**알고리즘**

**<조회하는 폼>**

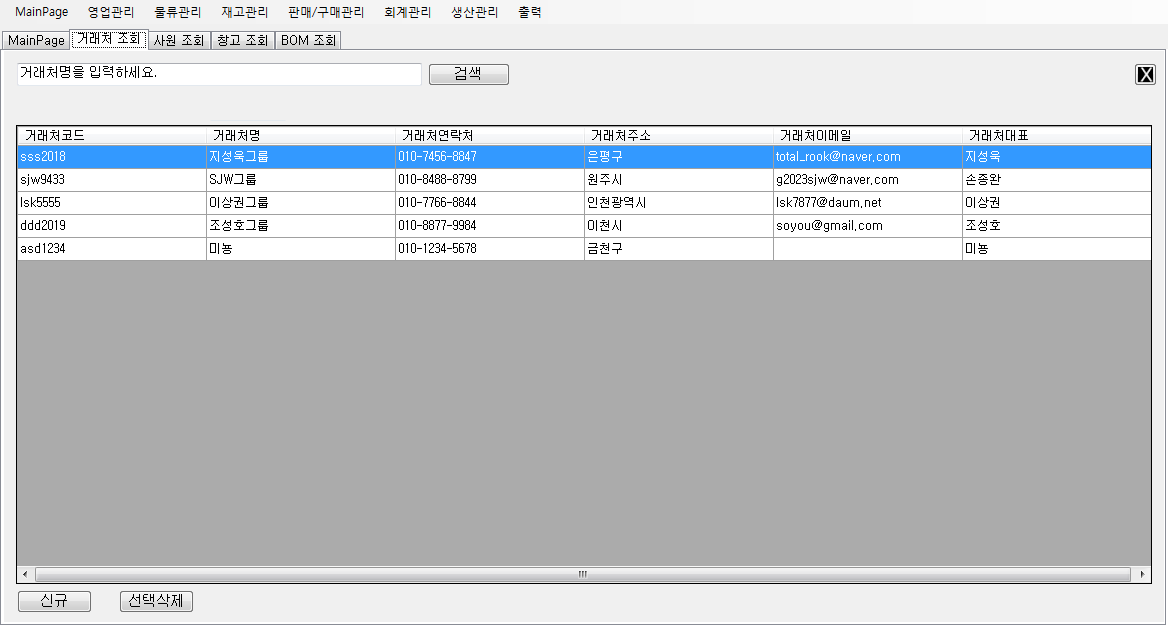
목록을 보여주는 폼들은 Load될 때 DAO를 통해 DB에 접속하여 Table의 내용을 읽어서 리스트에 저장한 후 DataGridView에 출력하는 식으로 되어있습니다.

일단 Table의 모든 컬럼을 읽어와서 리스트에 저장하고, 필요없는 컬럼이 존재한다면 리스트를 바탕으로 DataTable객체를 만들어 사용했습니다.

