        public void setFactno(String factno) {          this.factno = factno;      }        public Date getPddate() {          return pddate;      }        public void setPddate(Date pddate) {          this.pddate = pddate;      }        public int getPdcost() {          return pdcost;      }        public void setPdcost(int pdcost) {          this.pdcost = pdcost;      }        public int getPdprice() {          return pdprice;      }        public void setPdprice(int pdprice) {          this.pdprice = pdprice;      }        public int getPdamount() {          return pdamount;      }        public void setPdamount(int pdamount) {          this.pdamount = pdamount;      }  }  [*Colored by Color Scripter*](http://colorscripter.com/info#e) | [cs](http://colorscripter.com/info#e) | | | |  |  |  | | --- | --- | --- | | 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51 | package vo;    public class Factory {        String factno;      String facname;      String facloc;        public Factory() {      }        public Factory(String factno, String facname, String facloc) {          this.factno = factno;          this.facname = facname;          this.facloc = facloc;      }        @Override      public String toString() {          return "Factory [factno=" + factno + ", facname=" + facname + ", facloc=" + facloc + "]";      }        public String toStringd() {           return ", factno=" + factno + ", facname=" + facname + ", facloc=" + facloc+" ]" ;      }        public String getFactno() {          return factno;      }        public void setFactno(String factno) {          this.factno = factno;      }        public String getFacname() {          return facname;      }        public void setFacname(String facname) {          this.facname = facname;      }        public String getFacloc() {          return facloc;      }        public void setFacloc(String facloc) {          this.facloc = facloc;      }  }    [*Colored by Color Scripter*](http://colorscripter.com/info#e) | [cs](http://colorscripter.com/info#e) | |   frame   |  |  |  |  | | --- | --- | --- | --- | | |  |  |  | | --- | --- | --- | | 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59 | package frame;    import java.sql.Connection;  import java.sql.DriverManager;  import java.sql.SQLException;  import java.util.ArrayList;    public abstract class Biz<K,V> {        String id;      String pwd;      String url;        public Biz() {          // 1. JDBC Driver Loading..          try {              Class.forName("oracle.jdbc.driver.OracleDriver");          } catch (ClassNotFoundException e) {              System.out.println("Driver Loading Error");          }          id = "db";          pwd = "db";          url = "jdbc:oracle:thin:@70.12.50.227:1521:xe";      }        public Connection getCon() {          Connection con = null;          try {              con =              DriverManager.getConnection(url,id,pwd);              con.setAutoCommit(false);          } catch (SQLException e) {              e.printStackTrace();          }          return con;      }        public void close(Connection con) {          if(con != null) {              try {                  con.close();              } catch (SQLException e) {                  e.printStackTrace();              }          }      }        public abstract void register(V v) throws Exception;        public abstract void remove(K k) throws Exception;        public abstract void modify(V v) throws Exception;        public abstract V get(K k) throws Exception;        public abstract ArrayList<V> get() throws Exception;    }    [*Colored by Color Scripter*](http://colorscripter.com/info#e) | [cs](http://colorscripter.com/info#e) | | | |  |  |  | | --- | --- | --- | | 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41 | package frame;    import java.sql.Connection;  import java.sql.PreparedStatement;  import java.sql.ResultSet;  import java.sql.SQLException;  import java.util.ArrayList;    public abstract class Dao<K, V> {          public void close(PreparedStatement con) {          if(con != null) {              try {                  con.close();              } catch (SQLException e) {                  e.printStackTrace();              }          }      }      public void close(ResultSet con) {          if(con != null) {              try {                  con.close();              } catch (SQLException e) {                  e.printStackTrace();              }          }      }        public abstract void insert(V v,Connection con) throws Exception;        public abstract void delete(K k,Connection con) throws Exception;        