

# 출고관리

## 출고내역 | 출고서작성

카테고리 ▾ 연도. 월. 일. ☰ ~ 연도. 월. 일. ☰ 지점명 검색

번호	지점	품목	금액	출고서등록일	출고완료일	출고상태							
3	전농점	닭 9호(마일드) 외 1개 품목	57000	2022-09-24		N							
상품코드	썸네일	상품명	카테고리	규격	제조사	재고량	발주수량	출고수량	출고입력	요청일	출고일	출고상태	총매출단가
1004	닭 8호(핫)	닭고기	ea	하림	9935	5	0		2022-09-22		N	27000	
1005	닭 9호(마일드)	닭고기	ea	하림	9973	5	0		2022-09-22		N	30000	
2	전농점	닭 8호(마일드) 외 1개 품목	51000	2022-09-24		N							
1	전농점	쌀파우더 외 1개 품목	48500	2022-09-24		N							

```
@GetMapping("/releaseSelect")
```

```
public ModelAndView releaseSelect(ModelAndView mv,
```

```
    @ModelAttribute ReleaseSelectCriteria releaseSelectCriteria,
```

```
    HttpServletRequest request){ @ModelAttribute로 검색 결과값을 ReleaseSelectCriteria 형태로 받아옵니다.
```

```
String currentPage = request.getParameter( name: "currentPage"); currentPage를 request.getParameter를 사용하여 받아옵니다.
```

```
int pageNo = 1; 기본 페이지 넘버를 1로 초기화 했습니다.
```

```
if(currentPage != null && !"".equals(currentPage)){
```

```
pageNo = Integer.parseInt(currentPage); currentPage에 아무 값도 들어오지 않으면 currentPage를 pageNo인 1로 설정합니다.
```

```
}
```

```
int totalCount = releaseService.totalCount(releaseSelectCriteria); 출고내역 조회 시 조회목록의 총 개수를 가져옵니다.  
parameter로 검색값을 전달합니다.
```

```
String searchCategory = releaseSelectCriteria.getSearchCategory();
```

```
String searchDate = releaseSelectCriteria.getSearchDate();
```

```
String searchDate2 = releaseSelectCriteria.getSearchDate2();
```

```
String searchValue = releaseSelectCriteria.getSearchValue();
```

```
int limit = 10; 한페이지에 출력하는 조회목록의 개수를 10개로 설정합니다.
```

```
int buttonAmount = 5; 페이지 버튼의 숫자를 5개로 설정합니다.
```

```
if(searchCategory != null && !"".equals(searchCategory)){
    검색시 카테고리를 선택할 경우 검색조건 및 검색키워드를 releaseSelectCriteria에 담습니다.
    releaseSelectCriteria = ReleasePagenation.getReleaseSelectCriteria(pageNo, totalCount, limit, buttonAmount,
                                                                     searchCategory, searchValue, searchDate, searchDate2);
} else {
    검색하지 않을땐 검색조건 및 키워드값을 releaseSelectCriteria에 담지 않습니다.
    releaseSelectCriteria = ReleasePagenation.getReleaseSelectCriteria(pageNo, totalCount, limit, buttonAmount);
}

List<ReleseaseDTO> 자료형으로 출고내역을 조회하여 releaseSelect에 담습니다.
List<ReleaseDTO> releaseSelect = releaseService.releaseSelect(releaseSelectCriteria);

List<List<ReleaseOrderDTO>> releaseSelectItem = new ArrayList<>(); 아래 반복문에서 가져온 결과값을 List의 형태로 담습니다.

for(int i = 0; i < releaseSelect.size(); i++){
    int relCode = releaseSelect.get(i).getRelCode(); 출고내역에 있는 출고번호를 가져옵니다.
    List<ReleaseOrderDTO> releaseSelect2 = releaseService.releaseSelect2(relCode); 출고번호를 파라미터로 전달하여 출고번호에 맞는 출고상세품목들을 조회합니다.

    releaseSelectItem.add(releaseSelect2); 상세품목 조회결과값을 releaseSelectItem에 담습니다.
}
    검색 후 페이지 이동할때 같은 검색값을 가지고 이동하기 위해 addObject를 사용하여 view로 값을 보냅니다.
mv.addObject( attributeName: "releaseSelectCriteria", releaseSelectCriteria);
mv.addObject( attributeName: "releaseSelect", releaseSelect); 출고내역 조회 결과를 addObject에 담아 view로 보냅니다.
mv.addObject( attributeName: "releaseSelectItem", releaseSelectItem); 출고내역 상세 조회 결과를 addObject에 담아 view로 보냅니다.
mv.setViewName("release/release_list"); 값을 출력할 페이지로 이동시킵니다.
return mv;
```

new

@Override 페이지 할 때 조회목록의 총 개수를 가져옵니다.

```
public int totalCount(ReleaseSelectCriteria selectCriteria) {  
  
    int result = releaseMapper.totalCount(selectCriteria);  
  
    return result;  
}
```

1 usage ▾ Jihwan-Kim

@Override 출고내역의 조회목록을 가져옵니다.

```
public List<ReleaseDTO> releaseSelect(ReleaseSelectCriteria selectCriteria) {  
  
    List<ReleaseDTO> releaseSelect = releaseMapper.releaseSelect(selectCriteria);  
  
    return releaseSelect;  
}
```