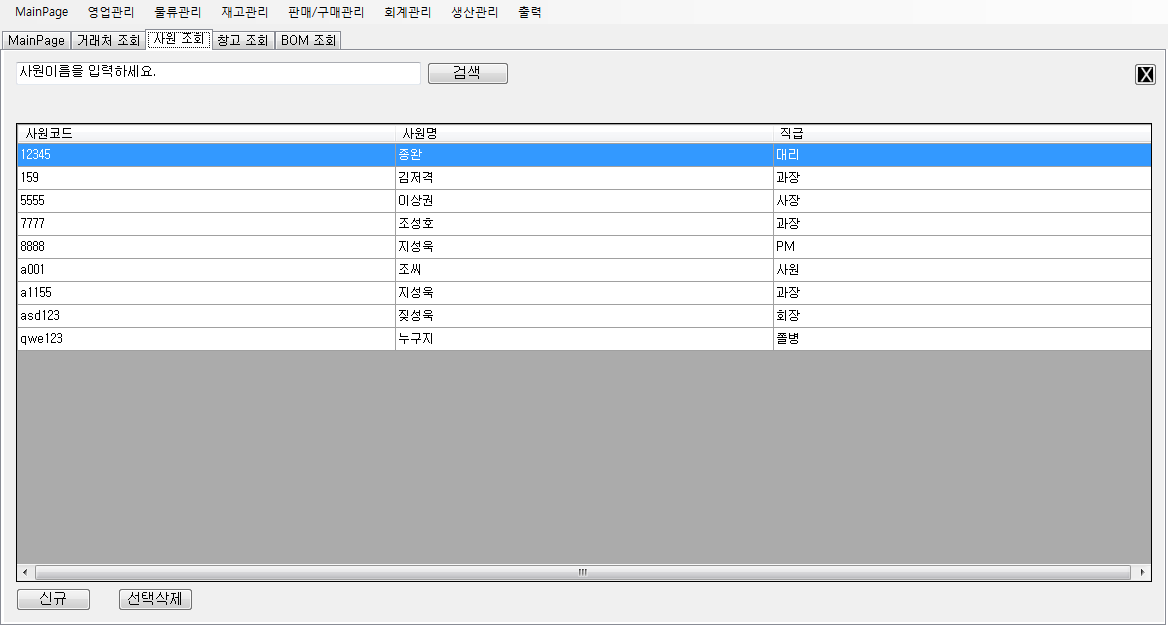
DataGridView에서는 하나의 행을 선택 후 ‘삭제’버튼을 누르게 되면 삭제할 것인지 한번 더 묻는 메시지박스가 생성되고 ‘예’와 ‘아니오’ 둘 중 무엇을 눌렀는지를 DialogResult변수를 하나 생성하여 판별하도록 했습니다.

예를 눌렀을 때에 DataGridView의 내용을 최신화하기 위해 Load될 때와 마찬가지로 DB에 접속하여 테이블을 내용을 다시 읽어옵니다.

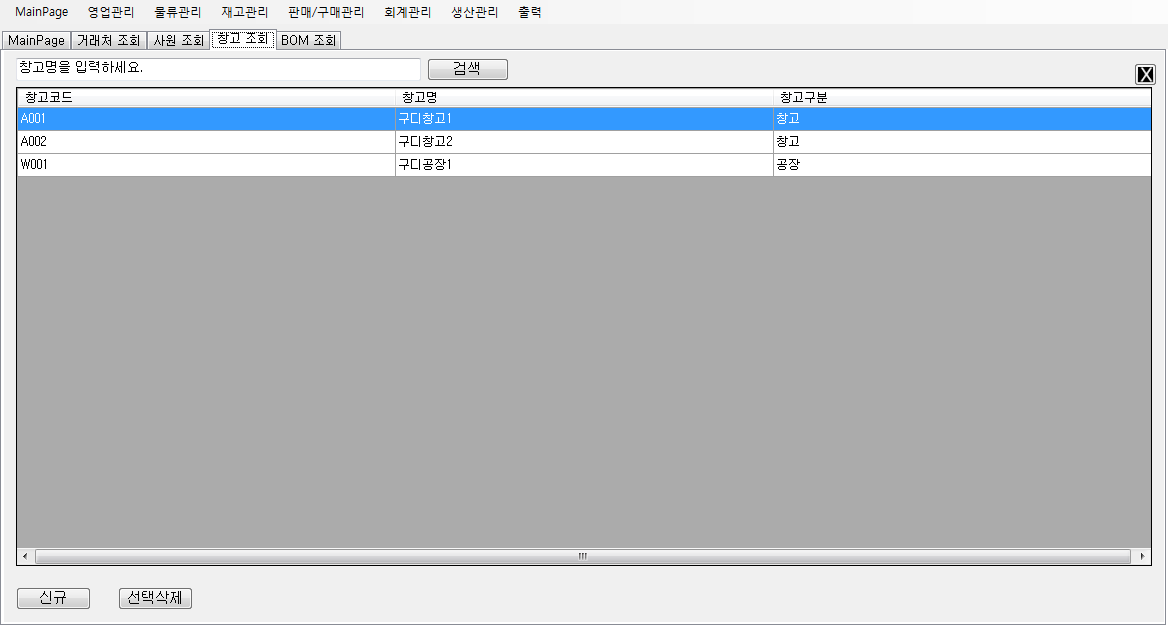
\* 거래처조회(Frm\_BusinessList)