public abstract void update(V v,Connection con) throws Exception;        public abstract V select(K k,Connection con) throws Exception;        public abstract ArrayList<V> select(Connection con) throws Exception;  }    [*Colored by Color Scripter*](http://colorscripter.com/info#e) | [cs](http://colorscripter.com/info#e) | | | |  |  |  | | --- | --- | --- | | 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49 | package frame;    import java.util.ArrayList;    import com.FactoryBiz;  import com.ProductsBiz;    import vo.Factory;  import vo.Products;    public class func {        public static void join(String k) {          Products p = null;          Factory f = null;          Biz<String, Products> dao = new ProductsBiz();          Biz<String, Factory> dao1 = new FactoryBiz();          try {                p = dao.get(k);              f = dao1.get(p.getFactno());              System.out.println("[pdno=" + p.getPdno() + ", pdname=" + p.getPdname()              + ", pdsubname=" + p.getPdsubname() + ", factno=" + p.getFactno()              + ", pddate=" + p.getPddate() + ", pdcost=" + p.getPdcost()              + ", pdprice=" + p.getPdprice() + ", pdamount=" + p.getPdamount()              + ", facname=" + f.getFacname() + ", facloc=" + f.getFacloc() + "]");            } catch (Exception e) {              e.printStackTrace();          }      }        public static void joinall() {          Biz<String, Products> dao = new ProductsBiz();          Biz<String, Factory> dao1 = new FactoryBiz();          ArrayList<Products> p = new ArrayList<>();          Factory fac = null;          try {              p = dao.get();              for (int i = 0; i < p.size(); i++) {                  fac = dao1.get((p.get(i).getFactno()));                  System.out.println(p.get(i).toStringd() + fac.toStringd());              }          } catch (Exception e) {              e.printStackTrace();          }      }  }    [*Colored by Color Scripter*](http://colorscripter.com/info#e) | [cs](http://colorscripter.com/info#e) | | | |  |  |  | | --- | --- | --- | | 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34 | package frame;    public class Sql {      public static String insertProduct =              "INSERT INTO PRODUCTS VALUES(?,?,?,?,?,?,?,?)";      public static String DeleteProduct =              "DELETE FROM PRODUCTS WHERE PDNO = ?";      public static String UpdateProduct =              "Update PRODUCTS SET PDNAME=?, PDSUBNAME=?, FACTNO=?, PDDATE=?, "              + "PDCOST=?, PDPRICE=?, PDAMOUNT = ?,  WHERE PDNO=?";      public static String SelectProduct =              "SELECT \* FROM PRODUCTS WHERE PDNO = ?";      public static String SelectAllProduct =              "SELECT \* FROM PRODUCTS";          public static String insertFactory =              "INSERT INTO FACTORY VALUES(?,?,?)";      public static String DeleteFactory =              "DELETE FROM FACTORY WHERE FACTNO = ?";      public static String UpdateFactory =              "Update FACTORY SET FACNAME =?, FACLOC=?, WHERE FACTNO=?";      public static String SelectFactory =              "SELECT \* FROM FACTORY WHERE FACTNO = ?";      public static String SelectAllFactory =              "SELECT \* FROM FACTORY";          public static String join =              "SELECT p.PDNO, p.PDNAME, p.PDSUBNAME, f.FACTNO, f.FACNAME, "              + "f.FACLOC FROM PRODUCTS p, FACTORY f\r\n"              + "WHERE p.FACTNO = f.FACTNO";  }    [*Colored by Color Scripter*](http://colorscripter.com/info#e) | [cs](http://colorscripter.com/info#e) | |   com   |  |  |  |  | | --- | --- | --- | --- | | |  |  |  | | --- | --- | --- | | 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90  91  92  93 | package com;    import java.sql.Connection;  import java.util.ArrayList;    import frame.Biz;  import frame.Dao;  import vo.Factory;    public class FactoryBiz extends Biz<String, Factory> {        Dao<String, Factory> dao;      public FactoryBiz() {          dao = new FactoryDao();      }        @Override      public void register(Factory v) throws Exception {          Connection con = null;          try {              con = getCon();              dao.insert(v, con);              con.commit();          } catch (Exception e) {              con.rollback();              throw e;          } finally {              close(con);          }      }        @Override      public void remove(String k) throws Exception {          Connection con = null;          try {              con = getCon();              dao.delete(k, con);              con.commit();          } catch (Exception e) {              con.rollback();              