1 usage ▾ Jihwan-Kim

@Override 출고내역의 상세조회목록을 가져옵니다.

```
public List<ReleaseOrderDTO> releaseSelect2(int relCode) {  
  
    List<ReleaseOrderDTO> releaseItemListSelect = releaseMapper.releaseSelect2(relCode);  
  
    return releaseItemListSelect;  
}
```

```
public class ReleaseSelectCriteria {  
    4 usages  
    private String searchDate;  
    4 usages  
    private String searchDate2;  
    4 usages  
    private String searchValue;  
    4 usages  
    private String searchCategory;  
    4 usages  
    private int pageNo;  
    4 usages  
    private int totalCount;  
    4 usages  
    private int limit;  
    4 usages  
    private int buttonAmount;  
    4 usages  
    private int maxPage;  
    4 usages  
    private int startPage;  
    4 usages  
    private int endPage;  
    4 usages  
    private int startRow;  
    4 usages  
    private int endRow;  
    3 usages  ↳ Jihwan-Kim  
    public ReleaseSelectCriteria() {}
```

```
Usage  ↳ Jihwan-Kim  
public ReleaseSelectCriteria(int pageNo, int totalCount, int limit, int buttonAmount, int maxPage,  
                            int startPage, int endPage, int startRow, int endRow, String searchCategory,  
                            String searchValue, String searchDate, String searchDate2) {  
    this.pageNo = pageNo;  
    this.totalCount = totalCount;  
    this.limit = limit;  
    this.buttonAmount = buttonAmount;  
    this.maxPage = maxPage;  
    this.startPage = startPage;  
    this.endPage = endPage;  
    this.startRow = startRow;  
    this.endRow = endRow;  
    this.searchDate = searchDate;  
    this.searchDate2 = searchDate2;  
    this.searchValue = searchValue;  
    this.searchCategory = searchCategory;  
}  
  
3 usages  ↳ Jihwan-Kim  
public String getSearchDate() { return searchDate; }  
  
    ↳ Jihwan-Kim  
public void setSearchDate(String searchDate) { this.searchDate = searchDate; }  
  
3 usages  ↳ Jihwan-Kim  
public String getSearchDate2() { return searchDate2; }  
  
    ↳ Jihwan-Kim  
public void setSearchDate2(String searchDate2) { this.searchDate2 = searchDate2; }
```

```
public class ReleasePagenation2 {

    public static ReleaseSelectCriteria getReleaseSelectCriteria(int pageNo, int totalCount, int limit, int buttonAmount){

        return getReleaseSelectCriteria(pageNo, totalCount, limit, buttonAmount,
            searchCategory: null, searchValue: null, searchDate: null, searchDate2: null);
    }

    1 usage
    public static ReleaseSelectCriteria getReleaseSelectCriteria(int pageNo, int totalCount, int limit,
        int buttonAmount, String searchCategory, String searchValue,
        String searchDate, String searchDate2) {

        int maxPage;
        int startPage;
        int endPage;
        int startRow;
        int endRow;

        maxPage = (int) Math.ceil((double) totalCount / limit);

        startPage = (int) (Math.ceil((double) pageNo / buttonAmount) - 1) * buttonAmount + 1;

        endPage = startPage + buttonAmount - 1;

        if(maxPage < endPage){
            endPage = maxPage;
        }

        if(maxPage == 0 && endPage == 0) {
            maxPage = startPage;
            endPage = startPage;
        }

        startRow = (pageNo - 1) * limit + 1;
        endRow = startRow + limit - 1;

        System.out.println("startRow : " + startRow);
        System.out.println("endRow : " + endRow);

        ReleaseSelectCriteria releaseSelectCriteria =
            new ReleaseSelectCriteria(pageNo, totalCount, limit, buttonAmount ,maxPage, startPage,
                endPage, startRow, endRow, searchCategory, searchValue, searchDate, searchDate2);

        return releaseSelectCriteria;
    }
}
```

```

<tbody>
<div th:each="release : ${ releaseSelect }"> controller에서 보낸 조회값을 th:each를 사용하여 반복출력합니다.
    <tr class="detail-main">
        <td>
            <div class="form-check">
                <input type="checkbox" class="form-check-input" name="option"
                    value="something">
                <label class="form-check-label"></label>
            </div>
        </td>
        <td class="relCode" th:text="${ release.relCode }"></td>
        <td th:text="${ release.storeName }"></td>
        <td th:text="${ release.relName }"></td>
        <td th:text="${ release.totalMoney }"></td>
        <td th:text="${ release.relListDate }"></td>
        <td th:text="${ release.relDate }"></td>
        <td th:text="${ release.relyn }"></td>
    </tr>

```

```

<div th:each="release : ${ releaseSelect }"> 출고내역조회
    <tr class="detail-main" ...>
        <tr class="detail-sub-tr">
            <td class="detail-sub-td p-0" colspan="9">
                <div class="detail-sub">
                    <table class="detail-table" width="100%">
                        <th:each="detail : ${ releaseSelectItem }" th:if="${ release.relCode eq detail[0].releaseItemDTO.relCode }">
                            <thead>
                                <tr ...>
                            </thead>
                            <tr class="detail-tr" th:each="subDetail : ${ detail }">
                                <form name="releaseItemForm" class="releaseItemForm" onsubmit="return false">
                                    <td>
                                        <div class="form-check">
                                            <input type="checkbox" class="form-check-input"
                                                name="option" value="something">
                                            <label class="form-check-label"></label>
                                        </div>
                                    </td>

```

조건문을 사용하여 relcode 별 출고내역 상세물품을 출력합니다.

```
<form th:action="@{ /release/releaseSelect }"> Form 태그를 사용하여 검색 조건 및 검색어를 controller로 전달합니다.  
<div class="input-group mb-3 col-5 float-right" style="...>  
    <select type="" class="form-control" id="searchCategory" th:name="searchCategory" value=""  
        style="height: 40px; min-width: max-content">  
        검색 후 페이지 이동 시 카테고리 및 검색어를 유지하도록 합니다.  
        <option value="category">카테고리</option>  
        <option value="searchRelDate" th:selected="${ releaseSelectCriteria.getSearchCategory() == 'searchRelDate' }">출고일</option>  
        <option value="searchRelRegistDate" th:selected="${ releaseSelectCriteria.getSearchCategory() == 'searchRelRegistDate' }">등록일</option>  
        <option value="searchStoreName" th:selected="${ releaseSelectCriteria.getSearchCategory() == 'searchStoreName' }">지점명</option>  
        <option value="relDateStoreName" th:selected="${ releaseSelectCriteria.getSearchCategory() == 'relDateStoreName' }">출고일+지점명</option>  
        <option value="registDateStoreName" th:selected="${ releaseSelectCriteria.getSearchCategory() == 'registDateStoreName' }">등록일+지점명</option>  
    </select>  
    <button class="btn btn-light px-1" type="button" disabled></button>  
    <input type="date" class="form-control" id="searchDate" name="searchDate" th:value="${ releaseSelectCriteria.getSearchDate() }"  
        style="...>  
    <button class="btn btn-light" type="button" disabled>~</button>  
    <input type="date" class="form-control" id="searchDate2" name="searchDate2" th:value="${ releaseSelectCriteria.getSearchDate2() }"  
        style="min-width: max-content">  
    <button class="btn btn-light px-1" type="button" disabled></button>  
    <input class="form-control me-2" type="text" id="storeName" name="searchValue" th:value="${ releaseSelectCriteria.getSearchValue() }"  
        placeholder="지점명" style="min-width: max-content">  
    <button class="btn btn-warning" onclick="find();" type="submit" style="color:#fff">검색</button>  
</div>  
</form>
```

```

<div class="pagingArea" align="center">
    <button class="btn btn-light mx-1" style="..." 현재 검색값을 url에 쿼리스트링형식으로 controller로 전달합니다.
        th:onclick="'location.href='/release/releaseSelect?currentPage=' + @${releaseSelectCriteria.startPage}
                    + '&searchCategory=' + @${releaseSelectCriteria.getSearchCategory()}
                    + '&searchValue=' + @${releaseSelectCriteria.getSearchValue()} + '\\''"
        th:disabled="${releaseSelectCriteria.pageNo == 1}" 현재페이지가 첫번째 페이지면 << 버튼을 비활성화 합니다.
    >
    <<
</button>
    <button class="btn btn-light" style="...">
        th:onclick="'location.href='/release/releaseSelect?currentPage=' + @${releaseSelectCriteria.pageNo - 1}
                    + '&searchCategory=' + @${releaseSelectCriteria.getSearchCategory()}
                    + '&searchValue=' + @${releaseSelectCriteria.getSearchValue()} + '\\''"
        th:disabled="${releaseSelectCriteria.pageNo == 1}" 현재페이지가 첫번째 페이지면 < 버튼을 비활성화 합니다.
    >
    <
</button> 타임리프 th:each 반복을 통해 버튼의 번호를 생성합니다.
<th:block th:each="page : ${numbers.sequence(releaseSelectCriteria.startPage, releaseSelectCriteria.endPage)}">
    <button class="btn btn-warning mx-1" style="cursor:pointer; color: #fff;">
        th:text="${page}"
        th:onclick="'location.href='/release/releaseSelect?currentPage=' + @${page}
                    + '&searchCategory=' + @${releaseSelectCriteria.getSearchCategory()}
                    + '&searchValue=' + @${releaseSelectCriteria.getSearchValue()} + '\\''"
        th:disabled="${releaseSelectCriteria.pageNo eq page}" 현재페이지의 버튼을 비활성화 합니다.
    </button>
</th:block>

