



---

# 데이터베이스실무 과제

---

UPDATE, DELETE 기능 구현



2501110203

김호석

2025.10.31



# 목차

<b>1. 과제 개요 .....</b>	<b>1</b>
1-1. 과제 개요.....	1
1-2. 과제 수행 목표 .....	1
<b>2. 코드.....</b>	<b>2</b>
3-1. Controller Package .....	2
3-2. Entity Package .....	14
3-3. View Package .....	16
<b>3. 결과 및 분석 .....</b>	<b>24</b>
3-1. 결과.....	24
3-2. 분석.....	32

# 1. 과제 개요

## 1-1. 과제 개요

1. mvc\_db\_test 패키지의 클래스를 사용합니다.
2. 고객테이블의 저장된 고객 정보를 수정하는 기능 추가
  - 아래의 진행 순서를 따를 것
    1. 업데이트할 고객 아이디 입력
    2. 해당 고객의 정보 출력
    3. 수정할 데이터 선택 (ID는 변경 불가)
    4. 수정할 데이터 입력
    5. 데이터 UPDATE
    6. 수정된 고객 정보 출력
  - 3. 고객테이블의 저장된 고객 정보를 삭제하는 기능 추가
    - 고객 id를 입력 받을 것
    - 삭제할 것인지 확인 절차를 진행할 것

## 1-2. 과제 수행 목표

1. 조회, 추가, 수정, 삭제 기능을 모두 이용할 수 있도록 하고자 함.
2. Switch Case 문을 사용하여 각 기능을 분기별로 관리하고자 함.
3. 고객 테이블 뿐만 아니라 제품, 주문 테이블도 접근할 수 있도록 하고자 함.

## 2. 코드

### GitHub Link

[https://github.com/WaltDev29/Course\\_SQL/tree/main/SQL/src/mvc\\_jdbc\\_test](https://github.com/WaltDev29/Course_SQL/tree/main/SQL/src/mvc_jdbc_test)

### 2-1. Controller Package

#### 2-1-1. MainController Class2 (MainController Class 를 제작 후 리팩토링 함.)

```
1 package mvc_jdbc_test.controller;
2
3 > import ...
4
5 > public class MainController2 { Waltdev29
6     public static void main(String[] args) { Waltdev29
7         // 변수 선언
8         Scanner sc = new Scanner( source: System.in );
9         Connection con;
10        MainView mv = new MainView();
11
12        // 결과 담을 List 선언
13        ArrayList<Customer> customerList;
14        ArrayList<Product> productList;
15        ArrayList<Order> orderList;
16
17        // JDBC 연결
18        con = JDBCConnector.getConnection();
19
20        int mainState = 0;
21        int subState;
22        String mode;
23        String pk = "";
24        boolean proceed;
25        // 예외 프로그램
26        while (true) {
27            // Mode 선택
28            if (mainState == 0) { // 예외
29                mv.showHomeView();
30                mainState = mv.inputAnswer( sc: sc, min: 0, max: 4);
31                if (mainState == 0) break;
32            }
33
34            // 잘못된 입력
35            if (mainState < 0 || mainState > 4) {
36                System.out.println("잘못된 입력입니다. 1~3의 정수를 입력해주세요.");
37            }
38
39
40
41
42
43
44
45
46
47
```

```
48 // Mode 설정
49 mode = switch (mainState) {
50     case 1 -> "조회";
51     case 2 -> "추가";
52     case 3 -> "수정";
53     case 4 -> "삭제";
54     default -> "";
55 };
56
57 // DB 선택
58 mv.showMainView(mode);
59 subState = mv.inputAnswer(sc: sc, min: 0, max: 3);
60
61 // 처음으로 돌아가기
62 if (subState == 0) {
63     mainState = 0;
64     continue;
65 }
66
67
68 // 데이터 추가
69 if (mainState == 2) {
70     if (subState == 1) {
71         insertCustomerInfo(con, sc: sc);
72     } else if (subState == 2) {
73         insertProductInfo(con, sc: sc);
74     } else if (subState == 3) {
75         insertOrderInfo(con, sc: sc);
76     }
77     mv.inputEnter(sc: sc);
78     continue;
79 }
80
```

```

81     // DB 출력
82     if (subState == 1) {
83         customerList = getCustomerList(con, target: null);
84         printItemList( itemList: customerList, view: new CustomerView());
85     } else if (subState == 2) {
86         productList = getProductList(con, target: null);
87         printItemList( itemList: productList, view: new ProductView());
88     } else if (subState == 3) {
89         orderList = getOrderList(con, target: null);
90         printItemList( itemList: orderList, view: new OrderView());
91     }
92
93
94     // 데이터 조회일 경우 바로 처음으로
95     if (mainState == 1) {
96         mv.inputEnter( sc: sc);
97         continue;
98     }
99
100
101    // PK 입력
102    if (subState == 1) {
103        pk = inputCustomerPk(con, sc: sc);
104        customerList = getCustomerList(con, target: pk);
105        if (mainState == 3) printItemWithIndex( item: customerList.get(0), view: new CustomerView());
106        else printItem( item: customerList.get(0), view: new CustomerView());
107    } else if (subState == 2) {
108        pk = inputProductPk(con, sc: sc);
109        productList = getProductList(con, target: pk);
110        if (mainState == 3) printItemWithIndex( item: productList.get(0), view: new ProductView());
111        else printItem( item: productList.get(0), view: new ProductView());
112    } else if (subState == 3) {
113        pk = inputOrderPk(con, sc: sc);
114        orderList = getOrderList(con, target: pk);
115        if (mainState == 3) printItemWithIndex( item: orderList.get(0), view: new OrderView());
116        else printItem( item: orderList.get(0), view: new OrderView());
117    }
118

```

```

119         // 삭제
120     if (_mainState == 4) {
121         System.out.println("\n해당 데이터를 삭제하시겠습니까? ");
122         if (mv.askYorN(_sc, _yesText: "삭제", _yesChar: "Y", _noText: "취소", _noChar: "N")) {
123             deleteInfo(con, _subState, target: pk);
124             mv.inputEnter(_sc: sc);
125         }
126     }
127
128     // 데이터 수정
129     if (_mainState == 3) {
130         System.out.println("\n수정할 항목의 번호를 입력해주세요. 뒤로가기 : 0");
131         _proceed = updateInfo(con, _sc: sc, _subState, target: pk);
132         if (!_proceed) continue;
133
134         System.out.println("\n수정이 완료되었습니다.");
135         System.out.print("\n--- 수정 정보 ---");
136         if (_subState == 1) {
137             _customerList = getCustomerList(con, target: pk);
138             printItemWithIndex(item: _customerList.get(0), view: new CustomerView());
139         } else if (_subState == 2) {
140             _productList = getProductList(con, target: pk);
141             printItem(item: _productList.get(0), view: new ProductView());
142         } else if (_subState == 3) {
143             _orderList = getOrderList(con, target: pk);
144             printItem(item: _orderList.get(0), view: new OrderView());
145         }
146         mv.inputEnter(_sc: sc);
147     }
148
149     System.out.println("\n프로그램을 종료합니다.");
150     sc.close();
151 }
152
153
154