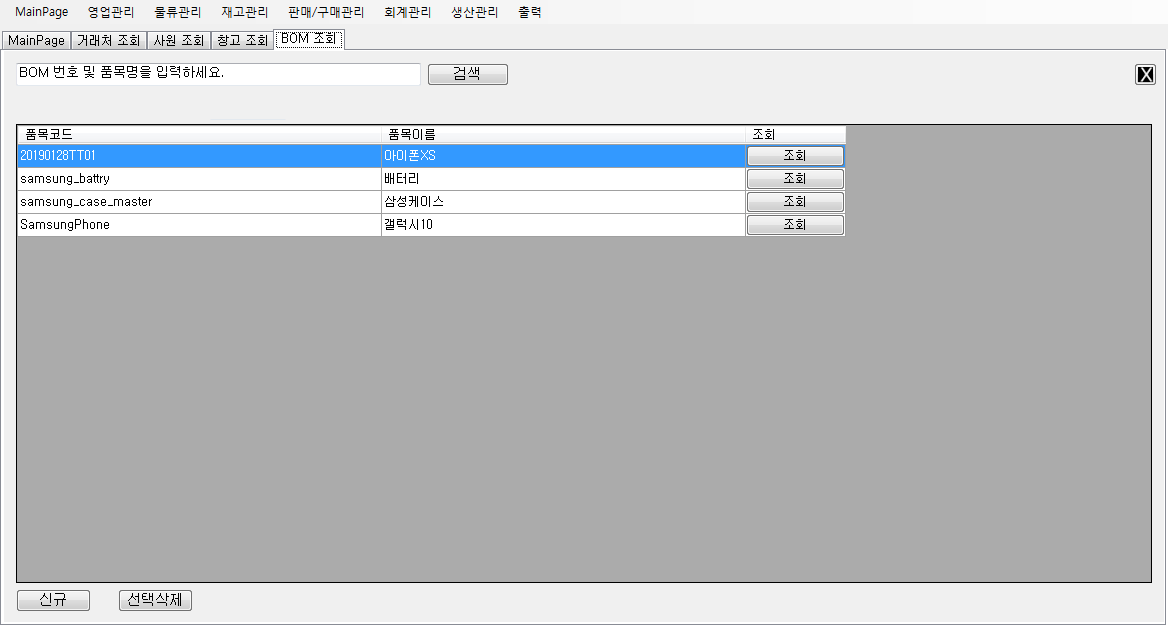
\* 사원조회 (Frm\_ClerkList)



\* 창고조회(Frm\_StockList)

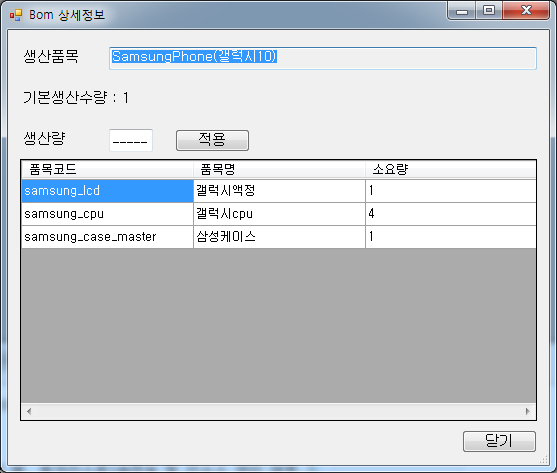


\* BOM조회(Frm\_BomList)