throw e;          } finally {              close(con);          }      }        @Override      public void modify(Factory v) throws Exception {          Connection con = null;          try {              con = getCon();              dao.update(v, con);              con.commit();          } catch (Exception e) {              con.rollback();              throw e;          } finally {              close(con);          }      }        @Override      public Factory get(String k) throws Exception {          Connection con = null;          Factory fac = null;          try {              con = getCon();              fac = dao.select(k, con);          } catch (Exception e) {              throw e;          } finally {              close(con);          }          return fac;      }        @Override      public ArrayList<Factory> get() throws Exception {          Connection con = null;          ArrayList<Factory> fac = null;          try {              con = getCon();              fac = dao.select(con);          } catch (Exception e) {              throw e;          } finally {              close(con);          }          return fac;      }    }    [*Colored by Color Scripter*](http://colorscripter.com/info#e) | [cs](http://colorscripter.com/info#e) | | | |  |  |  | | --- | --- | --- | | 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90  91  92  93  94  95  96  97  98  99  100  101  102  103  104  105  106  107  108  109  110  111  112 | package com;    import java.sql.Connection;  import java.sql.PreparedStatement;  import java.sql.ResultSet;  import java.util.ArrayList;    import frame.Dao;  import frame.Sql;  import vo.Factory;    public class FactoryDao extends Dao<String, Factory> {        @Override      public void insert(Factory v, Connection con) throws Exception {          PreparedStatement pstmt = null;          try {              pstmt = con.prepareStatement(Sql.insertFactory);              pstmt.setString(1, v.getFactno());              pstmt.setString(2, v.getFacname());              pstmt.setString(3, v.getFacloc());              pstmt.executeUpdate();          }catch(Exception e) {              throw e;          }finally {              close(pstmt);          }        }        @Override      public void delete(String k, Connection con) throws Exception {          PreparedStatement pstmt = null;          try {          pstmt = con.prepareStatement(Sql.DeleteFactory);          pstmt.setString(1, k);          pstmt.executeUpdate();          }catch(Exception e) {              throw e;          }finally {              close(pstmt);          }      }        @Override      public void update(Factory v, Connection con) throws Exception {          PreparedStatement pstmt = null;          try {          pstmt = con.prepareStatement(Sql.UpdateFactory);          pstmt.setString(1, v.getFacname());          pstmt.setString(2, v.getFacloc());          pstmt.setString(3, v.getFactno());          pstmt.executeUpdate();          }catch(Exception e) {              throw e;          }finally {              close(pstmt);          }        }        @Override      public Factory select(String k, Connection con) throws Exception {          Factory f = null;          ResultSet rset=null;          PreparedStatement pstmt = null;          try {          pstmt = con.prepareStatement(Sql.SelectFactory);          pstmt.setString(1, k);          rset = pstmt.executeQuery();          rset.next();          String factno = rset.getString("FACTNO");          String factname = rset.getString("FACNAME");          String facloc = rset.getString("FACLOC");          f = new Factory(factno, factname, facloc);          return f;          }catch(Exception e) {              throw e;          }finally {              close(pstmt);              close(rset);          }      }        @Override      public ArrayList<Factory> select(Connection con) throws Exception {          Factory f = null;          ArrayList<Factory> fs = new ArrayList<Factory>();          ResultSet rset=null;          PreparedStatement pstmt = null;          try {          pstmt = con.prepareStatement(Sql.SelectAllFactory);          rset = pstmt.executeQuery();          while(rset.next()) {              String factno = rset.getString("FACTNO");              String factname = rset.getString("FACNAME");              String facloc = rset.getString("FACLOC");              System.out.println(factno+" "+factname+" "+facloc);              f = new Factory(factno,factname, facloc);              fs.add(f);          }          return fs;          }catch(Exception e) {              throw e;          }finally {              