```

\* >, >> 버튼 또한 코드구성이 상기와 같습니다.

```

<select id="releaseSelect" resultMap="releaseResultMap">
    SELECT
        A.RNUM
        , A.REL_CODE
        , A.REL_NAME
        , A.REL_DATE
        , A.REL_YN
        , A.STORE_NAME
        , A.TOTAL MONEY
        , A.REL_LIST_DATE
    FROM (SELECT
            ROWNUM RNUM
            , B.REL_CODE
            , B.REL_NAME
            , B.REL_DATE
            , B.REL_YN
            , B.STORE_NAME
            , B.TOTAL MONEY
            , B.REL_LIST_DATE
        FROM (SELECT
                REL_CODE
                , REL_NAME
                , REL_DATE
                , REL_YN
                , STORE_NAME
                , TOTAL MONEY
                , REL_LIST_DATE
            FROM RELEASE A

```

```

        <if test="searchCategory == 'searchStoreName'">
            WHERE A.STORE_NAME LIKE '%' || #{searchValue} || '%'
        </if>
        <if test="searchCategory == 'searchRelRegistDate'">
            WHERE TO_CHAR(A.REL_LIST_DATE, 'YYYY-MM-DD') BETWEEN #{searchDate}
                AND #{searchDate2}
        </if>
        <if test="searchCategory == 'searchRelDate'">
            WHERE TO_CHAR(A.REL_DATE, 'YYYY-MM-DD') BETWEEN #{searchDate}
                AND #{searchDate2}
        </if>
        <if test="searchCategory == 'relDateStoreName'">
            WHERE A.STORE_NAME LIKE '%' || #{searchValue} || '%'
                AND TO_CHAR(A.REL_DATE, 'YYYY-MM-DD') BETWEEN #{searchDate}
                AND #{searchDate2}
        </if>
        <if test="searchCategory == 'registDateStoreName'">
            WHERE A.STORE_NAME LIKE '%' || #{searchValue} || '%'
                AND TO_CHAR(A.REL_LIST_DATE, 'YYYY-MM-DD') BETWEEN #{searchDate}
                AND #{searchDate2}
        </if>
        ORDER BY A.REL_CODE DESC) B
        <![CDATA[
            WHERE RNUM <= #{endRow}
        ]]>
    ) A
    WHERE A.RNUM >= #{startRow}
    ORDER BY 1 ASC

```

## 출고내역 | 출고서작성

카테고리

연도. 월. 일.

연도. 월. 일.

지점명

검색

번호	지점	품목	금액	출고서등록일	출고완료일	출고상태
3	전농점	닭 9호(마일드) 외 1개 품목	57000	2022-09-24	2022-09-25	Y
2	전농점	닭 8호(마일드) 외 1개 품목	51000	2022-09-24		N
1	전농점	쌀파우더 외 1개 품목	48500	2022-09-24		N

Jquery의 slideToggle을 활용하여 상세조회

```
$(".detail-main").click(function () {
    $(this).next(".detail-sub-tr").find(".detail-sub").slideToggle(300);
});
```

번호	지점	품목	금액	출고서등록일	출고완료일	출고상태
3	전농점	닭 9호(마일드) 외 1개 품목	57000	2022-09-24	2022-09-25	Y

상품코드	썸네일	상품명	카테고리	규격	제조사	재고량	발주수량	출고수량	출고입력	요청일	출고일	출고상태	총매출단가
1004		닭 8호(핫)	닭고기	ea	하림	9930	5	5		2022-09-22	2022-09-25	Y	27000
1005		닭 9호(마일드)	닭고기	ea	하림	9968	5	5		2022-09-22	2022-09-25	Y	30000

## 출고내역 | 출고서작성

카테고리

연도. 월. 일.

연도. 월. 일.

지점명

검색

<input type="checkbox"/>	번호	지점	품목	금액	출고서등록일	출고완료일	출고상태							
<input type="checkbox"/>	상품코드	썸네일	상품명	카테고리	규격	제조사	재고량	발주수량	출고수량	출고입력	요청일	출고일	출고상태	총매출단가
<input type="checkbox"/>	3	전농점	닭 9호(마일드) 외 1개 품목	57000		2022-09-24	N							
<input type="checkbox"/>	1004		닭 8호(핫)	닭고기	ea	하림	9932	5	3	<input type="button" value="2"/> 	2022-09-22	N	27000	
<input type="checkbox"/>	1005		닭 9호(마일드)	닭고기	ea	하림	9930	5	5		2022-09-22	2022-09-25	Y	27000
<input type="checkbox"/>	2	전농점	닭 8호(마일드) 외 1개 품목	48500		2022-09-24	N							
<input type="checkbox"/>	1	전농점	쌀파우더 외 1개 품목											

클릭!