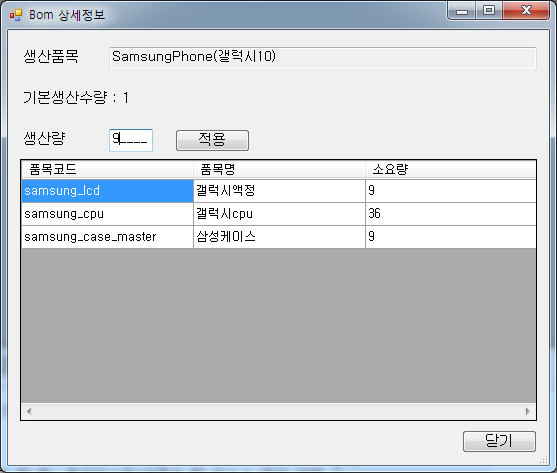


‘조회’버튼을 누르면 해당 품목의 코드를 매개변수로 넘겨서

BOM 상세정보(Frm\_BomDetail)폼이 Load될 때 DB에 접속하여 해당 품목을 만드는데 필요한 파츠정보를 얻어옵니다.



생산량 쪽에 0이상의 숫자를 입력하고 Enter또는 ‘적용’버튼을 클릭하면 그 숫자를 곱하여 소요량 컬럼의 내용이 변합니다.

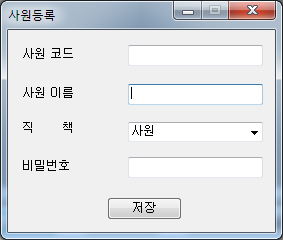


**<새로운 정보를 입력하는 폼>**

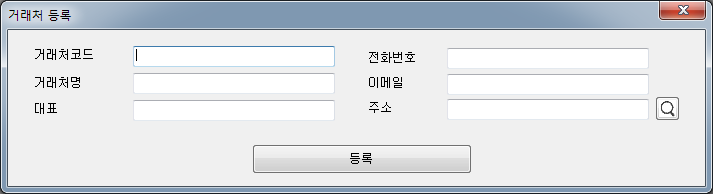
이 폼들은 Table정의에 맞게, 반드시 입력해야 받아야 하는 부분을 입력받지 않았을 경우를 조건문으로 처리해 두었습니다.

PK부분인 코드부분은 입력받은 내용으로 DB에서 Select할 때 where조건으로 넘겨주어 행이 존재하면 중복임을 알리는 메시지박스가 띄워지도록 했습니다.

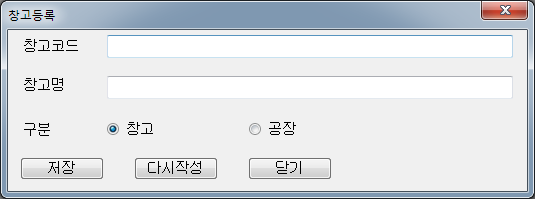
\* 사원등록(Frm\_ClerkInsert)



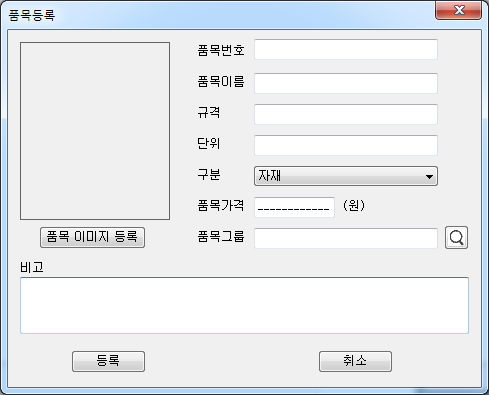
\* 거래처등록(Frm\_BusinessInsert)



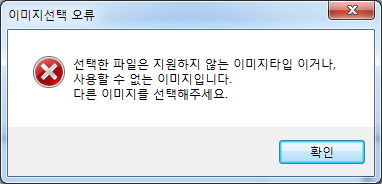
\* 창고등록(Frm\_StockInsert)