close(pstmt);              close(rset);          }      }    }    [*Colored by Color Scripter*](http://colorscripter.com/info#e) | [cs](http://colorscripter.com/info#e) | | | |  |  |  | | --- | --- | --- | | 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90  91  92  93  94  95  96  97  98  99  100  101  102  103  104  105  106  107  108  109 | package com;    import java.sql.Connection;  import java.util.ArrayList;    import frame.Biz;  import frame.Dao;  import vo.Products;    public class ProductsBiz extends Biz<String, Products> {        Dao<String, Products> dao;        public ProductsBiz() {          dao = new ProductsDao();      }        @Override      public void register(Products v) throws Exception {          Connection con = null;          try {              con = getCon();              dao.insert(v, con);                con.commit();          } catch (Exception e) {              con.rollback();              throw e;          } finally {              close(con);          }      }        @Override      public void remove(String k) throws Exception {          Connection con = null;          try {              con = getCon();              dao.delete(k, con);              con.commit();          } catch (Exception e) {              con.rollback();              throw e;          } finally {              close(con);          }      }        @Override      public void modify(Products v) throws Exception {          Connection con = null;          try {              con = getCon();              dao.update(v, con);              con.commit();          } catch (Exception e) {              con.rollback();              throw e;          } finally {              close(con);          }      }        @Override      public Products get(String k) throws Exception {          Connection con = null;          Products u = null;          try {              con = getCon();              u = dao.select(k, con);          } catch (Exception e) {              throw e;          } finally {              close(con);          }          return u;      }        @Override      public ArrayList<Products> get() throws Exception {          Connection con = null;          ArrayList<Products> pro = null;          try {              con = getCon();              pro = dao.select(con);          } catch (Exception e) {              throw e;          } finally {              close(con);          }          return pro;      }        // 조인함수      public Products join(String k) throws Exception {          Connection con = null;          Products u = null;          try {              con = getCon();              u = dao.select(k, con);          } catch (Exception e) {              throw e;          } finally {              close(con);          }          return u;      }  }    [*Colored by Color Scripter*](http://colorscripter.com/info#e) | [cs](http://colorscripter.com/info#e) | | | |  |  |  | | --- | --- | --- | | 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90  91  92  93  94  95  96  97  98  99  100  101  102  103  104  105  106  107  108  109  110  111  112  113  114  115  116  117  118  119  120  121  122  123  124  125  126 | package com;    import java.sql.Connection;  import java.sql.Date;  import java.sql.PreparedStatement;  import java.sql.ResultSet;  import java.sql.SQLException;  import java.util.ArrayList;    import frame.Dao;  import frame.Sql;  import vo.Products;    public class ProductsDao extends Dao<String, Products> {        public void insert(Products v, Connection con) throws Exception {          PreparedStatement pstmt = null;          try {              pstmt = con.prepareStatement(Sql.insertProduct);              pstmt.setInt(1, v.getPdno());              pstmt.setString(2, v.getPdname());              pstmt.setString(3, v.getPdsubname());              pstmt.setString(4, v.getFactno());              pstmt.setDate(5, v.getPddate());              pstmt.setInt(6, v.getPdcost());              pstmt.setInt(7, v.getPdprice());              pstmt.setInt(8, v.getPdamount());              pstmt.executeUpdate();          } catch (Exception e) {              throw e;          } finally {              close(pstmt);          }      }        @Override      public void delete(String k, Connection con) throws Exception {          PreparedStatement pstmt = null;          try {              pstmt = con.prepareStatement(Sql.DeleteProduct);              pstmt.setString(1, k);              int result = pstmt.executeUpdate();              System.out.println(result);          } catch (SQLException e) {              throw e;          } finally {              