ajax를 이용하여 update

```
$(".rel-btn").on('click', function (e) {  
  
    let tr = $(this).parents('tr');      해당 위치에 출고버튼의 부모요소인 tr을 지정합니다.  
    let orderAmount = tr.children().eq(9).text();  
    let relSumAmount = tr.children().eq(10).text();      부모의 자식요소 중 발주량, 출고합계량, 출고량을 찾습니다.  
    let relAmount = tr.children().eq(11).find('.relAmount-input').val();  
  
    console.log("orderAmount : " + orderAmount);  
    console.log("relSumAmount : " + relSumAmount);  
    console.log("relAmount : " + relAmount);  
  
    orderAmount = parseInt(orderAmount);      발주량, 출고합계량, 출고량을 int 형태로 형변환합니다.  
    relSumAmount = parseInt(relSumAmount);  
    relAmount = parseInt(relAmount);  
  
    let sum = relSumAmount + relAmount;      출고합계량과 출고할 수량의 합계를 구합니다.  
  
    console.log("sum : " + sum);  
  
    if (sum > orderAmount) {      발주량보다 출고수량이 크게 되면 경고창을 출력하고 출고되지 않도록 막습니다.  
        alert("출고수량이 발주수량을 초과하였습니다.");  
        e.preventDefault();  
    } else {      출고가 가능하면 update 할 요소들의 위치를 지정합니다.  
  
        let form = $(this).parents('tr').children().eq(0)  
        let relForm = form.serialize();      Form 태그 내의 값들을 serialize로 한번에 받아옵니다.  
        console.log(relForm);  
  
        let itemAmount8 = tr.children().eq(8);  
        let relSumAmount10 = tr.children().eq(10);  
        let relDate10 = tr.children().eq(13);  
        let relYn14 = tr.children().eq(14);  
        let orderAmount9 = tr.children().eq(9)  
  
        let hddnItemAmount = tr.children().eq(11).children().eq(1);  
        let hddnRelSumAmount = tr.children().eq(11).children().eq(3);
```

```
$ajax({
    url: "/release/releaseItem",      ajax를 활용하여 페이지 새로고침 없이 출고이후 값을 변경합니다.
    type: "POST",
    data: relForm,
    contentType: 'application/x-www-form-urlencoded',
    // contentType: 'application/json',
    dataType: 'json',                controller에서 값을 ajax로 받아올때는 json형태로 받아옵니다.
    async : false ,                 출고 수량이 발주수량을 초과하는 값의 중복 업데이트 방지를 위하여 async를 false로 주어 동기화처리 하였습니다.
    success: function(result) {

        if(confirm("출고하시겠습니까?")){
            console.log("result : " + result);
            JSON.stringify(result);      출고확인창을 출력하여 true일시 controller에서 받아온 result 값으로 해당 요소들의 값을 변경해주었습니다.
            let itemAmount = result.itemAmount;
            let relDate = result.relDate;
            let relYn = result.relYn;
            let relSumAmount = result.relSumAmount;

            itemAmount8.html(itemAmount);
            relSumAmount10.html(relSumAmount);
            relDate10.html(relDate);
            relYn14.html(relYn);
            tr.children().eq(11).find('.relAmount-input').val("");

            hddnItemAmount.attr('value',itemAmount);
            hddnRelSumAmount.attr('value',relSumAmount);

            if(relSumAmount == orderAmount){
                relSumAmount10.css('color', 'green');
                orderAmount9.css('color', 'green');
            }

            alert("출고하였습니다.");
        }
    },
    error: function() {
        alert("실패하였습니다.")
    }
});
```

이벤트 버블때문에 stopImmediatePropagation을 이용하여 이벤트가 다른 요소에 연쇄적으로 발생하는것을 막았습니다.

```
e.stopImmediatePropagation();
});
```

```
@RequestMapping(value="/releaseItem", method=RequestMethod.POST)
@ResponseBody
public Object releaseItemInsertUpdate(@ModelAttribute RelItemDTO relItemDTO, HttpServletResponse response) {
    ajax로 부터 받은 값들은 @ModelAttribute를 사용하여 DTO형식으로 받아줍니다.
    response.setContentType("application/json");

    System.out.println("relItemDTO :" + relItemDTO);

    int itemAmount = Integer.parseInt(relItemDTO.getItemAmount());   String으로 넘어온 값을 int로 형변환 해주었습니다.
    int orderAmount = Integer.parseInt(relItemDTO.getOrderAmount());
    int itemNo = Integer.parseInt(relItemDTO.getItemNo());
    int relCode = Integer.parseInt(relItemDTO.getRelCode());
    int relCodeDetail = Integer.parseInt(relItemDTO.getRelCodeDetail());
    int relSumAmount = Integer.parseInt(relItemDTO.getRelSumAmount());
    int relAmount = Integer.parseInt(relItemDTO.getRelAmount());

    int amountUpdate = itemAmount - relAmount;
    int relSum = relAmount + relSumAmount;

    Map<String, Integer> itemAmountUpdate = new HashMap<>();   형변환 한 값을 자료형이 Map인 형태로 담아주었습니다.
    itemAmountUpdate.put("itemAmount", itemAmount);
    itemAmountUpdate.put("cartAmount", orderAmount);
    itemAmountUpdate.put("relAmount", relAmount);
    itemAmountUpdate.put("amountUpdate", amountUpdate);
    itemAmountUpdate.put("relSum", relSum);
    itemAmountUpdate.put("itemNo", itemNo);
    itemAmountUpdate.put("relCode", relCode);
    itemAmountUpdate.put("relCodeDetail", relCodeDetail);
```

```
int result1 = releaseService.itemAmountUpdate(itemAmountUpdate); Map에 담을값들을 parameter로 하여 본사 품목의 재고량을 update를 하였습니다.
```

```
int result4 = releaseService.itemHistoryInsert(relCode, itemNo); relCode와 itemNo를 parameter로 넘겨 출고할때마다 재고량 변화의 기록을 남기도록 insert를 진행하였습니다.
```

```
if(orderAmount != relSum) { 발주량과 출고수량의 총 합이 같지 않은경우 출고 및 출고기록이 insert 되도록 하였습니다.
```

```
int result2 = releaseService.relItemHistoryInsert(itemAmountUpdate);  
} else {
```

발주수량과 출고수량의 값이 같은경우에는 출고품목에 대한 출고일과 출고상태를 update 하였습니다.

```
int result3 = releaseService.releaseItemUpdateY(itemAmountUpdate);
```

```
List<ReleaseItemDTO> relItemSelectY = releaseService.relItemSelectY(relCode); 출고번호를 조건으로 출고품목들을 조회해옵니다.
```

```
int sum = 0;
```

```
for(int i = 0; i < relItemSelectY.size(); i++){
```

출고품목들의 개수만큼 반복문을 실행하여 품목들의 출고상태가 Y이면 sum을 1씩 증가시켰습니다.

```
if(relItemSelectY.get(i).getRelYn().equals("Y")){
```

```
    sum++;
```

```
} else {
```

```
    break;
```

```
}
```

```
System.out.println("sum :" + sum);
```