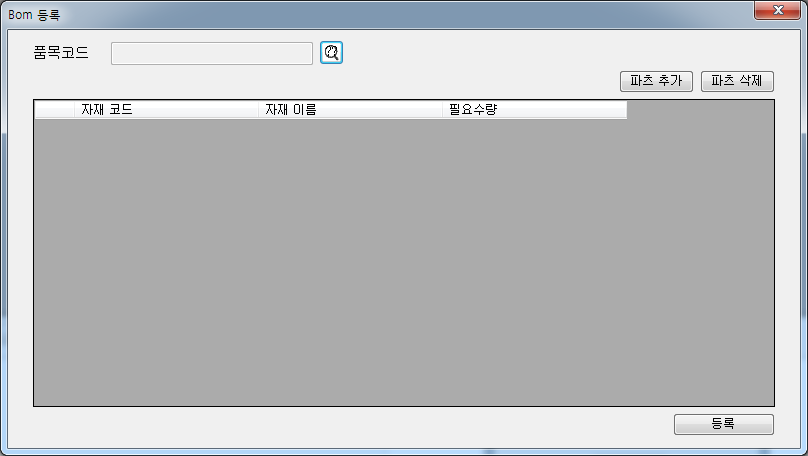
\* 품목등록(Frm\_ItemInsert)



품목등록(Frm\_ItemInsert)의 이미지의 경우 정규표현식을 이용하여 선택한 파일의 확장자가 제가 정해놓은 형식에 포함되지 않는다면 등록이 되지 않도록 했습니다.

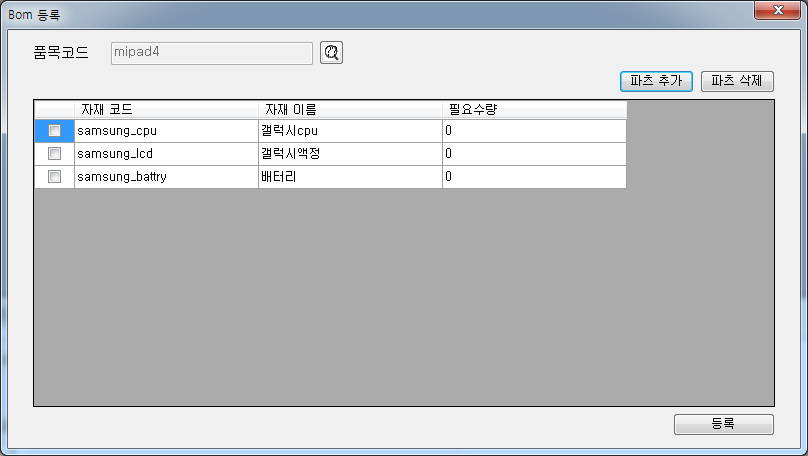


\* BOM등록(Frm\_BomInsert)



BOM등록(Frm\_BomInsert)은 다른 입력폼들과 형태가 약간 다른데

그 이유는 어떤 품목을 만들 때 들어가는 재료의 종류가 하나가 아니라 여러개일 수 있기때문에 그것을 처리하기 위함입니다.



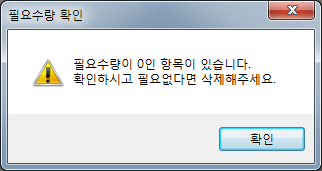
위와 같이 등록을 할 수 있습니다.

체크박스를 이용하여 선택된 행을 ‘파츠 삭제’버튼을 통해 삭제하거나 ‘파츠 추가’버튼을 눌러 파츠를 추가할 수 있습니다.