```

```

155 // Input
156 @
157 private static String inputCustomerPk(Connection con, @NotNull Scanner sc) { 1개 사용 위치 ☰ Waltdev29
158     ArrayList<Customer> customerList = getCustomerList(con, target: null);
159     String target;
160     System.out.println("고객의 고객 아이디를 입력하세요.\n");
161     while (true) {
162         System.out.print("고객 아이디 : ");
163         target = sc.nextLine();
164         if (validatePK( EntityList: customerList, pk: target)) break;
165         else System.out.println("존재하지 않는 고객 아이디입니다. 다시 입력해주세요.\n");
166     }
167     return target;
168 }
169 @
170 private static String inputProductPk(Connection con, @NotNull Scanner sc) { 1개 사용 위치 ☰ Waltdev29
171     ArrayList<Product> productList = getProductList(con, target: null);
172     String target;
173     System.out.println("제품의 제품번호를 입력하세요.\n");
174     while (true) {
175         System.out.print("제품번호 : ");
176         target = sc.nextLine();
177         if (validatePK( EntityList: productList, pk: target)) break;
178         else System.out.println("존재하지 않는 제품입니다. 다시 입력해주세요.\n");
179     }
180     return target;
181 }
182 @
183 private static String inputOrderPk(Connection con, @NotNull Scanner sc) { 1개 사용 위치 ☰ Waltdev29
184     ArrayList<Order> orderList = getOrderList(con, target: null);
185     String target;
186     System.out.println("주문의 주문번호를 입력하세요.\n");
187     while (true) {
188         System.out.print("주문번호 : ");
189         target = sc.nextLine();
190         if (validatePK( EntityList: orderList, pk: target)) break;
191         else System.out.println("존재하지 않는 주문번호입니다. 다시 입력해주세요.\n");
192     }
193     return target;
194 }
195

```

```

196 // Validate
197 @
198     private static boolean validatePk( @NotNull ArrayList<? extends Entity> EntityList, String pk) { 3개 사용 위치 ↳ Waltdev28
199         for (Entity e : EntityList) if (e.getId().equalsIgnoreCase( anotherString: pk)) return true;
200         return false;
201     }
202
203 // Print
204 @
205     private static <T> void printItem(T item, @NotNull ObjectView<T> view) { 5개 사용 위치 ↳ Waltdev29
206         view.printHead();
207         view.printCols();
208         view.printItem(item);
209         System.out.println();
210     }
211
212     private static <T> void printItemList( @NotNull ArrayList<T> itemList, @NotNull ObjectView<T> view) { 6개 사용 위치 ↳ Wa
213         view.printHead();
214         view.printCols();
215         for (T item : itemList) {
216             view.printItem(item);
217             System.out.println();
218         }
219         view.printFoot();
220     }
221
222     private static <T> void printItemWithIndex(T item, @NotNull ObjectView<T> view) { 4개 사용 위치 ↳ Waltdev29
223         view.printItemWithIndex(item);
224     }
225
226 // SELECT
227 @
228     @NotNull
229     private static ArrayList<Customer> getCustomerList(Connection con, String target) { 4개 사용 위치 ↳ Waltdev29
230         ArrayList<Customer> customerList = new ArrayList<>();
231         Customer customer;
232         String sql;
233
234         if (target != null) sql = "SELECT * FROM 고객 WHERE 고객아이디 = ?";
235         else sql = "SELECT * FROM 고객";
236         try {
237             PreparedStatement ps = con.prepareStatement(sql);
238             if (target != null) ps.setString( parameterIndex: 1, target);
239             ResultSet rs = ps.executeQuery();
240
241             while (rs.next()) {
242                 customer = new Customer(
243                     id: rs.getString( columnLabel: "고객아이디"),
244                     name: rs.getString( columnLabel: "고객이름"),
245                     age: rs.getInt( columnLabel: "나이"),
246                     grade: rs.getString( columnLabel: "등급"),
247                     job: rs.getString( columnLabel: "직업"),
248                     point: rs.getInt( columnLabel: "적립금"));
249                 customerList.add(customer);
250             }
251
252             ps.close();
253             rs.close();
254
255         } catch (SQLException e) {
256             System.out.println("Statement or SQL Error");
257             throw new RuntimeException(e);
258         }
259         return customerList;
260     }

```

```

260 @
261     private static ArrayList<Product> get productList(Connection con, String target) { 4개 사용 위치 월드 dev29
262         ArrayList<Product> productList = new ArrayList<>();
263         Product product;
264         String sql;
265
266         if (target != null) sql = "SELECT * FROM 제품 WHERE 제품번호 = ?";
267         else sql = "SELECT * FROM 제품";
268         try {
269             PreparedStatement ps = con.prepareStatement(sql);
270             if (target != null) ps.setString( parameterIndex: 1, x: target);
271             ResultSet rs = ps.executeQuery();
272
273             while (rs.next()) {
274                 product = new Product(
275                     id: rs.getString( columnLabel: "제품번호"),
276                     productName: rs.getString( columnLabel: "제품명"),
277                     productAmount: rs.getInt( columnLabel: "재고량"),
278                     productPrice: rs.getInt( columnLabel: "단가"),
279                     manufacturer: rs.getString( columnLabel: "제조업체")
280                 );
281                 productList.add(product);
282             }
283
284             ps.close();
285             rs.close();
286         } catch (SQLException e) {
287             System.out.println("Statement or SQL Error");
288             throw new RuntimeException(e);
289         }
290
291         return productList;
292     }
293
294 @
295     private static ArrayList<Order> getOrderList(Connection con, String target) { 4개 사용 위치 월드 dev29
296         ArrayList<Order> ordersList = new ArrayList<>();
297         Order order;
298         String sql;
299
300         if (target != null) sql = "SELECT * FROM 주문 WHERE 주문번호 = ?";
301         else sql = "SELECT * FROM 주문";
302         try {
303             PreparedStatement ps = con.prepareStatement(sql);
304             if (target != null) ps.setString( parameterIndex: 1, x: target);
305             ResultSet rs = ps.executeQuery();
306
307             while (rs.next()) {
308                 order = new Order(
309                     id: rs.getString( columnLabel: "주문번호"),
310                     customerid: rs.getString( columnLabel: "주문고객"),
311                     orderedProduct: rs.getString( columnLabel: "주문제품"),
312                     amount: rs.getInt( columnLabel: "수량"),
313                     deliveryAddress: rs.getString( columnLabel: "배송지"),
314                     orderDate: rs.getDate( columnLabel: "주문일자")
315                 );
316                 ordersList.add(order);
317             }
318             ps.close();
319             rs.close();
320         } catch (SQLException e) {
321             System.out.println("Statement or SQL Error");
322             throw new RuntimeException(e);
323         }
324         return ordersList;
325     }
326

```

```

327 // INSERT
328 private static void insertCustomerInfo(Connection con, Scanner sc) { 1개 사용 위치  ↳ Waltdev29
329     MainView mv = new MainView();
330     InputCustomerInfoView iciv = new InputCustomerInfoView();
331     ArrayList<Customer> inputCustomerList = new ArrayList<>();
332
333     while (true) {
334         Customer customer = iciv.inputCustomerInfo( sc: sc );
335
336         // 데이터 저장 여부 확인
337         if (!mv.askYorN( sc: sc, yesText: "데이터 저장", yesChar: "S", noText: "다시 입력", noChar: "R")) continue;
338
339         // 데이터 INSERT
340         try {
341             String sql = "INSERT INTO 고객 VALUES(?,?,?,?,?,?)";
342             PreparedStatement ps = con.prepareStatement(sql);
343             ps.setString( parameterIndex: 1, value: customer.getId());
344             ps.setString( parameterIndex: 2, value: customer.getName());
345             ps.setInt( parameterIndex: 3, value: customer.getAge());
346             ps.setString( parameterIndex: 4, value: customer.getGrade());
347             ps.setString( parameterIndex: 5, value: customer.getJob());
348             ps.setInt( parameterIndex: 6, value: customer.getPoint());
349             ps.executeUpdate();
350             ps.close();
351         } catch (SQLException e) {
352             System.out.println("Statement or SQL Error");
353             throw new RuntimeException();
354         }
355         System.out.println("데이터가 저장되었습니다. \n");
356         inputCustomerList.add(customer);
357
358         // 계속 입력받을지 선택
359         if (!mv.askYorN( sc: sc, yesText: "계속 입력", yesChar: "C", noText: "입력 종료", noChar: "E")) break;
360     }
361
362     // 입력 내용 출력
363     System.out.println("\n--- 입력 종료 ---");
364     System.out.println("===== 입력 내용 =====\n");
365     printItemList( itemList: inputCustomerList, view: new CustomerView() );
366 }

