MEDICAL INVENTORY MANAGEMENT

Date	**20-06-2025**
Team ID	**LTVIP2025TMID29579**
Project Name	**Medical Inventory Management**
Maximum Marks	

CHAPTER - 12
APPENDIX(apex Code):

```
trigger CalculateTotalAmountTrigger on Order_Item__c (after insert, after update,
after delete, after undelete) {
    // Call the handler class to handle the logic
    CalculateTotalAmountHandler.calculateTotal(Trigger.new, Trigger.old,
Trigger.isInsert, Trigger.isUpdate, Trigger.isDelete, Trigger.isUndelete);
}
```

```
  public class CalculateTotalAmountHandler {
   // Method to calculate the total amount for Purchase Orders based on related
  Order Items
      public static void calculateTotal(List<Order_Item__c> newItems,
  List<Order_Item__c> oldItems, Boolean isInsert, Boolean isUpdate, Boolean isDelete,
  Boolean isUndelete) {
        // Collect Purchase Order IDs affected by changes in Order_Item__c records
         Set<Id> parentIds = new Set<Id>();
         // For insert, update, and undelete scenarios
         if (isInsert || isUpdate || isUndelete) {
             for (Order_Item__c ordItem : newItems) {
                 parentIds.add(ordItem.Purchase_Order_Id__c);
             }
         }
                 // For update and delete scenarios
         if (isUpdate || isDelete) {
             for (Order_Item__c ordItem : oldItems) {
                 parentIds.add(ordItem.Purchase_Order_Id__c);
      }
    }
    // Calculate the total amounts for affected Purchase Orders
    Map<Id, Decimal> purchaseToUpdateMap = new Map<Id, Decimal>();
    if (!parentIds.isEmpty()) {
      // Perform an aggregate query to sum the Amount__c for each Purchase
Order
      List<AggregateResult> aggrList = [
        SELECT Purchase_Order_Id__c, SUM(Amount__c) totalAmount
        FROM Order_Item__c
        WHERE Purchase_Order_Id__c IN :parentIds
        GROUP BY Purchase_Order_Id__c
      ];
          // Map the result to Purchase Order IDs
      for (AggregateResult aggr : aggrList) {
        Id purchaseOrderId = (Id)aggr.get('Purchase_Order_Id__c');
        Decimal totalAmount = (Decimal)aggr.get('totalAmount');
        purchaseToUpdateMap.put(purchaseOrderId, totalAmount);
      }
      // Prepare Purchase Order records for update
      List<Purchase_Order__c> purchaseToUpdate = new
List<Purchase_Order__c>();
      for (Id purchaseOrderId : purchaseToUpdateMap.keySet()) {
        Purchase_Order__c purchaseOrder = new Purchase_Order__c(Id =
purchaseOrderId, Total_Order_cost__c =
purchaseToUpdateMap.get(purchaseOrderId));
        purchaseToUpdate.add(purchaseOrder);
      }
      // Update Purchase Orders if there are any changes
      if (!purchaseToUpdate.isEmpty()) {
        update purchaseToUpdate;
```

```
            }
        }
    }
}
```