삭제를 할 때 for문을 사용했는데 여러 행을 선택 후 삭제를 진행할 때,

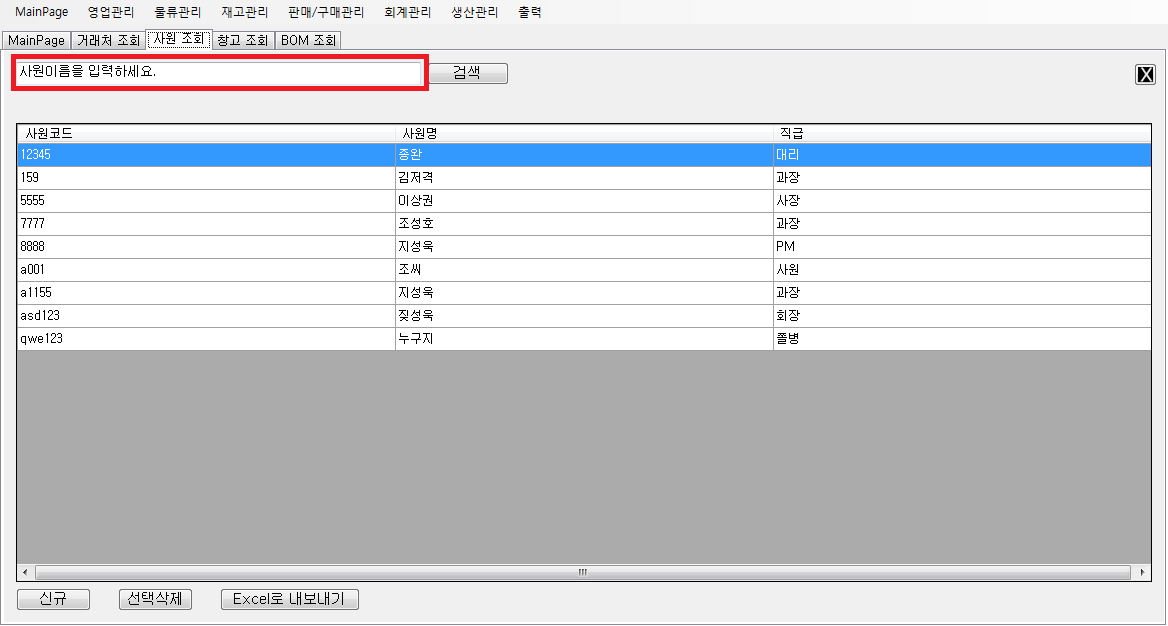
첫번째 선택행부터 지우게 되면 DataGridView의 행의 갯수가 자꾸 바뀌기 때문에 이를 int변수에 저장해놓고 증가식이 아닌 감소식을 이용해 마지막 행부터 지우도록 했습니다.

‘등록’버튼을 눌렀을 때 DataGridView의 모든 행을 검사하여 ‘필요수량’ 필드부분이 0으로 되어 있으면 필요없는 파츠이기 때문에 삭제해 달라는 메시지 박스를 띄웁니다.