```

```

367
368     private static void insertProductInfo(Connection con, Scanner sc) { 1개 사용 위치 : Waltdev29
369         MainView mv = new MainView();
370         InputProductInfoView ipiv = new InputProductInfoView();
371         ArrayList<Product> inputProductList = new ArrayList<>();
372
373         while (true) {
374             Product product = ipiv.inputProductInfo( sc: sc );
375
376             // 데이터 저장 여부 확인
377             if (!mv.askYorN( sc: sc, yesText: "데이터 저장", yesChar: "S", noText: "다시 입력", noChar: "R")) continue;
378
379             // 데이터 INSERT
380             try {
381                 String sql = "INSERT INTO 제품 VALUES(?, ?, ?, ?, ?)";
382                 PreparedStatement ps = con.prepareStatement(sql);
383                 ps.setString( parameterIndex: 1, x: product.getId());
384                 ps.setString( parameterIndex: 2, x: product.getProductName());
385                 ps.setInt( parameterIndex: 3, x: product.getProductAmount());
386                 ps.setInt( parameterIndex: 4, x: product.getProductPrice());
387                 ps.setString( parameterIndex: 5, x: product.getManufacturer());
388                 ps.executeUpdate();
389                 ps.close();
390             } catch (SQLException e) {
391                 System.out.println("Statement or SQL Error");
392                 throw new RuntimeException(e);
393             }
394             System.out.println("데이터가 저장되었습니다. \n");
395             inputProductList.add(product);
396
397             // 계속 입력받을지 선택
398             if (!mv.askYorN( sc: sc, yesText: "계속 입력", yesChar: "C", noText: "입력 종료", noChar: "E")) break;
399         }
400
401         // 입력 내용 출력
402         System.out.println("\n--- 입력 종료 ---");
403         System.out.println("===== 입력 내용 =====\n");
404         printItemList( itemList: inputProductList, view: new ProductView() );
405     }

```

```

407     private static void insertOrderInfo(Connection con, Scanner sc) { 1개 사용 위치 : Waltdev29
408         MainView mv = new MainView();
409         InputOrderInfoView ioiv = new InputOrderInfoView();
410         ArrayList<Order> inputOrderList = new ArrayList<>();
411
412         while (true) {
413             Order order = ioiv.inputOrderInfo( sc, sc );
414
415             // 데이터 저장 여부 확인
416             if (!mv.askYorN( sc, sc, yesText: "데이터 저장", yesChar: "S", noText: "다시 입력", noChar: "R")) continue;
417
418             // //
419             // SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd");
420             // Date date = Date.valueOf(sdf.format(order.getOrderDate()));
421             Date date = new Date( date: order.getOrderDate().getTime() );
422
423             // SQL INSERT
424             try {
425                 String sql = "INSERT INTO 주문 VALUES(?,?,?,?,?,?)";
426                 PreparedStatement ps = con.prepareStatement(sql);
427                 ps.setString( parameterIndex: 1, x: order.getId());
428                 ps.setString( parameterIndex: 2, x: order.getCustomerId());
429                 ps.setString( parameterIndex: 3, x: order.getOrderedProduct());
430                 ps.setInt( parameterIndex: 4, x: order.getAmount());
431                 ps.setString( parameterIndex: 5, x: order.getDeliveryAddress());
432                 ps.setDate( parameterIndex: 6, x: date);
433                 ps.executeUpdate();
434                 ps.close();
435             } catch (SQLException e) {
436                 System.out.println("Statement or SQL Error");
437                 throw new RuntimeException(e);
438             }
439             System.out.println("데이터가 저장되었습니다.\n");
440             inputOrderList.add(order);
441
442             // 계속 입력받을지 선택
443             if (!mv.askYorN( sc, sc, yesText: "계속 입력", yesChar: "C", noText: "입력 종료", noChar: "E")) break;
444
445             // 입력 내용 출력
446             System.out.println("\n--- 입력 종료 ---");
447             System.out.println("===== 입력 내용 =====\n");
448             printItemList( itemList: inputOrderList, view: new OrderView() );
449
450
451

```

```

452     // UPDATE
453     private static boolean updateInfo(Connection con, Scanner sc, int subState, String target) { 1개 사용 위치 ↳ Waltdev29
454         MainView mv = new MainView();
455         String value = "";
456         int valueInt = 0;
457         String[] cols = new String[]{};
458         String table = " ";
459         String pk = " ";
460         int intCol1 = 0;
461         int intCol2 = 0;
462
463         if (subState == 1) {
464             cols = new String[]{"고객이름", "나이", "등급", "직업", "적립금"};
465             intCol1 = 2;
466             intCol2 = 5;
467             table = "고객";
468             pk = "고객아이디";
469         } else if (subState == 2) {
470             cols = new String[]{"제품명", "재고량", "단가", "제조업체"};
471             intCol1 = 2;
472             intCol2 = 3;
473             table = "제품";
474             pk = "제품번호";
475         } else if (subState == 3) {
476             cols = new String[]{"제품명", "배송지", "수량", "주문일자"};
477             intCol2 = 3;
478             table = "주문";
479             pk = "주문번호";
480         }
481
482         int index = mv.inputAnswer(sc, min: 0, max: cols.length);
483
484         if (index == 0) return false;
485
486         System.out.printf("\n수정할 %s 입력\n", cols[index - 1]);
487
488         if (index == intCol1 || index == intCol2) valueInt = mv.inputAnswer(sc, min: 0, max: 100000);
489         else {
490             System.out.print("\n입력 : ");
491             value = sc.nextLine();
492         }
493
494         String sql = "UPDATE " + table + " SET " + cols[index - 1] + " = ? WHERE " + pk + " = ?";
495         try {
496             PreparedStatement ps = con.prepareStatement(sql);
497             if (index == 2 || index == 5) ps.setInt(parameterIndex: 1, valueInt);
498             else ps.setString(parameterIndex: 1, value);
499             ps.setString(parameterIndex: 2, target);
500             ps.executeUpdate();
501             ps.close();
502         } catch (SQLException e) {
503             System.out.println("Statement or SQL Error");
504             throw new RuntimeException(e);
505         }
506         return true;
507     }
508
509

```

```
510 // DELETE
511     private static void deleteInfo(Connection con, int subState, String target) { 1개 사용 위치 : Waltdev29
512         String table = "";
513         String pk = "";
514
515         if (subState == 1) {
516             table = "고객";
517             pk = "고객아이디";
518         } else if (subState == 2) {
519             table = "제품";
520             pk = "제품번호";
521         } else if (subState == 3) {
522             table = "주문";
523             pk = "주문번호";
524         }
525
526         String sql = "DELETE FROM " + table + " WHERE " + pk + " = ?";
527
528         try {
529             PreparedStatement ps = con.prepareStatement(sql);
530             ps.setString( parameterIndex: 1, x target);
531             ps.executeUpdate();
532             ps.close();
533         } catch (SQLException e) {
534             System.out.println("Statement or SQL Error");
535             throw new RuntimeException(e);
536         }
537         System.out.println("\n데이터가 삭제되었습니다.");
538     }
539 }
```