번호	지점	품목			금액	출고서등록일	출고완료일	출고상태					
상품코드	썸네일	상품명	카테고리	규격	제조사	재고량	발주수량	출고수량	출고입력	요청일	출고일	출고상태	총매출단가
1004		닭 8호(핫)	닭고기	ea	하림	9930	5	5		2022-09-22	2022-09-25	Y	27000
1005		닭 9호(마일드)	닭고기	ea	하림	9968	5	5		2022-09-22	2022-09-25	Y	30000

```
if(sum == relItemSelectY.size()){ 최종으로 증가된 sum 의 숫자(총 Y의 갯수)와 출고품목들의 숫자가 같아지면 출고서의 출고 상태가 완료로 update되도록 하였습니다.
```

```
int relYnResult = releaseService.relYnUpdate(relCode);
```

```
}
```

```
}
```

```
ReleaseOrderDTO relItemDetailAjax = releaseService.relItemDetailSelect(relCodeDetail);
```

출고상세품목의 고유번호인 relCodeDetail을 parameter로 해당 품목에 대해 조회합니다.

```
System.out.println("relItemDetailAjax : " + relItemDetailAjax); 조회한 값들을 형식에 맞게 형변환합니다.
```

```
int reItemAmount = relItemDetailAjax.getReleaseItemInfoDTO().getItemAmount();
```

```
int reRelSumAmount = relItemDetailAjax.getReleaseItemHistroyDTO().getRelSumAmount();
```

```
java.sql.Date reRelDate = relItemDetailAjax.getReleaseItemDTO().getRelDate();
```

```
String reRelYn = relItemDetailAjax.getReleaseItemDTO().getRelYn();
```

```
Map<String, Object> ajaxMap = new HashMap<>(); 형변환한 값들은 json 형태로 ajax로 보내기 위해 Map의 key : value 형태로 담았습니다.
```

```
ajaxMap.put("itemAmount", reItemAmount);
```

```
ajaxMap.put("relSumAmount", reRelSumAmount);
```

```
ajaxMap.put("relDate", reRelDate);
```

```
ajaxMap.put("relYn", reRelYn);
```

```
return ajaxMap;
```

출고상세품목의 품목코드를 ajax를 사용하여 조회합니다.

상세조회 툴은 Layer popup 형태로 만들었습니다.

카테고리

연도. 월. 일.

연도. 월. 일.

지점명

검색

번호	지점	품목	금액	출고서등록일	출고완료일	출고상태
3	전농점	 chicken stock				N
	상품코드	썸네일	상품명	상품명	제조사	출고상태
	클릭!		닭 8호(핫)	닭 8호(핫)	하림	총매출단가
	1004					N 27000
1005		닭 9호(마일드)				N 30000
2	전농점					N
1	전농점					N

상품명: 닭 8호(핫) | 상품 코드: 1004 | 제조사: 하림

매입단가: 4500 | 매출단가: 5400 | 판매규격: ea

재고량: 9935 | 발주수량: 5 | 출고수량: 0

**상품 상세 조회** | **수정** | **입고**

출고 기록이 없습니다.

부분출고시 출고기록이 생깁니다.

1차출고	2022-09-25	1
2차출고	2022-09-25	1
3차출고	2022-09-25	1
4차출고	2022-09-25	2

```
$(".open-btn").on('click', function (e){  
    let relCodeDetail = $(this).parents('tr').children().eq(11).find("input").eq(6).val();  
    let sendValue = "relCodeDetail=" + relCodeDetail;  상세품목의 고유품목번호를 찾아 json의 형태로 controller에 전달합니다.  
    console.log(relCodeDetail);
```

```
$.ajax({  
  
    url: '/release/releaseItemDetail',  
    type: 'get',  
    data: sendValue,  
    contentType: 'application/json',  
    async : false,  
    success: function(data){  
        const selectDetail = (JSON.parse(data));  controller에서 return받은 값을 json형태로 형변환 합니다.  
        console.table(data);  
        $(".form-control")[4].value = selectDetail.itemName;      상세조회의 value값을 변경시킵니다.  
        $(".form-control")[5].value = selectDetail.itemPurchase;  
        $(".form-control")[6].value = selectDetail.itemAmount;  
        $(".form-control")[7].value = selectDetail.itemNo;  
        $(".form-control")[8].value = selectDetail.itemSales;  
        $(".form-control")[9].value = selectDetail.orderAmount;  
        $(".form-control")[10].value = selectDetail.itemMake;  
        $(".form-control")[11].value = selectDetail.itemStandard;  
        $(".form-control")[12].value = selectDetail.relAmount;
```

```
$("#imgTab").empty();  Img 를 삭제 시키지 않으면 img가 계속 해서 추가되기 때문에 empty() 를 사용하였습니다.
```

```
$("#imgTab").attr({  
    "src":"/itemImage/"+selectDetail.fileName,  Img를 추가해줍니다.  
    "alt":"Image",  
    "class":"thum-detail",  
    "width":"200",  
    "height":"200"  
});
```

```
let $list = $("#detailList");
```

```
$list.html("");
```

부분출고 상세조회 부분을 먼저 삭제하고 이후 반복문을 사용하여 controller에서 조회해온 출고기록을 추가시켜줍니다.

```
for(let index in selectDetail.releaseItemHistroyDTO){
```

```
    if(selectDetail.releaseItemHistroyDTO[index].relDateDetail == null){
```

```
        $div = $("<div>");
```

```
        $noInput = $("<input class='form-control' style='text-align: center;'>").val("출고 기록이 없습니다.");
```

```
        $div.append($noInput);
```

```
        $list.append($div);
```

```
    } else {
```

```
        $div = $("<div>");
```

```
        $noInput = $("<input class='form-control listInput' >").val(parseInt(index) + 1 + "차출고");
```

```
        $dateInput = $("<input class='form-control listInput' >").val(selectDetail.releaseItemHistroyDTO[index].relDateDetail);
```

```
        $amountInput = $("<input class='form-control listInput' >").val(selectDetail.releaseItemHistroyDTO[index].relAmountDetail);
```

```
        $div.append($noInput);
```

```
        $div.append($dateInput);
```

```
        $div.append($amountInput);
```

```
        $list.append($div);
```

```
}
```

```
,
```