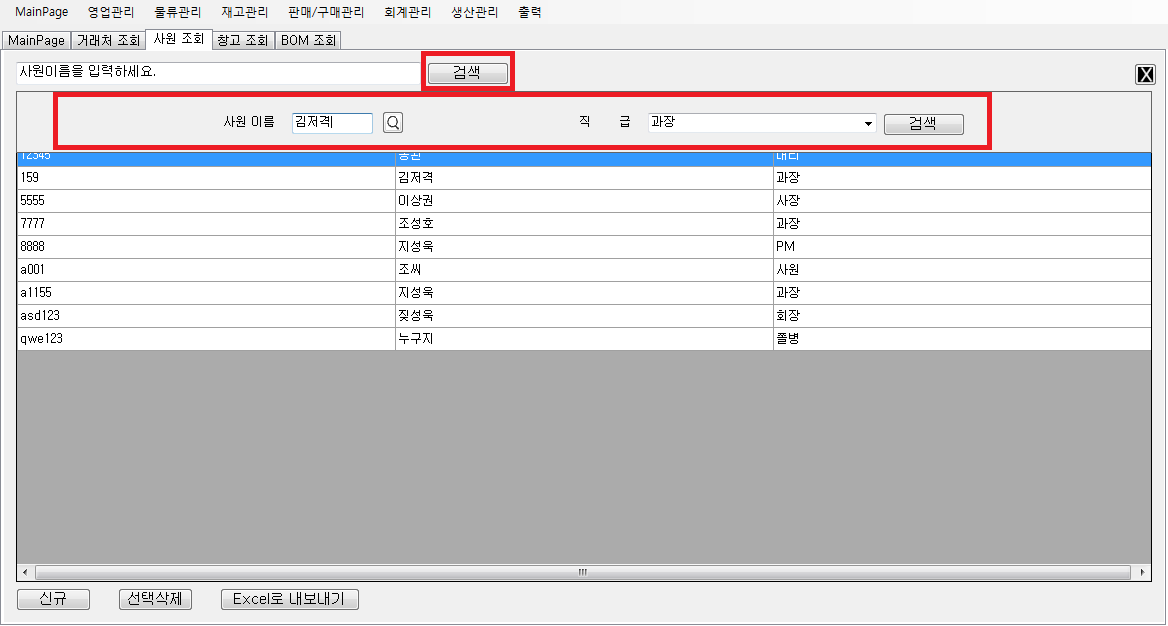


**<검색기능>**

조회하는 폼에서 텍스트박스에 검색하고자 하는 단어를 입력한 후 엔터를 누르거나,



‘검색’버튼을 클릭하면 판넬이 보이게 되는데 이를 통해 검색을 하면 저장프로시저를 이용하여 DB에서 해당 내용을 얻어와 DataGridView의 내용을 변경합니다.

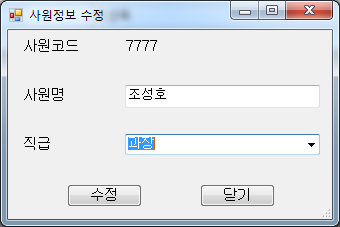


**<정보수정하는 폼>**

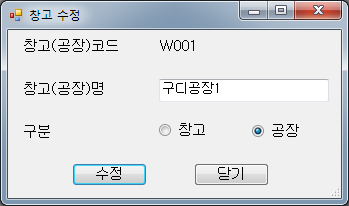
사원조회 (Frm\_ClerkList), 창고조회(Frm\_StockList), BOM조회(Frm\_BomList) 폼에서 DataGridView의 행을 더블클릭하면 해당 행의 정보를 수정할 수 있는 폼이 출력됩니다.

마찬가지로 수정을 눌렀을 때 ‘예’와 ‘아니오’를 선택하는 메시지 박스가 띄워지고 둘 중 무엇을 선택했는지를 DialogResult변수를 생성해 판별하고 ‘예’를 눌렀을 경우 수정폼이 Close되면서 DB에 접속하여 다시 Table의 내용을 읽어와 DataGridView의 내용을 최신화합니다.

\* 사원정보수정(Frm\_ClerkUpdate)



\* 창고수정(Frm\_WarehouseUpdate)



\* BOM수정(Frm\_BomUpdate)



**<선택한 행의 내용을 반환하는 폼>**

만약 보여지는 필드가 해당 테이블의 모든 내용이라면 선택된 행의 내용을 Class객체에 넣어 객체의 프로퍼티로 접근 가능하도록 했지만,

보여지는 필드가 해당 테이블의 내용의 일부라면 DataGridView의 내용을 채울 때 사용했던 리스트를 foreach문으로 반복하면서 ‘코드’컬럼과 일치하는 인덱스번째의 객체를 찾아 Class객체에 저장하고 이 객체의 프로퍼티로 접근할 수 있도록 만들었습니다.