## 2-2. Entity Package

### 2-2-1. Entity Class

```
1 package mvc_jdbc_test.entity;
2
3 @public abstract class Entity { 9개 사용 위치 3개 상속자 Waltdev29
4     public abstract String getId(); 11개 사용 위치 3개 구현 Waltdev29
5 }
```

### 2-2-2. Customer Class

```
1 package mvc_jdbc_test.entity;
2
3 public class Customer extends Entity { 26개 사용 위치 Waltdev29
4     String id; 3개 사용 위치
5     String name; 3개 사용 위치
6     int age; 3개 사용 위치
7     String grade; 3개 사용 위치
8     String job; 3개 사용 위치
9     int point; 3개 사용 위치
10
11     public Customer(String id, String name, int age, String grade, String job, int point) { 3개 사용 위치 Waltdev29
12         this.id = id;
13         this.name = name;
14         this.age = age;
15         this.grade = grade;
16         this.job = job;
17         this.point = point;
18     }
19
20     public String getId() { return id; }
21     public void setId(String id) { this.id = id; }
22
23     public String getName() { return name; }
24     public void setName(String name) { this.name = name; }
25
26     public int getAge() { return age; }
27     public void setAge(int age) { this.age = age; }
28
29     public String getGrade() { return grade; }
30     public void setGrade(String grade) { this.grade = grade; }
31
32     public String getJob() { return job; }
33     public void setJob(String job) { this.job = job; }
34
35     public int getPoint() { return point; }
36     public void setPoint(int point) { this.point = point; }
37 }
```

## 2-2-3. Product Class

```
1 package mvc_jdbc_test.entity;
2
3 public class Product extends Entity{ 2개 사용 위치 ▲ Waltdev29
4     String id; 2개 사용 위치
5     String productName; 2개 사용 위치
6     int productAmount; 2개 사용 위치
7     int productPrice; 2개 사용 위치
8     String manufacturer; 2개 사용 위치
9
10    public Product(String id, String productName, int productAmount, int productPrice, String manufacturer) { 3개 사용 위치
11        this.id = id;
12        this.productName = productName;
13        this.productAmount = productAmount;
14        this.productPrice = productPrice;
15        this.manufacturer = manufacturer;
16    }
17
18    public String getId() { return id; }
19
20    public String getProductName() { return productName; }
21
22    public int getProductAmount() { return productAmount; }
23
24    public int getProductPrice() { return productPrice; }
25
26    public String getManufacturer() { return manufacturer; }
27
28}
```

## 2-2-4. Order Class

```
1 package mvc_jdbc_test.entity;
2
3 import java.util.Date;
4
5 public class Order extends Entity { ▲ Waltdev29 *
6     private String id;
7     private String customerId;
8     private String orderedProduct;
9     private int amount;
10    private String deliveryAddress;
11    private Date orderDate;
12
13    public Order(String id, String customerId, String orderedProduct, int amount, String deliveryAddress, Date orderDate)
14        this.id = id;
15        this.customerId = customerId;
16        this.orderedProduct = orderedProduct;
17        this.amount = amount;
18        this.deliveryAddress = deliveryAddress;
19        this.orderDate = orderDate;
20    }
21
22    public String getId() { return id; }
23
24    public String getCustomerId() { return customerId; }
25
26    public String getOrderedProduct() { return orderedProduct; }
27
28    public int getAmount() { return amount; }
29
30    public String getDeliveryAddress() { return deliveryAddress; }
31
32    public Date getOrderDate() { return orderDate; }
33
34}
```

## 2-3. View Package

### 2-3-1. ObjectView Class

```
1 package mvc_jdbc_test.view;
2
3 @public abstract class ObjectView<T> { 8개 사용 위치 3개 상속자 ▲ Waltdev29
4
5 ⓘ↓ public abstract void printItem(T item); 4개 사용 위치 3개 구현 ▲ Waltdev29
6
7 ⓘ↓ public abstract void printHead(); 4개 사용 위치 3개 구현 ▲ Waltdev29
8
9 ⓘ↓ public abstract void printCols(); 4개 사용 위치 3개 구현 ▲ Waltdev29
10
11 public void printFoot() { 2개 사용 위치 ▲ Waltdev29
12     System.out.println("\n=====");
13     System.out.println("===== Print Done =====");
14     System.out.println("=====\\n");
15 }
16
17 ⓘ↓ public abstract void printItemWithIndex(T item); 4개 사용 위치 3개 구현 ▲ Waltdev29
18 }
```

## 2-3-2. CustomerView Class

```
1 package mvc_jdbc_test.view;
2
3 import mvc_jdbc_test.entity.Customer;
4
5 public class CustomerView extends ObjectView<Customer> { 13개 사용 위치 ▾ Waltdev29
6     String title = "고객 정보"; 1개 사용 위치
7
8     public void printHead() { 4개 사용 위치 ▾ Waltdev29
9         System.out.println("\n=====");
10        System.out.println("=====" + title + "====");
11        System.out.println("=====\\n");
12    }
13
14     public void printCols() { 4개 사용 위치 ▾ Waltdev29
15         System.out.printf("%-10s %-10s %-4s %-10s %-8s\\n", "고객아이디", "고객이름", "나이", "등급", "직업", "적립금");
16         System.out.println("-----");
17    }
18
19     @Override 4개 사용 위치 ▾ Waltdev29
20     public void printItem( @NotNull Customer customer) {
21         System.out.printf("%-14s ", customer.getId());
22         System.out.printf("%-10s ", customer.getName());
23         System.out.printf("%-5d ", customer.getAge());
24         System.out.printf("%-11s ", customer.getGrade());
25         System.out.printf("%-10s ", customer.getJob());
26         System.out.printf("%-8d ", customer.getPoint());
27    }
28
29     public void printItemWithIndex( @NotNull Customer customer) { 4개 사용 위치 ▾ Waltdev29
30         System.out.println("\n===== 고객 정보 =====");
31         System.out.printf("1. 고객이름 : %s\\n", customer.getName());
32         System.out.printf("2. 나이 : %d\\n", customer.getAge());
33         System.out.printf("3. 등급 : %s\\n", customer.getGrade());
34         System.out.printf("4. 직업 : %s\\n", customer.getJob());
35         System.out.printf("5. 적립금 : %d\\n", customer.getPoint());
36    }
37 }
```