1차출고	2022-09-25	1
2차출고	2022-09-25	1
3차출고	2022-09-25	1
4차출고	2022-09-25	2

• Jihwan-Kim \*

```
@GetMapping("/releaseItemDetail")
@ResponseBody
public String storitemDetail(@RequestParam("relCodeDetail") String relCodeDetail, HttpServletResponse response)
    throws JsonProcessingException {

    response.setContentType("application/json; charset=UTF-8"); 조회결과를 전달할때 json의 형태로 전달합니다.

    System.out.println("relCodeDetail : " + relCodeDetail);

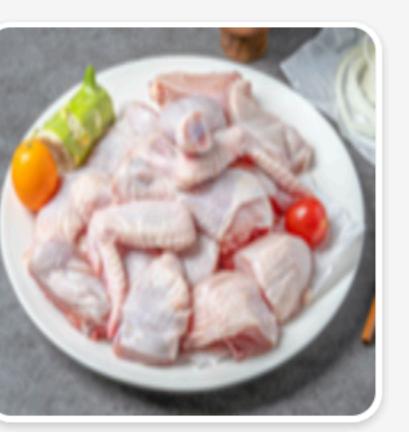
    ReleaseDetailDTO selectDetail = releaseService.selectDetail(relCodeDetail); Ajax로부터 전달받은 값으로 해당 상세품목을 조회합니다.

    System.out.println(selectDetail);

    ObjectMapper mapper = new ObjectMapper(); 조회결과를 json의 형태로 변환시킵니다.

    return mapper.writeValueAsString(selectDetail);
}
```

## 출고내역 | 출고서작성

번호	지점	품목	금액	출고서등록일	출고완료일	출고상태
3	전농점	 chicken stock	상품명: 닭 9호(마일드)   상품 코드: 1005   제조사: 하림 매입단가: 5000   매출단가: 6000   판매규격: ea 재고량: 9968   발주수량: 5   출고수량: 5	2022-09-25	2022-09-25	Y
2	전농점					N
1	전농점					N

상품명: 닭 9호(마일드) | 상품 코드: 1005 | 제조사: 하림  
매입단가: 5000 | 매출단가: 6000 | 판매규격: ea  
재고량: 9968 | 발주수량: 5 | 출고수량: 5

**클릭!** 수정 임고

페이지 이동시 url에 쿼리스트링 형태로 검색값을 넘겨주어  
페이지를 이동하면서 해당 품목이 검색되도록 구성하였습니다.

품목 관리 ←

번호	상품번호	썸네일	상품명	카테고리	규격	제조사	매입단가	매출단가	재고량	품절여부
1	1005		닭 9호(마일드)	닭고기	ea	하림	5000	6000	9968	● 보유

```

<script>
  $("#itemInfoFix-btn").on('click',function (){
    let itemName = $(".form-control")[4].value;
    location.href="/item/admin/list?searchCondition=itemName" + "&searchValue=" + itemName;
  });

```

<< < 1 > >>

## 출고내역 | 출고서작성

번호	지점	품목	금액	출고서등록일	출고완료일	출고상태
3	전농점	 chicken stock				Y
1004	닭 8호(핫)					Y 27000
1005	닭 9호(마일드)					Y 30000
2	전농점					N
1	전농점					

상품명: 닭 9호(마일드) 상품 코드: 1005 제조사: 하림  
 매입단가: 5000 매출단가: 6000 판매규격: ea  
 재고량: 9968 발주수량: 5 출고수량: 5

1차출고: 2022-09-25 수령: 5 입고: [클릭!]

페이지 이동시 url에 쿼리스트링 형태로 검색값을 넘겨주어  
페이지를 이동하면서 해당 품목이 검색되도록 구성하였습니다.

입고 등록

번호	상품코드	썸네일	상품명	카테고리	규격	제조사(원산지)	매입단가	매출단가	재고량	품절여부	액
	1005		닭 9호(마일드)	닭고기	ea	하림	5,000원	6,000원	9,968	● 보유	0원

```
$("#itemInfo-btn").on('click',function (){
    let itemName = $(".form-control")[4].value;
    location.href="/receive/admin/list/regist?searchCondition=itemName" + "&searchValue=" + itemName;
});
```

## 출고내역 | 출고서작성

출고서작성을 위해 가맹점 발주품목 을 조회합니다.

											연도. 월. 일.	□	~	연도. 월. 일.	□	지점명	검색
상품코드	발주번호	상품명	카테고리	규격	제조사(원산지)	매입단가	매출단가	발주수량	지점명	요청일	추가						
1004	60	닭 8호(핫)	닭고기	ea	하림	4500	5400	5	전농점	2022-09-22	<span style="color: blue;">+</span>						
1005	60	닭 9호(마일드)	닭고기	ea	하림	5000	6000	5	전농점	2022-09-22	<span style="color: blue;">+</span>						

추가 버튼을 누르면 임시출고서에 품목을 insert 합니다.



insert된 임시출고서 품목들을 조회합니다.

출고코드	2	등록일	2022-09-24	지점명	전농점	품목명	닭 8호(마일드) 외 1개 품목	총매출액	51000	등록	
상품코드	발주번호	상품명	카테고리	규격	제조사(원산지)	매입단가	매출단가	발주수량	지점명	요청일	추가
1003	60	닭 8호(마일드)	닭고기	ea	하림	4500	5400	5	전농점	2022-09-22	<span style="color: red;">-</span>
1002	60	닭 7호(핫)	닭고기	ea	하림	4000	4800	5	전농점	2022-09-22	<span style="color: red;">-</span>

```

@RequestMapping("/orderSelect")
public ModelAndView releaseOrderSelect(ModelAndView mv, Model model,
                                         @ModelAttribute ReleaseSelectCriteria releaseSelectCriteria){

    /* 발주서에 등록된 발주품목 조회 */
    List<ReleaseOrderDTO> orderList = releaseService.releaseOrderSelect(releaseSelectCriteria);

    /* 임시출고서에 등록된 출고품목 조회 */
    List<ReleaseOrderDTO> orderListN = releaseService.releaseOrderSelectN();

    /* 출고서 목록 조회 및 출고번호 생성 */
    List<ReleaseDTO> releaseDTO = releaseService.releaseDtoOrderSelect();
    int relCode = releaseDTO.size()+1;
    int intRelCode = relCode;

    /* 출고서 작성시 물품 총 금액 조회 */
    Integer totalMoney = releaseService.totalMoneySelect(intRelCode);

    mv.addObject( attributeName: "orderList", orderList);
    mv.addObject( attributeName: "orderListN", orderListN);
    mv.addObject( attributeName: "releaseSelectCriteria", releaseSelectCriteria);
    mv.addObject( attributeName: "totalMoney", totalMoney);
    mv.addObject( attributeName: "relCode", relCode);
    mv.addObject( attributeName: "releaseDTO", releaseDTO);

    mv.setViewName("release/release_order");

    return mv;
}