close(pstmt);          }      }        @Override      public void update(Products v, Connection con) throws Exception {          PreparedStatement pstmt = null;          try {              pstmt = con.prepareStatement(Sql.UpdateProduct);              pstmt.setInt(1, v.getPdno());              pstmt.setString(2, v.getPdname());              pstmt.setString(3, v.getPdsubname());              pstmt.setString(3, v.getFactno());              pstmt.setDate(4, v.getPddate());              pstmt.setInt(5, v.getPdcost());              pstmt.setInt(6, v.getPdprice());              pstmt.setInt(7, v.getPdamount());              pstmt.executeUpdate();          } catch (SQLException e) {              throw e;          } finally {              close(pstmt);          }      }        @Override      public Products select(String k, Connection con) throws Exception {          PreparedStatement pstmt = null;          ResultSet rset = null;          Products pro;          try {              pstmt = con.prepareStatement(Sql.SelectProduct);              pstmt.setString(1, k);              rset = pstmt.executeQuery();              rset.next();              int pno = rset.getInt("PDNO");              String pname = rset.getString("PDNAME");              String psubname = rset.getString("PDSUBNAME");              String fno = rset.getString("FACTNO");              Date pdate = rset.getDate("PDDATE");              int pcost = rset.getInt("PDCOST");              int pprice = rset.getInt("PDPRICE");              int pamount = rset.getInt("PDAMOUNT");              pro = new Products(pno, pname, psubname, fno, pdate, pcost, pprice, pamount);          } catch (SQLException e) {              throw e;          } finally {              close(pstmt);          }          return pro;      }        @Override      public ArrayList<Products> select(Connection con) throws Exception {          ArrayList<Products> pro = new ArrayList<>();          PreparedStatement pstmt = null;          ResultSet rset = null;          try {              pstmt = con.prepareStatement(Sql.SelectAllProduct);              rset = pstmt.executeQuery();              while (rset.next()) {                  int pno = rset.getInt("PDNO");                  String pname = rset.getString("PDNAME");                  String psubname = rset.getString("PDSUBNAME");                  String fno = rset.getString("FACTNO");                  Date pdate = rset.getDate("PDDATE");                  int pcost = rset.getInt("PDCOST");                  int pprice = rset.getInt("PDPRICE");                  int pamount = rset.getInt("PDAMOUNT");                  pro.add(new Products(pno, pname, psubname, fno, pdate, pcost, pprice, pamount));              }          } catch (SQLException e) {              throw e;          } finally {              close(pstmt);          }          return pro;      }  }    [*Colored by Color Scripter*](http://colorscripter.com/info#e) | [cs](http://colorscripter.com/info#e) | |   test   |  |  |  |  | | --- | --- | --- | --- | | |  |  |  | | --- | --- | --- | | 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32 | package test;    import java.util.ArrayList;    import com.FactoryBiz;  import com.ProductsBiz;  import com.ProductsDao;    import frame.Biz;  import frame.Dao;  import vo.Factory;  import vo.Products;    public class test {        public static void main(String[] args) {              ArrayList<Factory> fac = new ArrayList<Factory>();          Factory f = new Factory("A04", "SONY", "JAPAN");          Biz<String, Factory> dao = new FactoryBiz();          try {                f = dao.get("A01");              System.out.println(f);            } catch (Exception e) {              e.printStackTrace();          }      }  }    [*Colored by Color Scripter*](http://colorscripter.com/info#e) | [cs](http://colorscripter.com/info#e) | | | |  |  |  | | --- | --- | --- | | 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26 | package test;    import java.util.ArrayList;    import com.ProductsBiz;    import frame.Biz;  import vo.Products;    public class test1 {        public static void main(String[] args) {          ArrayList<Products> pro = new ArrayList<Products>();          Products p = new Products(33, "BB", "CC", "A01", null, 10, 10, 10);          Biz<String, Products> dao = new ProductsBiz();          try {                p = dao.get("197505");              System.out.println(p);            } catch (Exception e) {              e.printStackTrace();          }      }  }    [*Colored by Color Scripter*](http://colorscripter.com/info#e) | [cs](http://colorscripter.com/info#e) | | | | | | | | |