## 2-3-3. ProductView Class

```
1 package mvc_jdbc_test.view;
2
3 import mvc_jdbc_test.entity.Product;
4
5 public class ProductView extends ObjectView<Product> { 13개 사용 위치 ▾ Waltdev29
6     String title = "제품 정보"; 1개 사용 위치
7
8     public void printHead() { 4개 사용 위치 ▾ Waltdev29
9         System.out.println("\n=====");
10        System.out.println("=====" + title + "====");
11        System.out.println("=====\\n");
12    }
13
14     public void printCols() { 4개 사용 위치 ▾ Waltdev29
15         System.out.printf("%-8s %-12s %-8s %-10s %-16s\\n", "제품번호", "제품명", "재고량", "단가", "제조업체");
16         System.out.println("-----");
17    }
18
19     @Override 4개 사용 위치 ▾ Waltdev29
20     public void printItem( @NotNull Product product) {
21         System.out.printf("%-10s ", product.getId());
22         System.out.printf("%-12s ", product.getProductName());
23         System.out.printf("%-10d ", product.getProductAmount());
24         System.out.printf("%-11d ", product.getProductPrice());
25         System.out.printf("%-16s ", product.getManufacturer());
26    }
27
28     public void printItemWithIndex( @NotNull Product product) { 4개 사용 위치 ▾ Waltdev29
29         System.out.println("\n===== 주문 정보 =====");
30         System.out.printf("1. 제품명 : %s", product.getProductName());
31         System.out.printf("2. 재고량 : %d", product.getProductAmount());
32         System.out.printf("3. 단가 : %d", product.getProductPrice());
33         System.out.printf("4. 제조업체 : %s\\n", product.getManufacturer());
34    }
35 }
```

## 2-3-4. OrderView Class

```
1 package mvc_jdbc_test.view;
2
3 import mvc_jdbc_test.entity.Order;
4
5 public class OrderView extends ObjectView<Order> { 13개 사용 위치 ↳ Waltdev29
6     String title = "주문 정보"; 1개 사용 위치
7
8     public void printHead() { 4개 사용 위치 ↳ Waltdev29
9         System.out.println("\n=====");
10        System.out.println("=====" + title + "====");
11        System.out.println("=====\\n");
12    }
13
14     public void printCols() { 4개 사용 위치 ↳ Waltdev29
15         System.out.printf("%-6s %-10s %-10s %-15s %-4s %-10s\\n", "주문번호", "고객아이디", "제품명", "배송지", "수량", "주문일자");
16         System.out.println("-----");
17    }
18
19     @Override 4개 사용 위치 ↳ Waltdev29
20     public void printItem( @NotNull Order order) {
21         System.out.printf("%-7s", order.getId());
22         System.out.printf("%-13s ", order.getCustomerId());
23         System.out.printf("%-10s ", order.getOrderedProduct());
24         System.out.printf("%-13s ", order.getDeliveryAddress());
25         System.out.printf("%-5d ", order.getAmount());
26         System.out.printf("%-10s ", order.getOrderDate());
27    }
28
29     public void printItemWithIndex( @NotNull Order order) { 4개 사용 위치 ↳ Waltdev29
30         System.out.println("\n===== 제품 정보 =====");
31         System.out.printf("1. 제품명 : %s", order.getOrderedProduct());
32         System.out.printf("2. 배송지 : %s", order.getDeliveryAddress());
33         System.out.printf("3. 수량 : %d", order.getAmount());
34         System.out.printf("4. 주문일자 : %s\\n", order.getOrderDate());
35    }
36 }
```

## 2-3-5. MainView Class

```
1 package mvc_jdbc_test.view;
2
3 import java.util.Scanner;
4
5 public class MainView { 26개 사용 위치 ▲ Waltdev29
6     public void showHomeView() { 2개 사용 위치 ▲ Waltdev29
7         System.out.println("\n##### Shop DB #####\n");
8         System.out.println("--- 모드 선택 ---");
9         System.out.println("0. 프로그램 종료");
10        System.out.println("1. 데이터 조회");
11        System.out.println("2. 데이터 추가");
12        System.out.println("3. 데이터 수정");
13        System.out.println("4. 데이터 삭제");
14    }
15
16    public void showMainView(String mode) { 5개 사용 위치 ▲ Waltdev29
17        System.out.printf("\n##### %s 모드 #####\n\n", mode);
18        System.out.printf("--- %s DB 선택 ---\n", mode);
19        System.out.println("0. 뒤로가기");
20        System.out.println("1. 고객 DB");
21        System.out.println("2. 제품 DB");
22        System.out.println("3. 주문 DB");
23    }
24
25    @
26    public static void inputEnter( @NotNull Scanner sc) { 16개 사용 위치 ▲ Waltdev29
27        System.out.println("\nEnter를 놀려 돌아가기");
28        sc.nextLine();
29    }
30    @
31    public int inputAnswer( @NotNull Scanner sc, int min, int max) { 15개 사용 위치 ▲ Waltdev29
32        int answer;
33
34        while (true) {
35            try {
36                System.out.print("\n입력 : ");
37                String line = sc.nextLine();
38                answer = Integer.parseInt( line);
39                if (answer < min || answer > max) {
40                    System.out.printf("%d~%d의 숫자를 입력해주세요.", min, max);
41                    continue;
42                }
43                break;
44            } catch (NumberFormatException e) {
45                System.out.println("잘못된 입력입니다. 숫자를 입력해주세요.");
46            }
47        }
48        return answer;
49    }
50    @
51    public boolean askYorN( @NotNull Scanner sc, String yesText, String yesChar, String noText, String noChar) { 14개 사용 위치 ▲ Waltdev29
52        String answer;
53        while (true) {
54            System.out.printf("\n%s : %s\n%s : %s\n", yesText, yesChar, noText, noChar);
55            System.out.print("\n입력 : ");
56            answer = sc.nextLine();
57
58            if (answer.equalsIgnoreCase( yesChar)) return true;
59            else if (answer.equalsIgnoreCase( noChar)) return false;
60            else System.out.println("잘못된 입력입니다. 다시 입력해주세요.\n");
61        }
62    }
}
```

## 2-3-6. InputCustomerInfoView Class

```
1 package mvc_jdbc_test.view;
2
3 import mvc_jdbc_test.entity.Customer;
4
5 import java.util.Scanner;
6
7 public class InputCustomerInfoView { 4개 사용 위치  ↳ Waltdev29
8
9     @
10    public Customer inputCustomerInfo( @NotNull Scanner sc) { 2개 사용 위치  ↳ Waltdev29
11        String customerId;
12        String customerName;
13        int customerAge;
14        String customerGrade;
15        String customerJob;
16        int customerPoint;
17
18        System.out.println("\n고객 정보를 입력해주세요.");
19        System.out.println("고객 아이디, 고객 이름, 고객 나이, 고객 등급, 고객 직업, 고객 적립금");
20        System.out.print("\n고객 아이디 : ");
21        customerId = sc.nextLine();
22        System.out.print("\n고객 이름 : ");
23        customerName = sc.nextLine();
24
25        while (true) {
26            try {
27                System.out.print("\n고객 나이 : ");
28                String line = sc.nextLine();
29                customerAge = Integer.parseInt( line.trim());
30                break;
31            } catch (NumberFormatException e) {
32                System.out.println("잘못된 입력입니다. 숫자를 입력해주세요.\n");
33            }
34        }
35
36        System.out.print("\n고객 등급 : ");
37        customerGrade = sc.nextLine();
38        System.out.print("\n고객 직업 : ");
39        customerJob = sc.nextLine();
40
41        while (true) {
42            try {
43                System.out.print("\n고객 적립금 : ");
44                String line = sc.nextLine();
45                customerPoint = Integer.parseInt( line.trim());
46                break;
47            } catch (NumberFormatException e) {
48                System.out.println("잘못된 입력입니다. 숫자를 입력해주세요.\n");
49            }
50        }
51
52        System.out.println("### 입력 정보 ###");
53        System.out.println("고객 아이디 : " + customerId);
54        System.out.println("고객 이름 : " + customerName);
55        System.out.println("고객 나이 : " + customerAge);
56        System.out.println("고객 등급 : " + customerGrade);
57        System.out.println("고객 직업 : " + customerJob);
58        System.out.println("고객 적립금 : " + customerPoint);
59
60        return new Customer( id: customerId, customerName, customerAge, customerGrade, customerJob, customerPoint);
61    }
62 }
```