```

조회 결과를 타임리프 반복문으로 html 출력

```
|
    <form th:action="@{ /release/orderInsertUpdate }" method="post">
        <td th:text="${ order.releaseItemInfoDTO.itemNo }"></td>
        <td th:text="${ order.storeOrderDTO.orderNo }"></td>
        <td th:text="${ order.releaseItemInfoDTO.itemName }"></td>
        <td th:text="${ order.releaseItemCategoryDTO.categoryName }"></td>
        <td th:text="${ order.releaseItemInfoDTO.itemStandard }"></td>
        <td th:text="${ order.releaseItemInfoDTO.itemMake }"></td>
        <td th:text="${ order.releaseItemInfoDTO.itemPurchase }"></td>
        <td th:text="${ order.releaseItemInfoDTO.itemSales }"></td>
        <td th:text="${ order.releaseCartDTO.cartAmount }"></td>
        <td th:text="${ order.releaseCartDTO.storeName }"></td>
        <td th:text="${ order.storeOrderDTO.orderDate }"></td>
        <td class="plus-btns" style="...>
            <button type="submit" style="...>
            
            </button>
        </td>
    </form>

|  |

```

```
<!--발주서에 등록된 발주품목 조회 -->
```

```
<select id="releaseOrderSelect" resultMap="releaseOrderResultMap">  
    SELECT      발주서에 등록된 발주품목 조회  
        *  
    FROM CART A  
    JOIN ORDER_HANDLER B ON (A.CART_NO = B.CART_NO)  
    JOIN STORE_ORDER C ON (B.ORDER_NO = C.ORDER_NO)  
    JOIN ITEM_INFO D ON (A.ITEM_NO = D.ITEM_NO)  
    JOIN ITEM_CATEGORY E ON (D.CATEGORY_NO = E.CATEGORY_NO)  
    WHERE A.CART_YN = 'N'  
    <choose>          검색조건에 따른 조회  
        <when test="searchValue != null && !searchValue.equals('')  
            && searchDate != null && !searchDate.equals('')  
            && searchDate2 != null && !searchDate2.equals('')">  
            AND A.STORE_NAME LIKE '%' || #{searchValue} || '%'  
            AND TO_CHAR(C.ORDER_DATE, 'YYYY-MM-DD') BETWEEN #{searchDate}  
            AND #{searchDate2}  
        </when>  
        <when test="searchDate != null && !searchDate.equals('')  
            && searchDate2 != null && !searchDate2.equals('')">  
            AND TO_CHAR(C.ORDER_DATE, 'YYYY-MM-DD') BETWEEN #{searchDate}  
            AND #{searchDate2}  
        </when>  
        <when test="searchValue != null && !searchValue.equals('')">  
            AND A.STORE_NAME LIKE '%' || #{searchValue} || '%'  
        </when>  
    </choose>  
    ORDER BY A.CART_NO  
</select>
```

```
<!-- 임시출고서에 등록된 출고품목 조회 -->
```

```
<select id="releaseOrderSelectN" resultMap="releaseOrderResultMap">  
    SELECT DISTINCT      임시출고서에 등록된 품목 조회  
        *  
    FROM RELEASE_ITEM A  
    JOIN ITEM_INFO B ON(A.ITEM_NO = B.ITEM_NO)  
    JOIN ITEM_CATEGORY C ON(B.CATEGORY_NO = C.CATEGORY_NO)  
    JOIN ITEM_FILE D ON(B.ITEM_NO = D.ITEM_NO)  
    JOIN REL_ORD_HANDLER E ON(A.REL_CODE_DETAIL = E.REL_CODE_DETAIL)  
    JOIN CART F ON(E.CART_NO = F.CART_NO)  
    WHERE A.REL_LIST_YN = 'N'  
</select>
```

```
function find(){  
    console.log($("#searchDate").val());  검색에 대한 제약 조건 script  
    console.log($("#searchDate2").val());  
    if($("#searchDate").val() != null && $("#searchDate").val() != "" && $("#searchDate2").val() == ""){  
        alert("마지막일을 입력하세요.");  
        return false;  
    }  
  
    if($("#searchDate2").val() != null && $("#searchDate2").val() != "" && $("#searchDate").val() == ""){  
        alert("시작일을 입력하세요.");  
        return false;  
    }  
  
    if($("#searchDate").val() > $("#searchDate2").val()) {  
        alert("시작일이 마지막일보다 작을 수 없습니다.");  
        return false;  
    }  
}
```

```

@PostMapping("/orderInsertUpdate")
public ModelAndView releaseInsertUpdate(@ModelAttribute ReleaseItemInfoDTO releaseItemInfoDTO,
                                         @ModelAttribute ReleaseItemDTO releaseItemDTO,
                                         @ModelAttribute ReleaseCartDTO releaseCartDTO,
                                         @ModelAttribute StoreOrderDTO storeOrderDTO,
                                         @RequestParam("relCode") int relCode,
                                         ModelAndView mv) {

    int itemSales = Integer.parseInt(releaseItemInfoDTO.getItemSales());
    int orderAmount = releaseCartDTO.getCartAmount();
    int totalItemMoney = itemSales * orderAmount;
    int cartNo = releaseCartDTO.getCartNo();
    int relCodeDetail = releaseItemDTO.getRelCodeDetail();

    /* 임시출고서에 가맹점 발주 물품 등록 */
    int resultUpdate = releaseService.cartYnUpdateR(releaseCartDTO);
    int resultInsert = releaseService.releaseItemInsert(releaseItemInfoDTO, releaseItemDTO,
                                                       storeOrderDTO, releaseCartDTO, relCode, totalItemMoney);
    int resultInsert2 = releaseService.releaseInsertHandler(cartNo, relCodeDetail);

    mv.setViewName("redirect:/release/orderSelect");

    return mv;
}

```

```

<!-- 임시출고서 등록 -->
<update id="cartYnUpdateR">
    UPDATE CART|           임시출고서 등록 시 발주품목에 대한 상태값 변 쿼리문
    SET CART_YN = 'R'
    WHERE CART_YN = 'N'
    AND CART_NO = #{ cartNo }
</update>

<!-- 임시출고서 등록 -->
<insert id="releaseItemInsert">
    INSERT INTO RELEASE_ITEM|   임시출고서 등록 쿼리문
    (
        REL_YN, REL_AMOUNT, REL_CODE, ITEM_NO
        , ORDER_AMOUNT, REL_DATE, RELITEM_REQUEST_DATE
        , STORE_NAME, TOTAL MONEY, ORDER_NO
        , REL_CODE_DETAIL, REL_LIST_YN
    )
    VALUES
    (
        'N', null, #{ relCode }, #{ releaseItemDTO.itemNo }
        , #{ releaseCartDTO.cartAmount }, null, #{ storeOrderDTO.orderDate }
        , #{ releaseItemDTO.storeName }, #{ totalItemMoney }
        , #{ releaseItemDTO.orderNo }, #{ releaseCartDTO.cartNo }, 'N'
    )
</insert>

<!-- 임시출고서 등록 -->
<insert id="releaseInsertHandler">
    INSERT INTO REL_ORD_HANDLER|   임시출고품목과 발주품목을 연결해주는
    ( CART_NO, REL_CODE_DETAIL )|   테이블에 insert 하는 쿼리문
    VALUES
    ( #{ cartNo }, #{ cartNo } )
</insert>