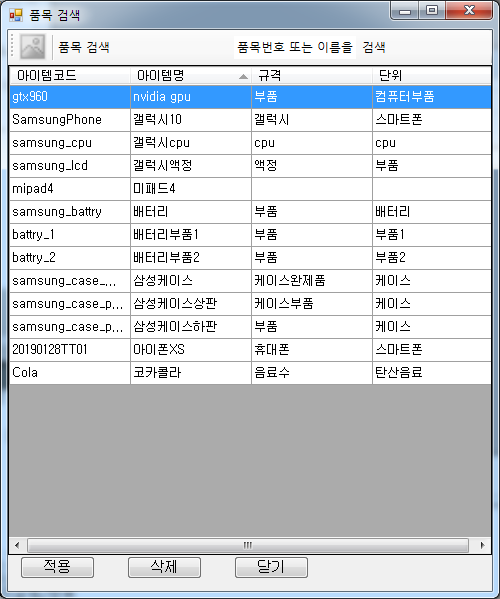
(폼의 형태가 다 비슷하므로 한장의 사진으로 대체합니다.)

\* 품목선택(Frm\_ItemSelect)

\* 창고선택(Frm\_WarehouseSelect)

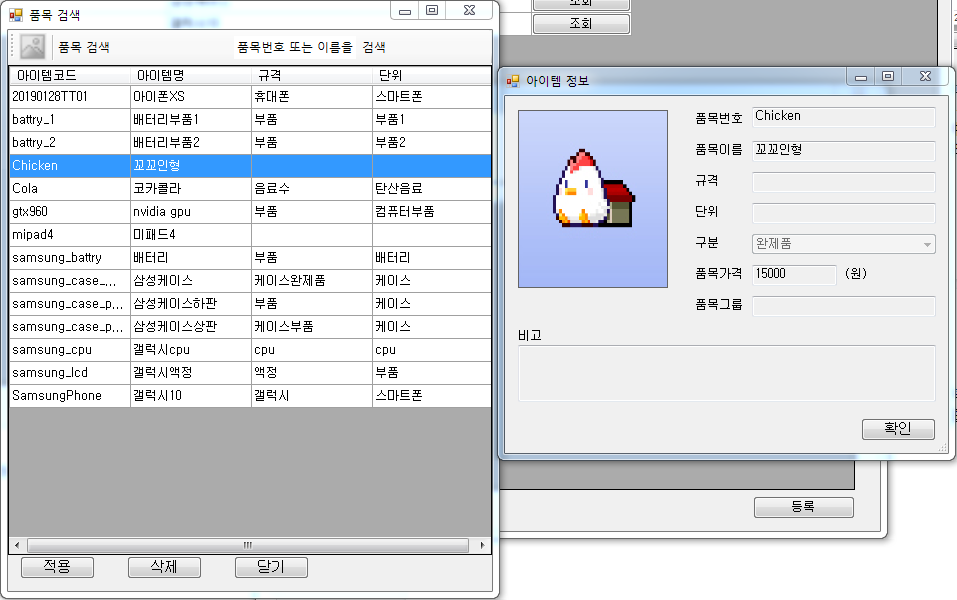
\* 거래처선택(Frm\_BusinessSelect)

\* 사원선택(Frm\_ClerkSelect)



품목선택(Frm\_ItemSelect)에는 다른 3개의 폼에는 없는 삭제버튼이 있는데 삭제버튼을 누르면 정말 지울것인지 묻는 ‘예’와 ‘아니오’ 메시지 박스가 출력되고 예를 눌렀을 경우에, 만약 해당 품목의 재고가 있다면 삭제할 수 없으므로 이를 try catch문으로 처리했습니다.

품목선택(Frm\_ItemSelect)에서 행을 더블클릭하면 DB에 접속하여 선택한 행의 아이템코드와 같은 행의 내용을 읽어서 새로운 폼인 품목정보(Frm\_ItemInfo)로 보여줍니다.



품목등록을 할 때 이미지를 바이트배열로 변환하여 저장했는데 이미지를 넣지 않으면 null값이 들어가고 이를 읽을 때는 바이트배열로 변환할 수 없기 때문에 try catch문을 사용해 읽은 값이 null이면 picturebox.Image를 null로 설정했습니다.

**순서도**