## 2-3-7. InputProductInfoView Class

```
1 package mvc_jdbc_test.view;
2
3 import mvc_jdbc_test.entity.Product;
4
5 import java.util.Scanner;
6
7 public class InputProductInfoView { 4개 사용 위치  ↳ Waltdev29
8     public Product inputProductInfo( @NotNull Scanner sc) { 2개 사용 위치  ↳ Waltdev29
9         String productId;
10        String productName;
11        int productAmount;
12        int productPrice;
13        String manufacturer;
14
15        System.out.println("\n제품 정보를 입력해주세요.");
16        System.out.println("제품번호, 제품명, 재고량, 단가, 제조업체");
17
18        System.out.print("\n제품번호 : ");
19        productId = sc.nextLine();
20        System.out.print("\n제품명 : ");
21        productName = sc.nextLine();
22
23        while (true) {
24            try {
25                System.out.print("\n재고량 : ");
26                String line = sc.nextLine();
27                productAmount = Integer.parseInt( line.trim());
28                break;
29            } catch (NumberFormatException e) {
30                System.out.println("잘못된 입력입니다. 숫자를 입력해주세요.\n");
31            }
32        }
33
34        while (true) {
35            try {
36                System.out.print("\n단가 : ");
37                String line = sc.nextLine();
38                productPrice = Integer.parseInt( line.trim());
39                break;
40            } catch (NumberFormatException e) {
41                System.out.println("잘못된 입력입니다. 숫자를 입력해주세요.\n");
42            }
43        }
44
45        System.out.print("\n제조업체 : ");
46        manufacturer = sc.nextLine();
47
48        System.out.println("### 입력 정보 ###");
49        System.out.println("제품번호 : " + productId);
50        System.out.println("제품명 : " + productName);
51        System.out.println("재고량 : " + productAmount);
52        System.out.println("단가 : " + productPrice);
53        System.out.println("제조사 : " + manufacturer);
54
55        return new Product( productId, productName, productAmount, productPrice, manufacturer);
56    }
57 }
58 }
```

## 2-3-8. InputOrderInfoView Class

```
1 package mvc_jdbc_test.view;
2
3 import mvc_jdbc_test.entity.Order;
4
5 import java.text.ParseException;
6 import java.text.SimpleDateFormat;
7 import java.util.Date;
8 import java.util.Scanner;
9
10 public class InputOrderInfoView { 4개 사용 위치  ↳ Waltdev29
11     @NotNull Scanner sc) { 2개 사용 위치  ↳ Waltdev29
12         String orderId;
13         String customerId;
14         String orderedProduct;
15         int amount;
16         String deliveryAddress;
17         Date orderDate;
18
19         System.out.println("\n주문 정보를 입력해주세요.");
20         System.out.println("(주문 번호, 주문 고객, 주문 제품, 수량, 배송지, 주문일자)");
21
22         System.out.print("\n주문 번호 : ");
23         orderId = sc.nextLine();
24         System.out.print("\n주문 고객 : ");
25         customerId = sc.nextLine();
26         System.out.print("\n주문 제품 : ");
27         orderedProduct = sc.nextLine();
28
29         while (true) {
30             try {
31                 System.out.print("\n수량 : ");
32                 String line = sc.nextLine();
33                 amount = Integer.parseInt( line.trim());
34                 break;
35             } catch (NumberFormatException e) {
36                 System.out.println("잘못된 입력입니다. 숫자를 입력해주세요.\n");
37             }
38         }
39
40         System.out.print("\n배송지 : ");
41         deliveryAddress = sc.nextLine();
42
43         while (true) {
44             try {
45                 System.out.print("\n주문일자 : ");
46                 String line = sc.nextLine();
47                 SimpleDateFormat sdf = new SimpleDateFormat( pattern: "yyyy-MM-dd");
48                 orderDate = sdf.parse( source: line);
49                 break;
50             } catch (ParseException e) {
51                 System.out.println("잘못된 입력입니다. yyyy-MM-dd 형식으로 입력해주세요.\n");
52             }
53         }
54
55         System.out.println("### 입력 정보 ###");
56         System.out.println("주문 번호 : " + orderId);
57         System.out.println("주문 고객 : " + customerId);
58         System.out.println("주문 제품 : " + orderedProduct);
59         System.out.println("수량 : " + amount);
60         System.out.println("배송지 : " + deliveryAddress);
61         System.out.println("주문일자 : " + orderDate);
62
63         return new Order( id: orderId, customerId, orderedProduct, amount, deliveryAddress, orderDate);
64     }
65 }
```

### 3. 결과 및 분석

#### 4-1. 결과

##### 1. 메인 화면

```
##### Shop DB #####
--- 모드 선택 ---
0. 프로그램 종료
1. 데이터 조회
2. 데이터 추가
3. 데이터 수정
4. 데이터 삭제

입력 : qw
잘못된 입력입니다. 숫자를 입력해주세요.

입력 : 5
0~4의 숫자를 입력해주세요.

입력 : 1
```

- 숫자입력 예외처리 적용됨

## 2-1. 조회 화면

```
##### 조회 모드 #####
--- 조회 DB 선택 ---
0. 뒤로가기
1. 고객 DB
2. 제품 DB
3. 주문 DB

입력 : 1
```

## 2-2. 고객 테이블 조회

```
=====
=====고객 정보=====
=====



| 고객아이디      | 고객이름 | 나이 | 등급     | 직업   | 적립금   |
|------------|------|----|--------|------|-------|
| apple      | 정소화  | 20 | gold   | 학생   | 1000  |
| banana     | 김선우  | 25 | vip    | 간호사  | 2500  |
| carrot     | 고명석  | 28 | gold   | 교사   | 4500  |
| orange     | 김용욱  | 22 | silver | 학생   | 0     |
| melon      | 성원용  | 35 | gold   | 회사원  | 5000  |
| peach      | 오형준  | 0  | silver | 의사   | 300   |
| pear       | 채광주  | 31 | silver | 회사원  | 500   |
| qwer       | qwer | 10 | qwer   | qwer | 20    |
| strawberry | 최유경  | 30 | vip    | 공무원  | 100   |
| grapes     | 허선희  | 0  | vip    | 회사원  | 1200  |
| moonberry  | 문세진  | 25 | gold   | 학생   | 10000 |


=====

===== Print Done =====

Enter를 눌러 돌아가기
```

### 3-1. 추가 화면

```
##### 추가 모드 #####
--- 추가 DB 선택 ---
0. 뒤로가기
1. 고객 DB
2. 제품 DB
3. 주문 DB

입력 : 1|
```

### 3-2. 추가 데이터 입력

```
고객 정보를 입력해주세요.
(고객 아이디, 고객 이름, 고객 나이, 고객 등급, 고객 직업, 고객 적립금)

고객 아이디 : qwer

고객 이름 : qwer

고객 나이 : 30

고객 등급 : qwre

고객 직업 : qwer

고객 적립금 : wne
잘못된 입력입니다. 숫자를 입력해주세요.

고객 적립금 : 36|
```

- 숫자입력 예외처리 적용됨

### 3-3. 입력 데이터 확인 및 저장

```
### 입력 정보 ###
고객 아이디 : qwer
고객 이름 : qwer
고객 나이 : 30
고객 등급 : qwre
고객 직업 : qwer
고객 적립금 : 30

데이터 저장 : S
다시 입력 : R

입력 : s
```