```

```

@PostMapping("orderDeleteUpdate")
public ModelAndView releaseDeleteUpdate(@ModelAttribute ReleaseItemInfoDTO releaseItemInfoDTO,
                                       @ModelAttribute ReleaseItemDTO releaseItemDTO,
                                       @ModelAttribute ReleaseCartDTO releaseCartDTO,
                                       @ModelAttribute StoreOrderDTO storeOrderDTO,
                                       @RequestParam("relCode") int relCode,
                                       ModelAndView mv) {
    /* 임시출고서에 가맹점 발주 물품 제거 */
    int resultUpdate = releaseService.releaseItemUpdateN(releaseItemDTO);
    int resultUpdate2 = releaseService.cartYnUpdateN(releaseCartDTO);
    int resultDelete = releaseService.releaseItemDelete(releaseItemInfoDTO, releaseCartDTO,
                                                          releaseItemDTO, storeOrderDTO, relCode);

    mv.setViewName("redirect:/release/orderSelect");
    return mv;
}

```

```

<!-- 임시출고서 제거 -->
<update id="releaseItemUpdateN">
    UPDATE RELEASE_ITEM
        SET REL_LIST_YN = 'N'
        WHERE REL_LIST_YN = 'Y'
        AND REL_CODE_DETAIL = #{relCodeDetail}
</update>

<!-- 임시출고서 제거 -->
<update id="cartYnUpdateN">
    UPDATE CART
        SET CART_YN = 'N'
        WHERE CART_YN = 'R'
        AND CART_NO = #{cartNo}
</update>

<!-- 임시출고서 제거 -->
<delete id="releaseItemDelete">
    DELETE
        FROM RELEASE_ITEM
        WHERE ITEM_NO = #{releaseItemInfoDTO.itemNo}
        AND REL_CODE = #{relCode}
        AND RELITEM_REQUEST_DATE = #{storeOrderDTO.orderDate}
        AND STORE_NAME = #{releaseCartDTO.storeName}
</delete>

```

## 출고내역 | 출고서작성

연도. 월. 일.  ~ 연도. 월. 일.  지점명

상품코드	발주번호	상품명	카테고리	규격	제조사(원산지)	매입단가	매출단가	발주수량	지점명	요청일	추가
1004	60	닭 8호(핫)	닭고기	ea	하림	4500	5400	5	전농점	2022-09-22	<input type="button" value="+"/>
1005	60	닭 9호(마일드)	닭고기	ea	하림	5000	6000	5	전농점	2022-09-22	<input type="button" value="+"/>

등록시 form태그로 묶어놓은 값들로 insert 하여 출고서를 작성합니다.



클릭!

출고코드	2	등록일	2022-09-24	지점명	전농점	품목명	닭 8호(마일드) 외 1개 품목	총매출액	51000	<input type="button" value="등록"/>	
상품코드	발주번호	상품명	카테고리	규격	제조사(원산지)	매입단가	매출단가	발주수량	지점명	요청일	추가
1003	60	닭 8호(마일드)	닭고기	ea	하림	4500	5400	5	전농점	2022-09-22	<input type="button" value="-"/>
1002	60	닭 7호(핫)	닭고기	ea	하림	4000	4800	5	전농점	2022-09-22	<input type="button" value="-"/>

```
<script>
  임시 출고서에 품목이 등록될때마다 지점명 및 품목명 변경 script
  var tr1 = $("#tableY > tbody > tr:nth-child(1)");
  var td = tr1.children();
  var relName = td.eq(3).text();
  var storeName = td.eq(10).text();
  var lengths = $("#tableY > tbody > tr").length;

  switch (lengths) {
    case 0: document.getElementById("relName").value = ""; break;
    case 1: document.getElementById("relName").value = relName;
              document.getElementById("storeName").value = storeName; break;
    default : var lengths = lengths -1;
              document.getElementById("relName").value = relName + " 외 " + lengths + "개 품목";
              document.getElementById("storeName").value = storeName; break;
  }
}
```

지점명	전농점	품목명	닭 8호(마일드) 외 1개 품목

```

<div style="..." >
  <form th:action="@{ /release/releaseInsertUpdate }" method="post"> 출고서 작성시 form 태그 내 값들을 controller로 전달하여 insert 합니다.
    <input type="text" class="release-input title" value="출고코드" style="..." readonly>
    <input type="text" class="release-input" th:name="relCode" th:value="${ relCode }" style="..." readonly>
    <input type="text" class="release-input title" value="등록일" style="..." readonly>
    <input type="text" class="release-input" th:name="relListDate"
      th:value="${ #calendars.format(#calendars.createNow(), 'yyyy-MM-dd') }" style="..." readonly>
    <input type="text" class="release-input title" value="지점명" style="..." readonly>
    <input id="storeName" class="release-input" type="text" th:name="storeName" style="..." readonly>
    <input type="text" class="release-input title" value="품목명" style="width: 120px;" readonly>
    <input id="relName" class="release-input" type="text"
      th:name="relName" style="width: 230px; background-color: #f5f5f9" readonly>
    <input type="text" class="release-input title" value="총매출액" style="..." readonly>
    <input type="text" class="release-input" th:name="totalMoney" th:value="${ totalMoney }" style="..." readonly>
    <button class="btn btn-warning" type="submit" style="color:#fff; vertical-align: baseline;">>등록</button>
  </form>
</div>

```

```

❸ Jihwan-Kim*
@PostMapping(@RequestMapping("/releaseInsertUpdate"))
public ModelAndView releaseInsert(ModelAndView mv, @ModelAttribute ReleaseDTO relDto){
  System.out.println(relDto);
  /* 출고서 등록 */
  출고서 작성 insert 및 임시출고서 출고품목 상태 update
  int releaseInsert = releaseService.releaseInsert(relDto);
  int resultUpdate = releaseService.releaseItemUpdateF();

  mv.setViewName("redirect:/release/orderSelect");
  return mv;
}

```

<!-- 출고서 등록 -->

```

<insert id="releaseInsert"> 출고서 작성 insert 및 임시출고서 출고품목 상태 update
  INSERT INTO RELEASE
  (
    REL_CODE, REL_NAME, REL_DATE, REL_YN, STORE_NAME, TOTAL MONEY, REL_LIST_DATE
  )
  VALUES
  (
    ${ relCode }, ${ relName }, null, 'N', ${ storeName }, ${ totalMoney }, ${ relListDate }
  )
</insert>

<!-- 출고서 등록 -->
<update id="releaseItemUpdateF">
  UPDATE RELEASE_ITEM
  SET REL_LIST_YN = 'Y'
  WHERE REL_LIST_YN = 'N'
</update>

```