### 3-4. 입력 종료 및 데이터 확인

```
데이터가 저장되었습니다.

계속 입력 : C
입력 종료 : E

입력 : e

--- 입력 종료 ---
===== 입력 내용 =====

=====
=====고객 정보=====
=====

고객아이디      고객이름      나이      등급      직업      적립금
-----
qwer          qwer        30      qwre      qwer        30

=====
===== Print Done =====
=====

Enter를 눌러 돌아가기
```

#### 4-1. 수정 화면

```
##### 수정 모드 #####
--- 수정 DB 선택 ---
0. 뒤로가기
1. 고객 DB
2. 제품 DB
3. 주문 DB

입력 : 1
```

#### 4-2. 고객 테이블 출력

```
=====
=====고객 정보=====
=====

고객아이디 고객이름 나이 등급 직업 적립금
-----
apple     정소화    20   gold   학생   1000
banana   김선우    25   vip    간호사   2500
carrot    고명석    28   gold   교사    4500
orange    김용욱    22   silver  학생   0
melon    성원용    35   gold   회사원  5000
peach    오형준    0    silver  의사    300
pear     채광주    31   silver  회사원  500
strawberry 최유경    30   vip    공무원 100
qwer     qwer      30   qwre   qwer   30
grapes   허선희    0    vip    회사원  1200
moonberry 문세진    25   gold   학생   10000

=====
===== Print Done =====
=====

고객의 고객 아이디를 입력하세요.

고객 아이디 : qwer
```

#### 4-3. 수정할 항목 결정

```
===== 고객 정보 =====  
1. 고객이름 : qwer  
2. 나이 : 30  
3. 등급 : qwre  
4. 직업 : qwer  
5. 적립금 : 30  
  
수정할 항목의 번호를 입력해주세요. 뒤로가기 : 0  
  
입력 : 2
```

#### 4-4. 수정할 데이터 입력

```
수정할 나이 입력  
  
입력 : qwe  
잘못된 입력입니다. 숫자를 입력해주세요.  
  
입력 : 25
```

- 숫자 입력 예외처리 적용됨

#### 4-5. 데이터 수정 완료

```
수정이 완료되었습니다.  
  
--- 수정 정보 ---  
===== 고객 정보 =====  
1. 고객이름 : qwer  
2. 나이 : 25  
3. 등급 : qwre  
4. 직업 : qwer  
5. 적립금 : 30  
  
Enter를 눌러 돌아가기
```

## 5-1. 삭제 화면

```
##### 삭제 모드 #####
--- 삭제 DB 선택 ---
0. 뒤로가기
1. 고객 DB
2. 제품 DB
3. 주문 DB

입력 : 1
```

## 5-2. 삭제할 데이터 선택

```
=====
=====고객 정보=====
=====

고객아이디      고객이름      나이      등급      직업      적립금
-----
apple          정소화        20      gold       학생      1000
banana         김선우        25      vip        간호사     2500
carrot          고명석        28      gold       교사      4500
orange          김용국        22      silver     학생      0
melon           성원용        35      gold       회사원    5000
peach           오형준        0       silver     의사      300
pear            채팜주        31      silver     회사원    500
qwer            qwer          10      qwer      qwer      20
strawberry      최유경        30      vip        공무원    100
grapes          허선희        0       vip        회사원    1200
moonberry       문세진        25      gold       학생     10000

=====
===== Print Done =====
=====

고객의 고객 아이디를 입력하세요.

고객 아이디 : qwer
```

### 5-3. 삭제 확인

=====고객 정보=====					
고객아이디	고객이름	나이	등급	직업	적립금
qwer	qwer	10	qwer	qwer	20
해당 데이터를 삭제하시겠습니까?					
삭제 : Y					
취소 : N					
입력 : <b>y</b>					

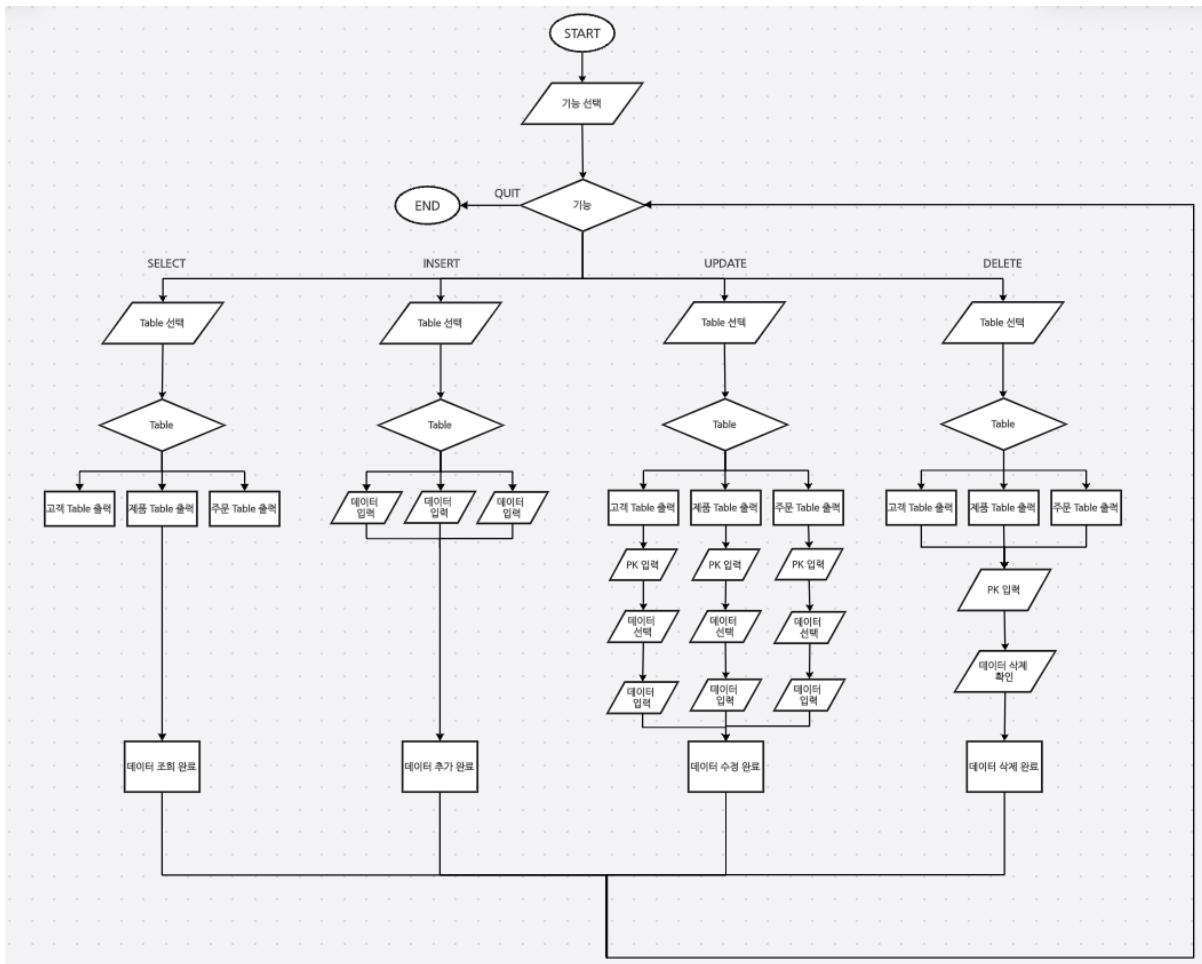
### 5-4. 데이터 삭제 완료

데이터가 삭제되었습니다.

**Enter**를 눌러 돌아가기

## 4-2. 분석

2. 처음 만들었던 mainController 는 아래와 같은 흐름으로 구현하였음.



- Switch Case 로 각 기능에 따라 분기를 나누었음.
- 기능별로 분기가 나누어져 흐름을 제어하고 수정하는 것이 용이하였음.

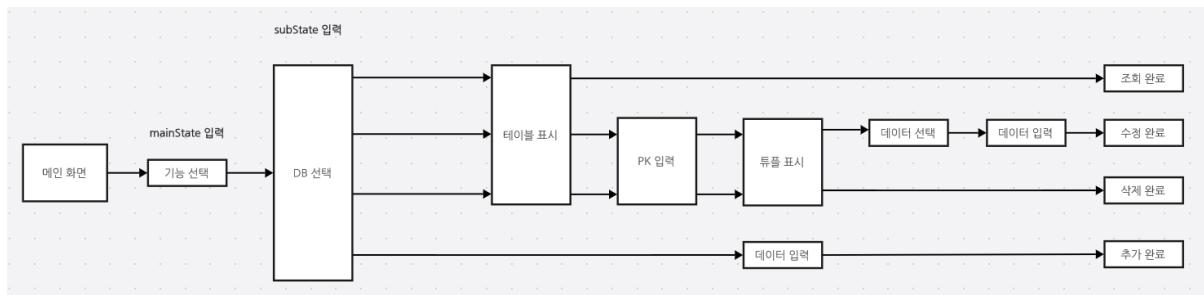
```

33     while (!quit) {
34         switch (mainState) {
35             // 메인
36             case 0:
37                 mv.showHomeView();
38                 mainState = mv.inputAnswer(sc, sc, min: 0, max: 4);
39                 if (mainState == 0) quit = true;
40                 break;
41
42             // 데이터 조회
43             case 1:
44                 switch (subState) {...}
45                 break;
46
47             // 데이터 추가 화면
48             case 2:
49                 switch (subState) {...}
50                 break;
51
52             // 데이터 수정 화면
53             case 3:
54                 switch (subState) {...}
55                 break;
56
57             // 데이터 삭제 화면
58             case 4:
59                 switch (subState) {...}
60                 break;
61
62             default:
63                 System.out.println("잘못된 입력입니다. 1~3의 정수를 입력해주세요.");
64             }
65             if (mainState == -1) break;
66         }
67
68         System.out.println("\n프로그램을 종료합니다.");
69         sc.close();
70     }

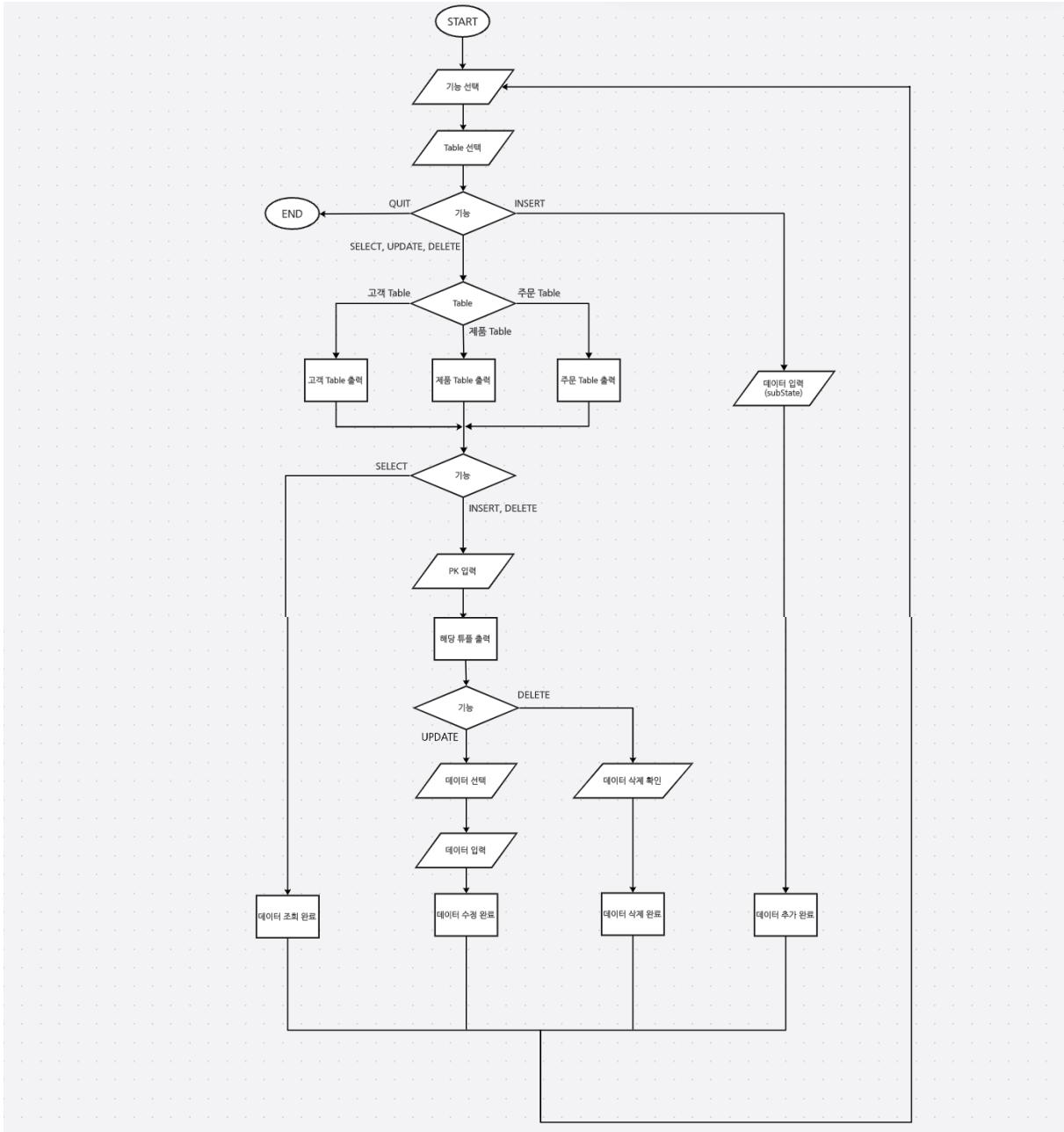
```

### [MainController main 함수]

- 하지만 분기별로 겹치는 프로세스가 있어 중복을 최대한 줄이고자 시스템 흐름을 아래와 같이 도식화 함.



### 3. 도식을 기반으로 중복을 줄여 코드를 재구성하였음.



- 이전의 코드가 처음부터 Switch Case로 분기가 나뉘어졌다면, 이번 코드는 흐름이 진행되며 if문을 통해 분기가 나누어지는 구조로 재구성 함.
- 코드의 흐름을 바꾸며 기능이 겹치는 함수들을 통합할 수 있었음.
- 중복을 많이 제거하여 코드의 효율성이 향상되었으며, 코드 라인 수가 100라인가량 짧아졌음.

3. 개발 중간부터 중복을 최소화하기 위해 추상화와 제네릭을 공부하였음.

Entity들을 다형성을 활용하여 관리하고 클래스, 함수들을 제네릭을 활용하여 하나로 통합하려 하였으나 중간부터 수정하려 하니 한계가 있었음.

설계부터 유연성을 고려하며 설계하는 것이 중요하다고 느꼈음.